

**VOLUME 1**

**Common Features of Application-Specific Functions**

**Application-Specific Functions**

---

**Discrete**

---

**AS-i Bus Setup**

---

**Man-Machine Interface**

---

**VOLUME 2**

**Counting**

**Application-Specific Functions**

**Axis Control**

**Stepper Motor Axis Control**

---

**VOLUME 3**

**Analog**

**Application-Specific Functions**

**PID Control**

**Process Control**

**Weighing**

---

**A**  
**B**  
**C**  
**D**



<b>Section</b>	<b>Page</b>
<b>1 General</b>	<b>1/1</b>
1.1 General	1/1
<b>2 Entering parameters and debugging application-specific functions</b>	<b>2/1</b>
2.1 Declaration of an I/O module or an integrated application-specific interface	2/1
2.2 General presentation of application-specific screens	2/3
2.3 Configuration	2/4
2.4 Adjustment	2/5
2.5 Debugging	2/6
<b>3 Objects associated with the application-specific functions</b>	<b>3/1</b>
3.1 General	3/1
3.2 Object addressing	3/2
3.2-1 In-rack I/O modules	3/2
3.2-2 Remote I/O modules	3/3
3.2-3 I/O objects on the AS-i bus	3/4
3.3 Addressing integrated application-specific interfaces	3/5
3.4 Addressing examples	3/8
3.5 Presymbolization	3/10
3.6 Implicit exchange objects	3/12

---

<b>Section</b>	<b>Page</b>
3.7 Explicit exchange objects	3/13
3.7-1 General	3/13
3.7-2 Reading status words and writing command words	3/16
3.7-3 Explicit reading and writing of adjustment parameters	3/17
3.7-4 Save/restore adjustment parameters	3/18
3.7-5 Exchange and report words	3/19
<b>4 Application-specific instructions</b>	<b>4/1</b>
4.1 General	4/1
4.2 Access to application-specific instructions	4/1
<b>5 Programming</b>	<b>5/1</b>
5.1 Principles	5/1
5.2 Access security management	5/2
5.3 Event processing	5/3
5.4 Operating modes	5/7
5.5 Processing application-specific faults by program	5/8
<b>6 Appendix</b>	<b>6/1</b>
6.1 Status words	6/1
6.2 Printing the parameters of I/O modules	6/2
<b>7 Index</b>	<b>7/1</b>

---

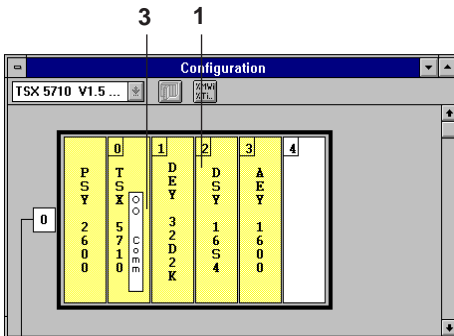
## 1.1 General

This part gives a general overview of the application-specific functions handled by PL7 software. The details of each application are dealt with in individual parts. An application-specific function is a control system function which interfaces between the control part (the PLC program) and the application (sensors, actuators and man-machine interface).

PL7 handles the following application-specific functions :

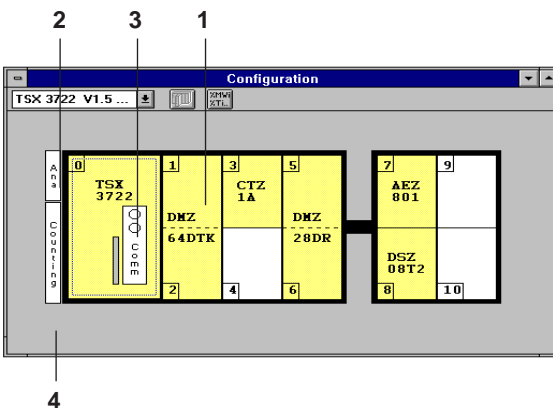
- Discrete I/O,
- Counting,
- Analog,
- Process control,
- Man-machine interface,
- Communication,
- Axis control (1),
- Stepper motor control (1),
- Weighing (1),
- Etc.

**Note** : There are hardware incompatibilities between the application-specific function modules of the TSX 37 and TSX 57. Their characteristics are very similar, if not identical.

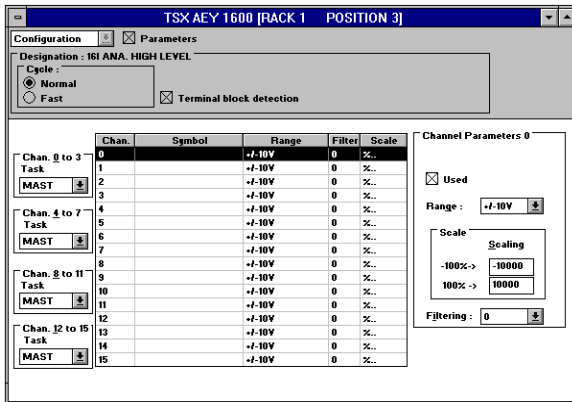


An application-specific function is executed by :

- a module **1**,
- or an integrated interface
  - analog interface **2** (2)
  - communication ports **3**
  - counter interface **4** (2)



- (1) Only the TSX 57  
(2) Only the TSX 37-22

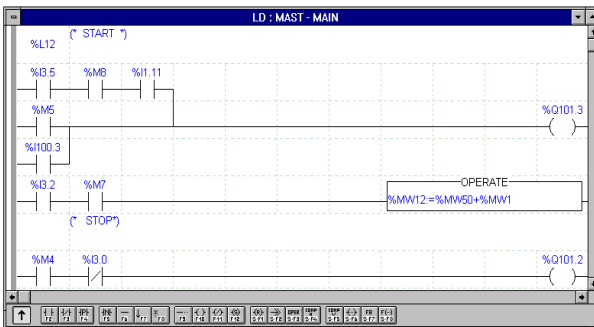


The software setup for an application-specific function is performed using :

- **screens** :
  - configuration screens,
  - adjustment screens,
  - debug screens.

Example :

Configuring a TSX AEY 1600 module : choice of filtering, operating range, measurement processing for an analog module, normal or fast cycle, terminal block detection.



- **Language objects** giving access via the program to the inputs and outputs of the module or integrated interface.

Examples :

- %I3.5 : input channel 5 of the module located in position 3 of rack 0,
- %I1.11 : input channel 11 of the module located in position 1 of rack 0,
- %I100.2 : input channel 2 of the module located in position 0 of rack 1 (1),
- %Q101.3 : output channel 3 of the module located in position 1 of rack 1 (1).

(1) For the TSX 57 only. The TSX 37 only has rack 0, see object and interface addressing in sections 3.2 and 3.3.

**Function call**

**Function Information:** Parameters

Family	Lib.V.	App.V.	Name	Comment
Character strings	1.1	-	CANCEL	Request to stop a current communication
Communication	1.5	-	DATA_EXCH	Send data and receive data
Dates, Times, Durations	1.1	-	INPUT_CHAR	Request to read character string
Digital conversions	1.1	-	OUT_IN_CHAR	Send and/or request to receive characters

**Call Format**

**Parameters of the PROCEDURE :**

Name	Type	Kind	Comment	Entry field
ADR	AR_w	IN	Address: ADR#[r..s] m.v.e or APP or SYS	
TYPE	w/ORD	IN	Exchange=1/Transmission=2/Reception=3	
EMIS	AR_w	IN	Data to send : %MWx:ll	

**Display the Call**

DATA\_EXCH (

OK Cancel

- **Application-specific instructions** can also be suggested.

Examples :  
 DATA\_EXCH transmission and/or reception of data for communication.  
 PID function for process control.

## Comment

The configuration/adjust/debug screens and the objects associated with an I/O module can be accessed via the software as soon as this module is declared in the configuration, without requiring a program line to be written (the debug screen can only be accessed in online mode).

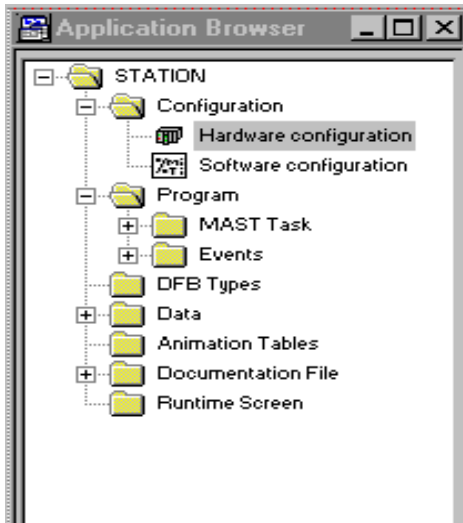
**Note :** In the case of the TSX 37, discrete I/O objects can be accessed by the PL7 program without having to declare modules (the software behaves as if the 32 I/O modules (16 inputs/16 outputs) have already been configured).





## 2 Entering parameters and debugging application-specific functions

### 2.1 Declaration of an I/O module or an integrated application-specific interface



An I/O module or an integrated application-specific interface is selected from the Configuration editor in **offline mode**.

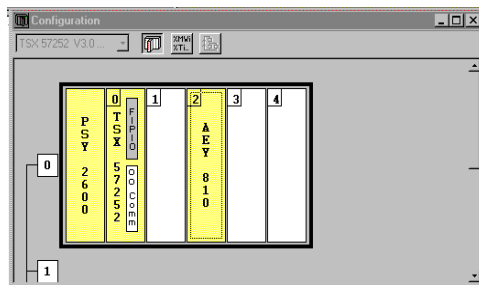
Access the editor via the Application Browser by clicking on the **Hardware configuration** icon.

If the Application Browser is not displayed :

- click on the Application Browser icon



- or select **Tools / Application Browser**

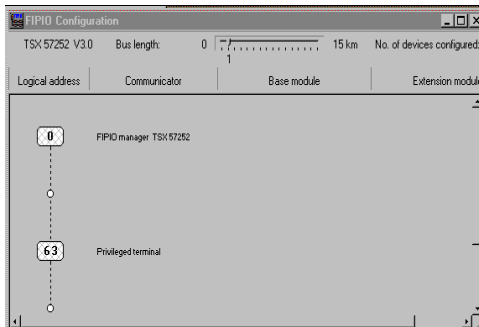


#### Choice of position

For in-rack modules :

Entails choosing the position where the I/O module is to be inserted (1).

Select, then confirm the position (double-click with the mouse).

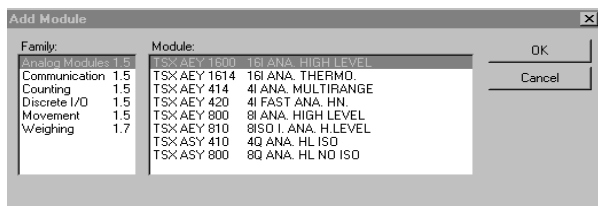


For remote modules on the FIPIO bus :

Entails selecting the FIPIO zone of the processor (TSX/PMX/PCX 57 •5) and selecting the connection point where the device is to be inserted.

Select, then confirm the position (double-click with the mouse).

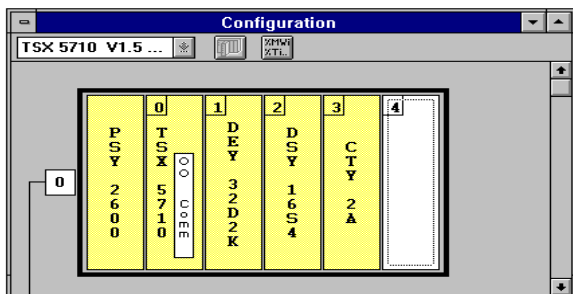
- (1) It is not necessary to declare interface functions, as their position is fixed and corresponds to position 0 on the TSX 37. The position of the communication ports is 0 or 1 depending on the number of slots occupied by the power supply (1 or 2) on the TSX 57.



### Choice of module type

In the dialog box choose the application-specific function (Family), then the relevant module for that application, then confirm with **OK**.

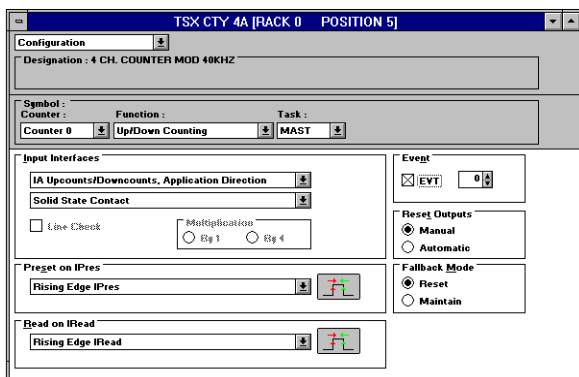
For a remote module : also enter the connection point number and select the communicator.



### Displaying the selected module

The reference of the module appears in the position chosen.

The module can have default parameters (this is the case for discrete I/O, analog or communication modules).



### Access to application-specific screens

Select then reconfirm the position (double-click with the mouse) to access the application-specific screens.

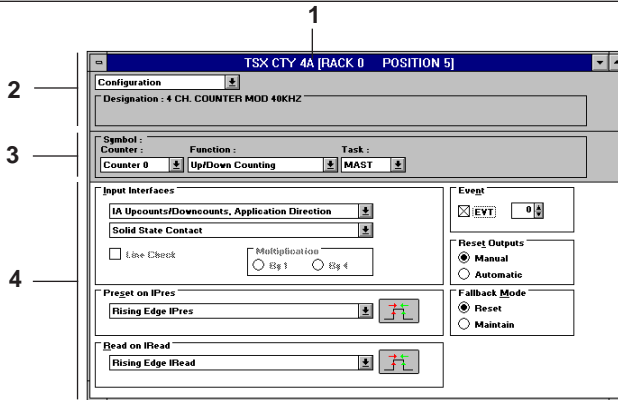
The user thus has access, depending on the module and mode (offline or online) to the application-specific screens for :

- Configuration
- Adjustment
- Debugging (online mode only).

### Note:

In the case of integrated interfaces, the function parameters can be accessed directly by selecting the function (double-click with the mouse).

## 2.2 General presentation of application-specific screens



- 1 Module catalog reference and position in the PLC (or connection point number).
- 2 Module zone : pulldown list for choosing the function to be performed :
  - **Configuration**, entry of configuration parameters offline or online (certain parameters cannot be modified online),
  - **Adjust**, display and modification of adjustment parameters offline or online (PLC running or stopped).  
The adjustment functions are, for certain modules, included in the configuration screen and the Debug screen (eg. Analog), or are nonexistent (eg. discrete I/O),
  - **Debug**, debugging the application-specific function online.
  - **Calibration**, for certain analog or weighing functions, in online mode.
- 3 Channel zone.
- 4 Channel parameters zone: contains the parameters of one channel (or several channels in the case of discrete and analog I/O).

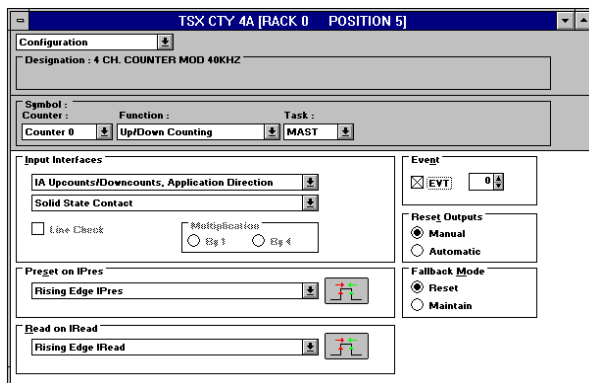
The **View** menu offers the following commands (depending on the module chosen) :

- Module zone : display or hide the module zone,
- Channel zone : display or hide the channel zone,
- Parameter zone : displays the parameter zone,
- Configuration/Adjust/Debug (as2),
- Zoom : access to detailed debugging functions (for the counting function),
- Input part/Output part for mixed discrete I/O modules  
Even part/Odd part for modules with more than 16 I/O (TSX 37).

**Note :**

**SHIFT F2** is used to move between zones within the screen.

## 2.3 Configuration



The module or integrated application-specific interface configuration function is used to select the operating characteristics.



Example for a counter module (TSX CTY 4A) :

- choice of the function assigned to a channel : upcounter, downcounter or up/down counter,
- choice of the task which updates the module I/O.
- event number,
- reactivation of outputs,
- fallback mode.

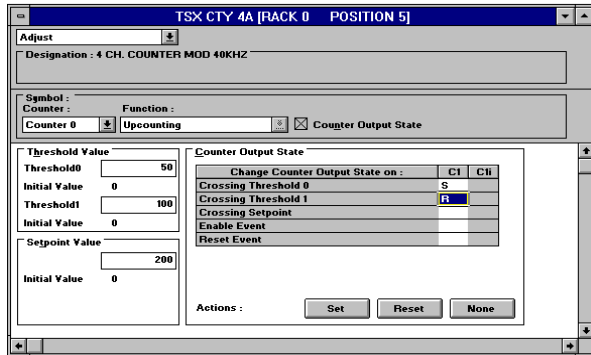
Discrete and analog I/O modules have a default configuration.

The configuration parameters can be modified in offline or online mode (if the application is in the non write-protected RAM memory). Certain parameters cannot be modified online (for example: task, event number). The configuration parameters cannot be modified by program.

### Notes :

- In offline mode, the configuration screen appears by default when the application-specific screen is accessed.
- In online mode, the debug screen appears by default.
- The **Confirm** command (or ) confirms the data entered in the screen and the **Undo** command (or ) cancels the data entered in the screen and returns to the last set of confirmed parameters.
- In offline mode, global confirmation of the configuration must be performed in the configuration editor so that the modifications are taken into account.
- In online mode, confirmation on the screen updates the configuration parameters in the PLC and reconfigures the module channel with its new parameters (the adjustment parameters take their initial value).

## 2.4 Adjustment



The module or integrated application-specific interface adjustment function is used to display and modify operating parameters which can be adjusted.

The application-specific functions which have an adjustment screen are :

- counting,
- axis control,
- stepper motor control,
- standard FIPIO profile,
- process control,
- etc.

**Note** : Analog and discrete I/O modules do not have specific adjustment screens. Adjustment parameters (filtering) are accessed from the configuration or debug screens by clicking with the right mouse button and selecting "Properties".

Example for a counter module :

- Threshold values,
- Setpoint values,
- Counter output state.

**Offline** : the parameters entered correspond to the initial parameters (value of parameters when first started or on a cold restart).

**Online** : the parameters entered correspond to the current parameters (they are lost on a cold restart, if they were not saved beforehand).

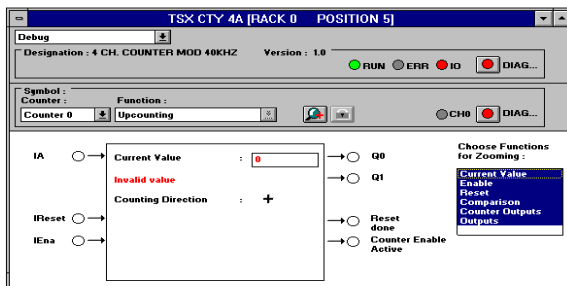
The **Save Parameters** command in the **Utilities** menu saves the current parameters (replaces the initial values with the current values) on the I/O module channel and in the PLC memory, if the application is in the RAM memory (not write-protected). The **Restore Parameters** command in the **Utilities** menu replaces the current values with the initial values on the I/O module channel and in the PLC memory.

### Comment

The **Save** and **Restore Parameters** commands can also be executed by program using the **SAVE\_PARAM** and **RESTORE\_PARAM** instructions, see section 3.7-4.

**Note** : In online mode, confirming the adjustment parameters on the screen updates the current parameters in the PLC and on the module channel.

## 2.5 Debugging



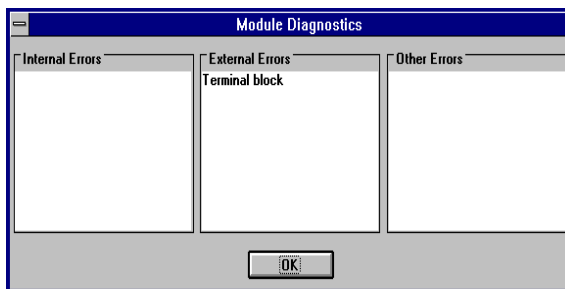
The module or integrated application-specific interface debugging function provides tools for assistance with debugging the application. It can be accessed online with the PLC stopped or running.

Example for a discrete I/O module:

- forcing to 0 or to 1,
- display of channel state.

The **Zoom** function in the **Utilities** menu provides detailed debugging for certain functions (function specific to counting).

The **DIAG** button is used to access the diagnostic screen associated with the module or the channel.



- **Internal faults** : I/O module fault (generally requires replacement of module).
- **External faults** : application fault (eg: range overshoot for an analog module).
- **Other faults** : module missing or not powered up, fault on one of the module channels.

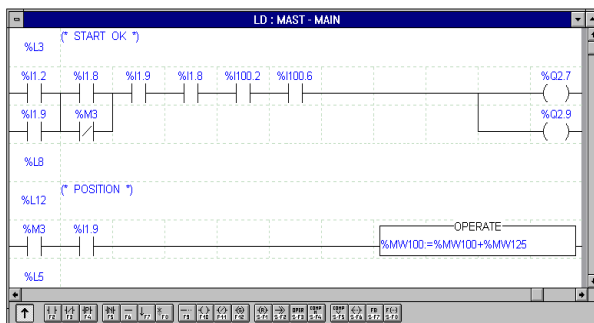
**Note :**

In online mode, the Debug screen appears by default when accessing the application-specific screen.

### 3 Objects associated with the application-specific functions

#### 3.1 General

An integrated application-specific interface or the addition of a module automatically enhances the application with the language objects for programming this interface or I/O module.



#### Implicit exchange objects

All the values of the I/O channels (bits, 16 or 32 bit words), as well as the associated fault bits are exchanged automatically on each scan of the task defined in the configuration of the module channels. These values are accessed directly using the associated objects.

##### Bits :

- %Ixy.i(.r) input channel bits,
- %Qxy.i(.r) output channel bits,
- %Ixy.i.ERR channel fault bit,
- %Ixy.MOD.ERR module fault bit.

##### Words :

- %IWxy.i(.r) input channels (16-bit words)
- %IDxy.i(.r) input channels (32-bit words)
- %QWxy.i(.r) output channels (16-bit words)

#### Explicit exchange objects

To process more detailed or specific information, it is possible to access other objects via language instructions (example : READ\_STS : read module status words).

##### Words :

- %MWxy.i(.r) 16-bit internal words,
- %MDxy.i(.r) 32-bit internal words,
- %MFxy.i(.r) 32-bit internal words,
- %MWxy.MOD (.r) 16-bit internal words.

##### Notes :

- Explicit exchange objects are only useful for advanced programming of the application.
- Objects %KWxy.i(.r)/ %KDxy.i(.r) can only be accessed in read mode. They correspond to the configuration parameters entered using the Configuration editor.
- Bits %Ixy.ERR, %Ixy.MOD.ERR and words %MWxy.MOD must not be written either by program or in adjust mode.

## 3.2 Object addressing

### 3.2-1 In-rack I/O modules

The addressing of the main bit and word objects of I/O modules mounted in racks is described in sections 1.2-2 (TSX 37) and 1.2-3 (TSX 57), part A.

**Note :**

%	I, Q, M, K	X,W,D or F	x	y	•	i	•	r
<b>Symbol</b>	<b>Type of object</b>	<b>Format</b>	<b>Rack</b>	<b>Position</b>		<b>Chan. no.</b>		<b>Rank</b>
	I = input Q = output M = internal variable K = internal constant	X = Boolean W = word D = double word F = floating point	x = rack number (1)	y = position number in the rack (2)		i = 0 to 127 or MOD		r = 0 to 255 or ERR

### Additional information

- **Type of object :**

**M** : read or write data exchanged at the request of the application.

**K** : configuration data, available in read only

Examples : %MW2.0.3 : status word 3 of channel 0 of the I/O module located in position 2 of rack 0.

%MW103.0.3 : status word 3 of channel 0 of the I/O module located in position 3 of rack 1

- **Channel no : i or MOD**

**MOD** : channel reserved for management of the module and the parameters common to all the channels that it supports.

Example : %I4.MOD.ERR : information on fault on module in position 4 of rack 0.  
%I102.MOD.ERR : information on fault on module in position 2 of rack 1.

- (1) In the case of a TSX 37, only one rack is possible, the rack address (0) does not appear, only the position (y) in the rack is displayed.  
For TSX\PMX\PCX 5710, the maximum number of racks is 2, therefore "x" = 0 or 1.  
For TSX\PMX\PCX 5720/25/30/35/40/45, the maximum number of racks is 8, therefore "x" varies between 0 and 7.
- (2) In the case of a TSX 57, if the rack number (x) ≠ 0, the position (y) is coded as 2 digits : 00 to 10 ; however if the rack number (x) = 0, non significant zeros are eliminated (elimination from the left) from "y" ("x" does not appear and "y" is as 1 digit for values from 0 to 9 and as 2 digits for values > 9).



- **Rank** : is used to indicate various objects of the same type associated with a single channel

Examples : %MW2.0.3 : word 3 of channel 0 of the I/O module located in position 2, rank 0 is omitted.

**ERR** : indicates a module or channel fault.

Example : %I204.MOD.ERR : information on fault on module in position 4 of rack 2.  
%I204.3.ERR : information on fault on channel 3 of module in position 4 of rack 2.

Variables			
<input type="checkbox"/> Parameters	I/O	Module Address	3
Address	Type	Symbol	
%CH3.MOD	CH		
%I3.MOD.ERR	BOOL		
%MW3.MOD	WORD		
%MW3.MOD.1	WORD		
%MW3.MOD.2	WORD		

The variables editor is used to access all the objects associated with a module or an integrated interface by selecting I/O from the pulldown list and the module position from the "Module address" zone.

All the variables associated with an application-specific function can be represented by symbols.

### 3.2-2 Remote I/O modules

Principle of addressing input and output image bits of remote I/O modules.

Syntax of the address or remote inputs and outputs :

%	I,Q,M,K	X,W or D	p.2.c	m	i	r
Symbol	Type of object	Format	Module/channel address and connection point	Module number	Channel number	Rank
	I = input Q = output M = internal variable K = internal constant	X = input W = mot D = double word F = floating point	p= 0 or 1 processor address 2= chan. no of integrated FIPIO link c= connection point number from 1 to 255	0= standard 1=extension	0 to 127 or MOD	0 to 255 or ERR

#### Example :

%I\0.2.6\0.5 : image bit of input 5 of the remote input base module located at connection point 6 of the FIPIO bus.

%Q\0.2.8\1.7 : image bit of output 7 of the remote output extension module located at connection point 8 of the FIPIO bus.

### 3.2-3 I/O objects on the AS-i bus

Principle of addressing input and output objects on the AS-i bus :

Syntax of the address of input and output objects on the AS-i bus :

<b>%</b>	<b>I or Q</b>	<b>\</b>	<b>xy.i</b>	<b>\</b>	<b>n</b>	<b>.</b>	<b>i</b>
Symbol	Type of object <b>I</b> = input <b>Q</b> = output		Module/channel address for TSX SAY 100 <b>x</b> = rack number <b>y</b> = position number <b>i</b> = channel number		Slave number <b>0 to 31</b>		Bit rank <b>0 to 3</b>

Note : on TSX 37 PLCs, xy.i =4.0

#### Example :

on TSX 57 :

%I\12.0\16.2 : input 2 of slave 16, with the TSX SAY 100 module located in slot 2 of rack 1 and the AS-i bus connected on channel 0.

on TSX 37 :

%Q\4.0\10.2 : output 2 of slave 10, with the TSX SAZ 10 module located in position 4 and the AS-i bus connected on channel 0.

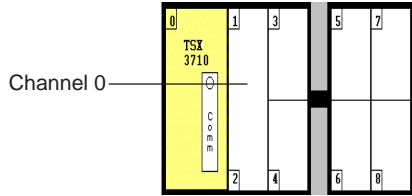
### 3.3 Addressing integrated application-specific interfaces

#### TSX 37 :

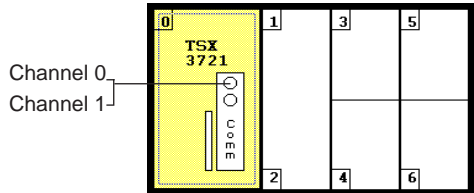
The integrated application-specific interfaces are all located in position 0.

#### Channel numbers :

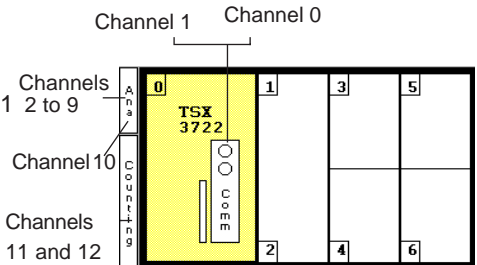
- On TSX 37-05/08/10 PLCs :
  - Terminal port = channel 0



- On TSX 37-21 PLCs :
  - Terminal port = channel 0
  - Communication interface = channel 1



- On TSX 37-22 PLCs :
  - Terminal port = channel 0
  - Communication interface = channel 1
  - 8 analog inputs = channels 2 to 9
  - 1 analog output = channel 10
  - 2 up/down counter channels = channels 11 and 12



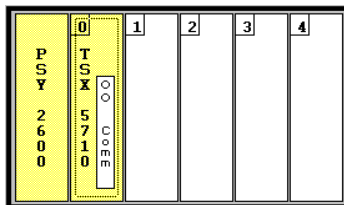
Example : %IW0.2 contains the channel 2 analog input measurement.

## TSX/PMX 57 :

### Channel numbers :

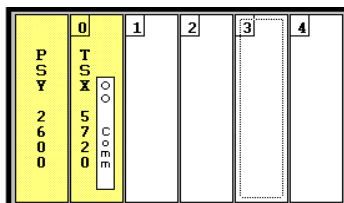
- On **TSX/PMX 57-10 PLCs** :

- Terminal port = channel 0
- Communication interface = channel 1



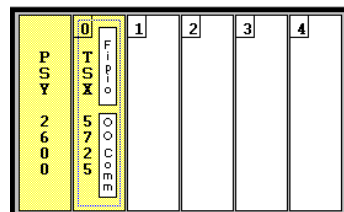
- On **TSX 57-20/30/40 and PMX 57-30 PLCs** :

- Terminal port = channel 0
- Communication interface = channel 1



- On **TSX/PMX 57-25/35/45 PLCs** :

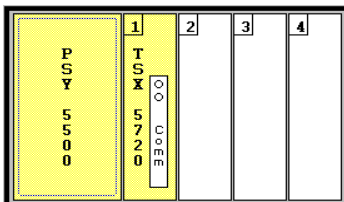
- Terminal port = channel 0
- Communication interface = channel 1
- FIPIO interface = channel 2



- On **TSX/PMX 57 PLCs with a double format power supply (2 positions)**

In this case, the processor is in position 1 :

- Terminal port = channel 1.0
- Communication interface = channel 1.1
- FIPIO interface = channel 1.2  
(for TSX/PMX 57-25/35/45)



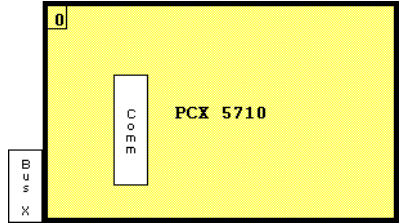
### Note :

The TSX 57 does not have analog or counter interfaces integrated in the processor.

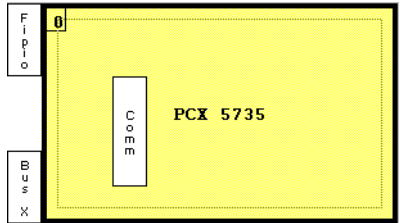
**PCX 57 :**

**Channel numbers :**

- **On PCX 57-10 PLCs :**
  - Communication interface = channel 1



- **On PCX 57-35 PLCs :**
  - Communication interface = channel 1
  - FIPIO interface = channel 2



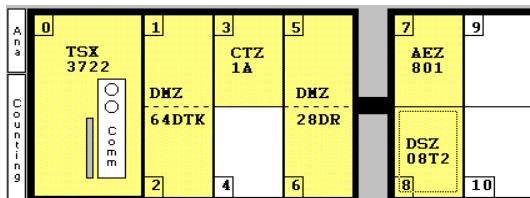
**Note :**

The PCX 57 does not have analog or counter interfaces integrated in the processor.

### 3.4 Addressing examples

#### TSX 37-22 :

- %IW0.8 contains the measurement of analog input channel 8 of the integrated analog interface.
- %I1.5 contains the state of the input on channel 5 of module 64DTK located in position 1 of the rack.
- %Q1.2 contains the state of the output on channel 2 of module 64 DTK located in position 1 of the rack.
- %IW7.1 contains the measurement of analog input channel 1 of module AEZ 801 located in position 7 of the rack.
- %Q8.5 contains the state of the output on channel 5 of module DSZ 08T2 located in position 8 of the rack.

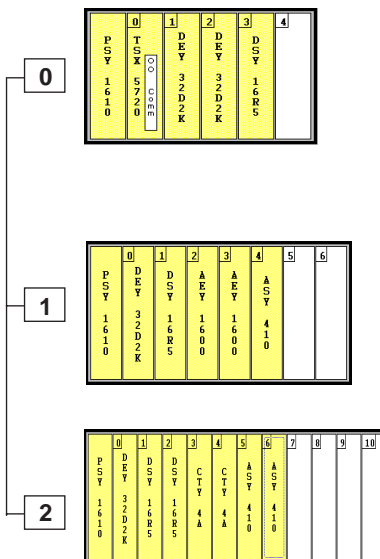


#### Note :

The position address varies between 0 and 10 for the TSX 37-21/22. The position address varies between 0 and 8 for the TSX 37-10. No rack addressing.

#### TSX 57-20 :

- %I1.3 contains the state of the input on channel 3 of module DEY 32D2K located in position 1 of rack 0.
- %Q101.2 contains the state of the output on channel 2 of module DSY 16R5 located in position 1 of rack 1.
- %IW102.1 contains the measurement of analog input channel 1 of module AEY 1600 located in position 2 of rack 1.
- %Q201.5 contains the state of the output on channel 5 of module DSY 16R5 located in position 1 of rack 2.

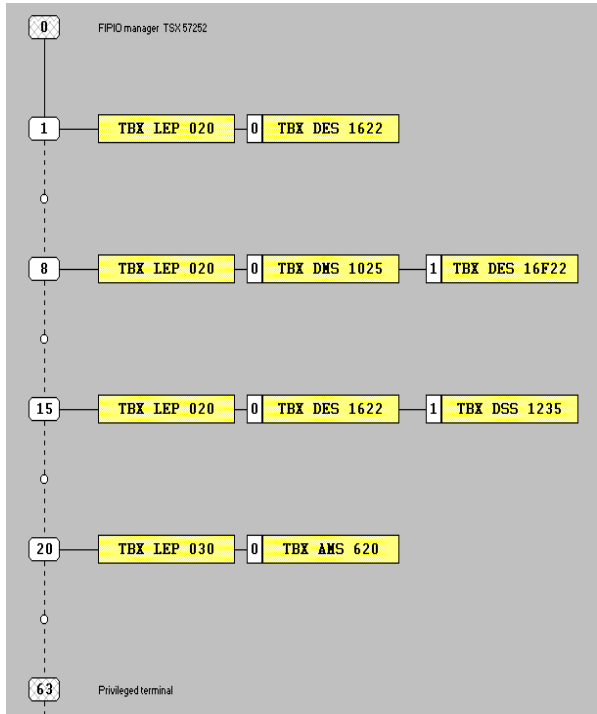


#### Note :

The maximum number of racks for a TSX/PMX/PCX 57-10 is 2, the rack address varies between 0 and 1. The maximum number of racks for a TSX/PMX/PCX 57-20/30/40 is 8, the rack address varies between 0 and 7. The position address varies between 00 and 10 (11 positions maximum).

**TSX 57-35 :**

- %I\0.2.1\0.2 contains the state of the discrete input on channel 2 of base module TBX ES 1622 located at connection point 1,
- %I\0.2.8\1.3 contains the state of the discrete input on channel 3 of extension module TBX DES 16F22 located at connection point 8,
- %Q\0.2.15\1.6 contains the state of the discrete output on channel 6 of extension module TBX DSS 1622 located at connection point 6,
- %Q\0.2.20\0.7 contains the state of the discrete output on channel 7 of base module TBX CSP 1625 located at connection point 20,
- %IW\0.2.30\0.1 contains the analog value (input word) of channel 1 of analog module TBX AMS 620 located at connection point 30.



### 3.5 Presymbolization

Application-specific modules provide a way of allocating symbols automatically to objects which are associated with them. The user gives the generic symbol for channel %CHxy.i of the module, and all the symbols for the objects associated with this channel can then be generated automatically on request.

These objects are symbolized using the following syntax :

#### User\_prefix\_Manufacturer\_suffix

where

The **User\_prefix** is the generic symbol given by the user to channel %CHxy.i (12 characters maximum).

The **Manufacturer\_suffix** is the part of the symbol which corresponds to the channel bit or word (20 characters maximum) given by the system.

In addition to the symbol, a manufacturer comment is generated automatically which gives a brief description of the role of the object.

**Example :** Auxil\_motor2\_param where "Auxil\_motor2" is the user prefix and "\_param" is the predefined manufacturer suffix.

The screenshot shows the 'PL7 PRO : CINZIA2 - [Variables]' window. The 'Parameters' tab is active, displaying a table of variables for the 'Auxil\_motor2' module. The table has columns for Address, Type, Symbol, and Comment. The variables listed include parameters like %CH9.0, %I9.0, %Q9.0, and %K9.0, as well as status bits like %I9.0, %I9.1, %I9.2, etc.

Address	Type	Symbol	Comment
%CH9.0	CH	Auxiliary_motor2	
%I9.0.6	WORD	Auxil_motor2_sync_nrun	Block SMOVE number in progress
%I9.0.2	DWORD	Auxil_motor2_w_pos	Measured position of the axis \$<VOIE>
%I9.0.4	DWORD	Auxil_motor2_speed	Measured speed of the axis \$<VOIE>
%I9.0.7	DWORD	Auxil_motor2_remain	---
%Q9.0.1	WORD	Auxil_motor2_mode_sel	Mode selection
%Q9.0.2	DWORD	Auxil_motor2_cmv	Speed modulation coefficient (Feedrate override p
%K9.0.1	WORD	Auxil_motor2_param	DIRDRV Parameter or INC, RP_HERE Paramete
%K9.0.2	WORD		
%K9.0.3	WORD		
%K9.0.8	WORD		
%K9.0.4	DWORD	Auxil_motor2_fmax	---
%K9.0.6	DWORD		
%I9.0	EBOOL	Auxil_motor2_next	Ready for next command block
%I9.1	EBOOL	Auxil_motor2_done	All instructions have been completed
%I9.2	EBOOL	Auxil_motor2_axflt	Fault on the axis \$<VOIE>
%I9.3	EBOOL	Auxil_motor2_axok	No fault on the axis \$<VOIE>
%I9.4	EBOOL	Auxil_motor2_hd_err	Hardware error on the axis \$<VOIE>
%I9.5	EBOOL	Auxil_motor2_ax_err	Presence of an error on the axis \$<VOIE>
%I9.6	EBOOL	Auxil_motor2_cmd_nok	Command refused
%I9.7	EBOOL	Auxil_motor2_nomotion	No motion on the axis \$<VOIE>
%I9.8	EBOOL	Auxil_motor2_at_pnt	Axis \$<VOIE> is in position
%I9.9	EBOOL		

#### Note :

The standard bits and words for application-specific modules can be presymbolized. It may be possible to presymbolize other objects, depending on the function selected.

Example of functions which offer symbolization of all objects : counting, axis control, stepper motor control, etc.



---

**List of suffixes for standard object symbols**

Object address	Suffix	Comment
%Ixy.ERR	_CH_ERROR	Channel error bit
%MWxy.0	_EXCH_STS	Standard status
%MWxy.0:X0	_STS_IN_PROGR	Exchange of status words in progress
%MWxy.0:X1	_COMMAND_IN_PROGR	Exchange of a command in progress
%MWxy.0:X2	_ADJUST_IN_PROGR	Exchange of an adjustment in progress
%MWxy.0:X15	_RECONF_IN_PROGR	Reconfiguration in progress
%MWxy.1	_EXCH_ERR	Standard status
%MWxy.1:X0	_STS_READ_ERR	Error while reading channel status
%MWxy.1:X1	_COMMAND_ERR	Error while sending a command on the channel
%MWxy.1:X2	_ADJUST_ERR	Error while adjusting the channel
%MWxy.1:X15	_RECONF_ERR	Error while reconfiguring the channel
%MWxy.2	_CH_FLT	Standard channel fault
%MWxy.2:X4	_INTERNAL_FLT	Internal channel fault or channel self-test
%MWxy.2:X5	_CONF_FLT	Different hardware and software configurations
%MWxy.2:X6	_COMMUNIC_FLT	Communication fault with PLC
%MWxy.2:X7	_APPLI_FLT	Applicatoin fault

**Presymbolization procedure**

Presymbolization is performed using the variables editor.

Select the I/O module, then double-click on the letter "p" preceding the %CH address of the channel to be symbolized. The screen suggests entering the prefix for the symbolization of objects for this channel.

**Note :** If a symbol has already been defined for the channel, the suggested prefix is the retrieved symbol abbreviated to 12 characters.

**Removing presymbolization**

Canceling presymbolization makes it possible, for a given logical channel, to delete all or some of the symbols of an object. There are 2 options :

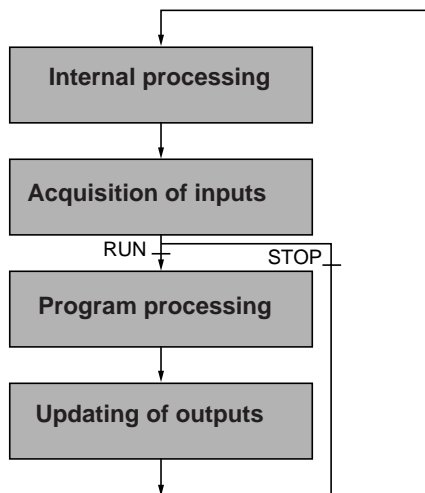
- **Delete All Presymbols** : No prefix is selected, and all the symbols are deleted (including those which the user may have modified directly in the editor).
- **Delete Presymbols With Prefixes**: The user indicates the prefix (for example, ANA) of the symbols to be deleted, in which case only the symbols of those objects which have the prefix are deleted.

### 3.6 Implicit exchange objects

These objects correspond to the images of the I/O module or the integrated application-specific interface.



The module or integrated interface channels are assigned by configuration to one of the PLC tasks (MAST,FAST).



The image bits and words (%I and %IW) of the input values of the module are updated implicitly in the PLC processor at the start of a task, whether the task is running or stopped.

The command bits and words (%Q and %QW) of the output values of the module are updated implicitly in the module by the processor at the end of the task, with the task running.

When the task is stopped, in accordance with the configuration chosen :

- fallback mode : the outputs are set to fallback position,
- maintain mode : the outputs maintain their last value.

#### Examples

- Discrete input module located in position 3 of rack 1 :  
%I103.1 gives the state of channel 1,
- Analog input module located in position 4 of rack 0 :  
%IW4.2 gives the analog value of channel 2,
- Counter module located in position 3 of rack 2 :  
%IW203.2 : X4 (bit n° 4 extracted from word %IW203.2) gives the state of the read input,
- Discrete output module located in position 6 of rack 3 :  
%Q306.5 controls the state of channel 5.

#### Fault bit

%Ixy.i.ERR : indicates whether channel i of the module in position y of rack x is faulty

#### Note :

In the case of the TSX 37, the rack number (x) does not appear.

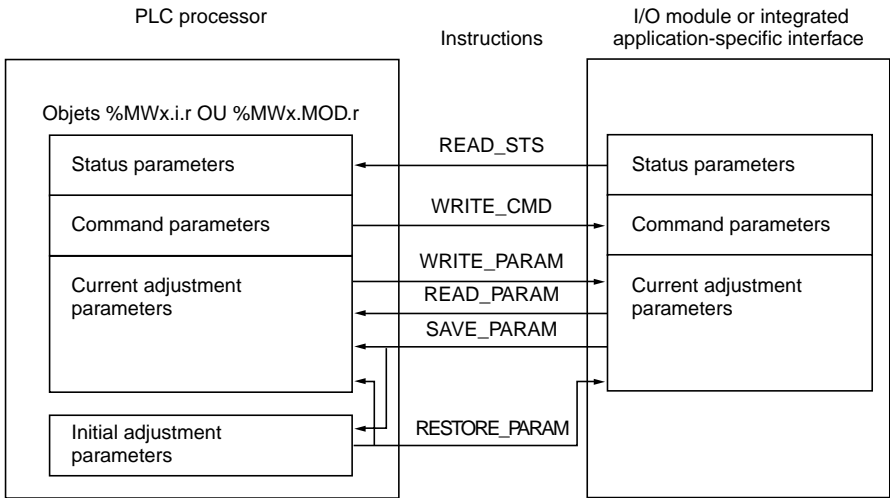
### 3.7 Explicit exchange objects

#### 3.7-1 General

These objects are not essential for the programming of an application-specific function. They provide information (eg : terminal block fault, module absent etc.) and additional commands for advanced programming of application-specific functions.

These words are only exchanged explicitly in a program by the execution of instructions, such as READ\_STS (read status words), WRITE\_CMD (write command words), WRITE\_PARAM (write adjustment parameters), READ\_PARAM (read adjustment parameters) etc.

Explicit exchanges are exchanges which are performed at the request of the user program, unlike implicit exchanges which are executed automatically at the start and end of each task.



%MW words are available at channel (%MWxy.i) and module (%MWxy.MOD) level.

---

### Limit on explicit exchanges on the FIPIO bus

TSX P57 25/35/45, PCX 5735/45 and PMX 57 35/45 PLCs have an integrated FIPIO channel. This means that a maximum of 24 explicit exchange functions can be activated simultaneously. An exchange request addressed on the FIPIO bus can take several master task scans, so the user must be able to manage the parameter words for exchange management for all exchanges of explicit variables (see part H of the communication manual).

### Logical channel %CHxy.i :

Explicit instructions apply to a series of %MW objects of the same type (status, command or parameter) of a single channel. The %CHxy.i channel is a general syntax for updating (using explicit instructions) all objects of the same type associated with this channel.

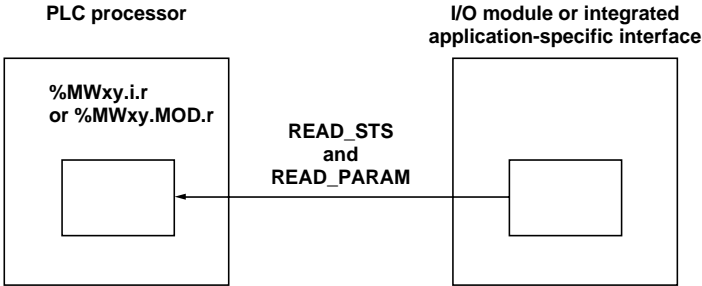
Example : READ\_STS %CH102.3 read status words for channel 3 of the module located at position 2 of rack 1.

### Note :

There is not necessarily one %CH object per channel. In the case of the analog application-specific function on the TSX 37, the address of the channel must be that of the first channel of each of the groups managed by the module : an explicit exchange on the first channel of the group managing the data associated with the other channels.

**General principle for using explicit instructions**

**Read instruction**

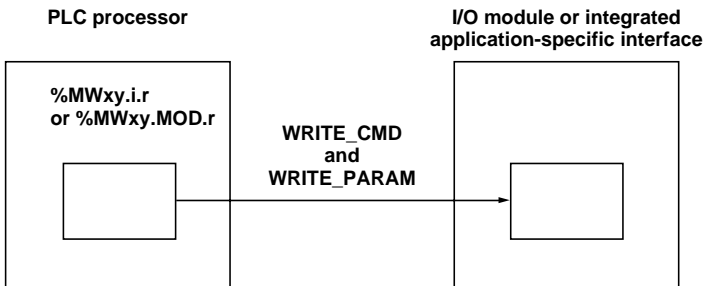


User program

- READ\_STS and READ\_PARAM instruction : triggers updating of %MW
- these objects can then be used by all the word instructions

(1) Only the READ\_STS instruction triggers updating of %MWxy.MOD.r words.

**Write instruction**



User program

- Write the %MW objects using word instructions
- WRITE\_CMD and WRITE\_PARAM instructions : triggers the writing of %MW to the module

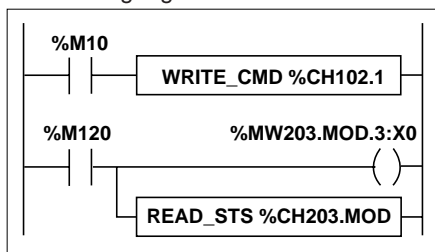
(2) Only the WRITE\_CMD instruction triggers the writing of %MWxy.MOD.r words.

### 3.7-2 Reading status words and writing command words

- **READ\_STS** : is for explicit reading in the module (or in the integrated interface) of status words associated with the module or the channel. This reading updates the %MW status words. Status words contain data on the operating status of the module or the channel. One of these words is standard (see appendices), the others are application-specific. They can be used to perform diagnostics via the program.
- **WRITE\_CMD** : is for explicit writing of the command words associated with the channel or the module to the module or integrated interface. These %MW words contain commands (eg : reset discrete outputs) for the module or the channel. They are application-specific.

#### Structure

Ladder language



Instruction list language

```
LD    %M10
[WRITE_CMD %CH102.1]

LD    %M120
ST    %MW203.MOD.3:X0
[READ_STS %CH203.MOD]
```

Structured Text language

```
If %M10 then
    WRITE_CMD %CH102.3 ;
End_If ;
If %M20 then
    SET %MW203.MOD.3:X0 ;
    READ_STS %CH203.MOD ;
End_If ;
```

#### Syntax

Function

**READ\_STS** %CHchannel address

Examples :

READ\_STS %CH102.1 read diagnostics data for counter channel 1 of the module located in position 2 of rack 1. This instruction triggers the updating of the diagnostics data in status words %MW102.1.2 and %MW102.1.3.

READ\_STS %CH203.MOD read general diagnostics information of the module located in position 3 of rack 2. This instruction triggers the updating of the diagnostics data in status word %MW203.MOD.2.

Function

**WRITE\_CMD** %CHchannel address

Example : WRITE\_CMD %CH3.4 write command data for channel 4 of the module located in position 3 of rack 0 (discrete output card).

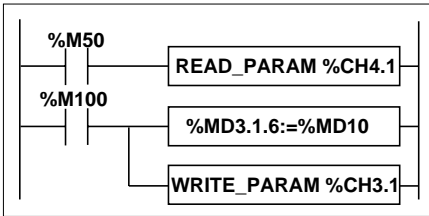
### 3.7-3 Explicit reading and writing of adjustment parameters

- **READ\_PARAM** : explicit reading of adjustment parameters in the module or integrated interface. This reading updates the %MWxy.i.r adjustment words.
- **WRITE\_PARAM** : explicit writing of adjustment parameters to the I/O module or the integrated interface.

This instruction is used for modifying the adjustment values defined at configuration by the program (reminder : these modified values are replaced by the initial parameters at a cold restart).

#### Structure

Ladder language



Instruction list language

```
LD    %M50
[READ_PARAM %CH4.1]

LD    %M100
[%MD3.1.6 := %MD10]
[WRITE_PARAM %CH3.1]
```

Structured Text language

```
If %M50 then
    READ_PARAM %CH4.1 ;
End_If ;
If %M100 then
    %MD3.1.6 := %MD10 ;
    WRITE_PARAM %CH3.1 ;
End_If ;
```

#### Syntax

Function

```
READ_PARAM %CHchannel address
```

Example : READ\_PARAM %CH4.1 read adjustment parameters of counter channel 1 of the module located in position 4 of rack 0.

Function

```
WRITE_PARAM %CHchannel address
```

Example : WRITE\_PARAM %CH3.1 write adjustment parameters of counter channel 1 of the module located in position 3 of rack 0.

#### Note :

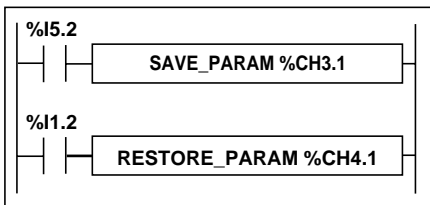
The following bits can be used to control the writing of parameters to the module :  
 %MWxy.i.0:X2 is set to 1 when the exchange is running and reset to 0 when writing is finished.  
 %MWxy.i.1:X2 is set to 1 if the parameters transmitted are outside limits or incorrect.  
 (see Appendix for further details).

### 3.7-4 Save/restore adjustment parameters

- **SAVE\_PARAM** : explicit save of adjustment parameters of the module or the integrated interface. These parameters replace the initial values defined using the configuration editor (or in the last save).
- **RESTORE\_PARAM** : explicit restore of the initial adjustment parameters (written during configuration or the last save).

#### Structure

Ladder language



Instruction list language

```
LD    %I5.2
[SAVE_PARAM %CH3.1]

LD    %I1.2
[RESTORE_PARAM %CH4.1]
```

Structured Text language

```
If %I5.2 then
    SAVE_PARAM %CH3.1;
End_If ;
If %I1.2 then
    RESTORE_PARAM %CH3.1 ;
End_If ;
```

#### Syntax

Function

**SAVE\_PARAM** %CHchannel address

Example : **SAVE\_PARAM** %CH3.1 restores the adjustment parameters of counter channel 1 of the module located in position 3 of rack 0.

Function

**RESTORE\_PARAM** %CHchannel address

Example : **RESTORE\_PARAM** %CH4.1 restores the adjustment parameters of counter channel 1 of the module located in position 4 of rack 0.

#### Comment :

The **SAVE\_PARAM** and **RESTORE\_PARAM** instructions are equivalent to the **Save Parameters** and **Restore Parameters** commands in the **Utilities** menu, in the adjustment function.

#### Note :

The following bits can be used to control writing of parameters in the module :  
 %MWxy.i.0:X2 is set to 1 during the exchange and reset to 0 when writing is completed.  
 %MWxy.i.1:X2 is set to 1 if the parameters transmitted are out of limits or incorrect.  
 (See Appendix section 6).



### 3.7-5 Exchange and report words

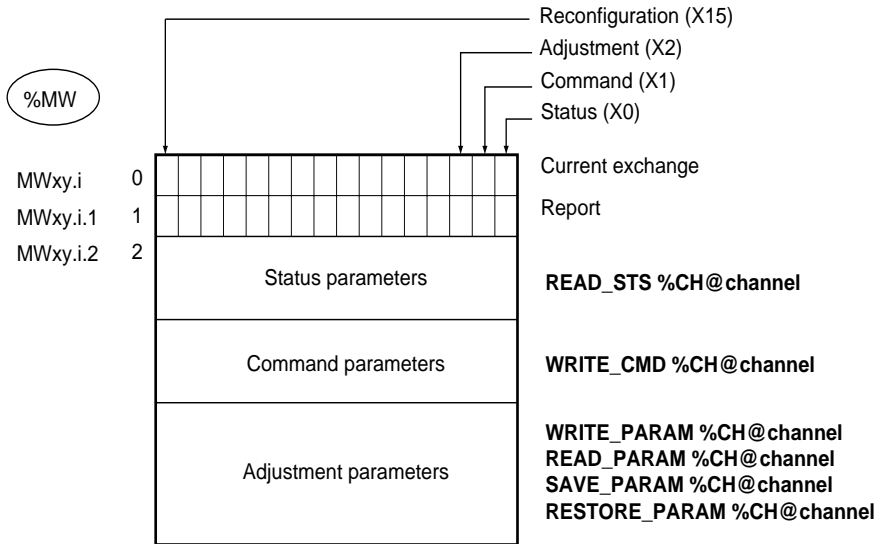
When data is being exchanged between the PLC memory and the module, acceptance by the module can require several task scans. Two words are used to manage the exchanges :

%MWxy.i : Exchange in progress.

%MWx.i.1 : Report.

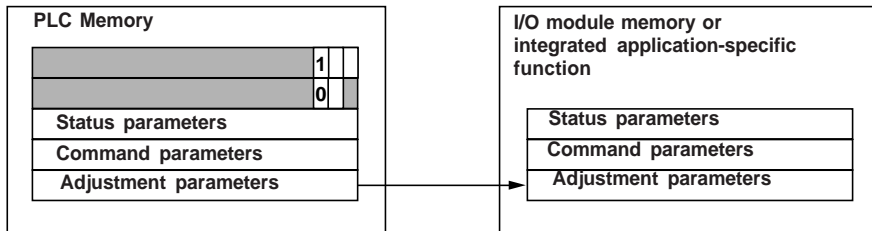
Each of the bits of these words is assigned to a type of parameter :

- Rank 0 bits are associated with the status parameters : bit %MWxy.i.0:X0 indicates if there is an active request to read the status word.
- Rank 1 bits are associated with the command parameters : bit %MWxy.i.0:X1 indicates if the command parameters are sent to the module, bit %MWxy.i.1:X1 states if the command parameters are accepted by channel i of the module.
- Rank 2 bits are associated with the adjustment parameters : bit %MWxy.i.0:X2 indicates if the adjustment parameters are exchanged with channel i of the module (by WRITE\_PARAM, READ-PARAM, SAVE-PARAM, RESTORE-PARAM), bit %MWxy.i.1:X2 states if the adjustment parameters are accepted by the module. If the exchange has been correctly executed, the bit changes to 0.
- Rank 15 bits indicate that channel i of the module has been reconfigured from the terminal (modification of configuration parameters + cold restart of the channel).



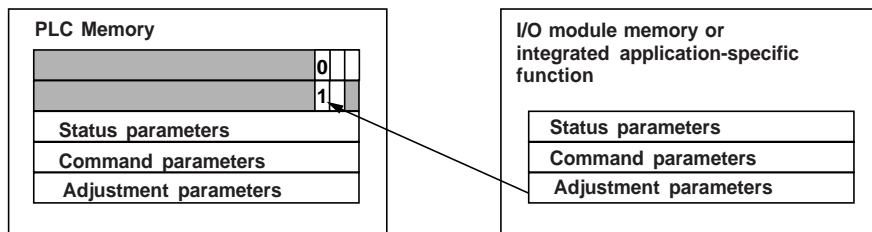
Note : exchange and report words also exist at module level (%MWxy.MOD and %MWxy.MOD.1).

- Transmission of data, example WRITE\_PRM instruction



When the instruction is scanned by the PLC processor, the Exchange in progress bit is set to 1 in %MWxy.

- Analysis of the data by the I/O module and report



When the data is exchanged between the PLC memory and the module, acceptance by the module is managed by bit %MWxy.i.1:X2 : Report (0 = exchange correct, 1 = exchange unsuccessful).

#### Comment :

There are no adjustment parameters at module level.

## 4 Application-specific instructions

### 4.1 General

Application-specific instructions are specific to a function. They are in addition to the basic and advanced instructions, and are described in the individual documentation for each of the application-specific functions.

**Example** : communication function, the READ\_VAR instruction enables the language objects to be read on another PLC connected to the network.

```
READ_VAR(ADR#{2.3}.SYS,'%M',200,32,%M10:2,%M20)
```

Each parameter has a particular significance (eg : the first parameter corresponds to the address of the PLC on which the object(s) will be read).

The parameters are always PL7 language objects : word, word tables, immediate values.

### 4.2 Access to application-specific instructions

Access to the entry of the application-specific function is either :

- by direct entry of the instruction and its parameters in an operation block,
- or via the entry help function which can be accessed in the program editors using the SHIFT F8 keys.

**Function call**

Function Information: **Parameters**

Family	Lib.V.	App.V.	Name	Comment
Movement Command	1.0	-	PID	Mixed PID controller
Orphee functions	1.2	-	PID_MMI	Manage the dedicated man-machine inte>>
Process control	1.8	-	PWM	Width modulation of a digital value pulse
Single length integers	1.0	-	SERVO	PID output processing for controlling a >>

Call Format

Parameters of the PROCEDURE :

Name	Type	Kind	Comment	Entry field
TAG	STRING	IN	PID name (8 char.), for MMI on CCX17	
UNIT	STRING	IN	Measurement unit (6 char.), for MMI on >>	
PV	WORD	IN	Measurement_format [0 ; +10000]	

Display the Call

PID ( )

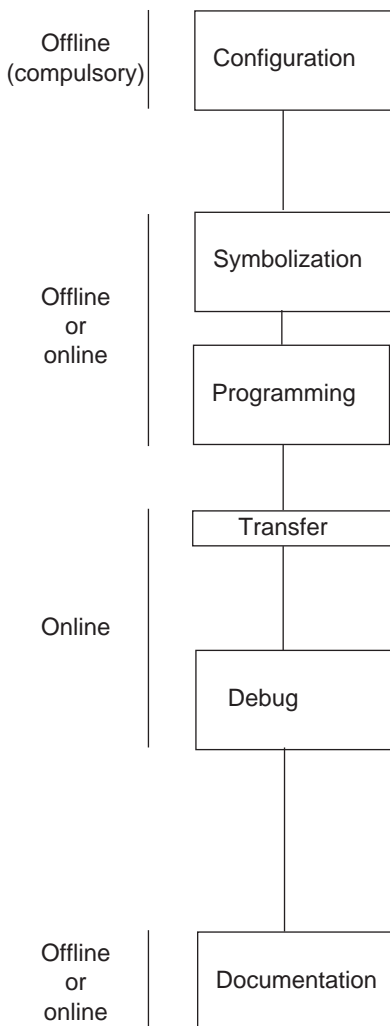
OK Cancel

- Select "**Parameters**" in the pulldown list box.
- Select the application in the **Family** list zone.
- Select the instruction in the adjacent list zone.
- Enter the parameters of the instruction in the "**Entry field**" text box.
- Confirm with **OK**.
- or via the **Tools/Library** command.



## 5.1 Principles

Setting up an application-specific function consists of :



Configuration of the module or the integrated interface :

- Select the rack number (TSX 57).
- Select the position in the rack.
- Enter the configuration parameters.
- Enter the initial adjustment parameters.

Symbolize the variables associated with the application-specific function using the variables editor.

Program the functions which the application-specific function must perform using :

- Bit and word objects associated with the module or the integrated interface.
- Application-specific instructions.

When the program is written offline, it must be transferred into the PLC in order for it to be debugged.

For debugging the application-specific function, the configuration editor offers :

- Debug help screens for controlling the inputs and outputs (forcing, etc).
- Adjustment screens for modifying the values of the adjustment parameters.
- Diagnostics screens for identifying faults.

The documentation editor prints the various pieces of information relating to an application-specific function :

- The configuration parameters.
- The adjustment parameters.
- Etc.

### Note :

The order defined above is for information only. With PL7 software the editors can be used interactively in any order (however, the data or program editors cannot be used before configuring the I/O modules, except for TSX 37 discrete I/O objects).

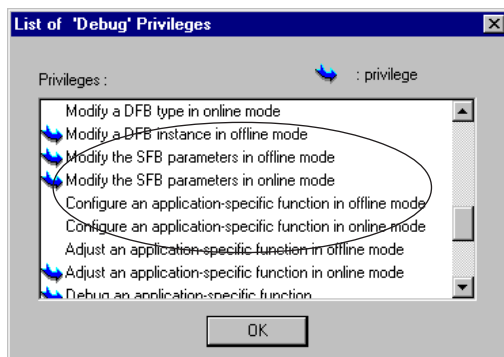
## 5.2 Access security management

The following table details the privileges for application-specific functions according to the different profiles :

Privileges/Profiles	Read only	Operate	Adjust	Debug	Program
Configure an app-spec. function offline	N	N	N	N	Y
Configure an app-spec. function online	N	N	N	N	Y
Debug an app-spec. function	N	N	N	Y	Y
Adjust an app-spec function offline	N	Y	Y	Y	Y
Adjust an app-spec function online	N	Y	Y	Y	Y

**Note :** for the Weighing and Analog functions, adjustment words %MW can be accessed in the Configuration screen in offline mode and the Debug screen in online mode. They are modified according to the Adjust profile and not that of the Configuration or Debug screen.

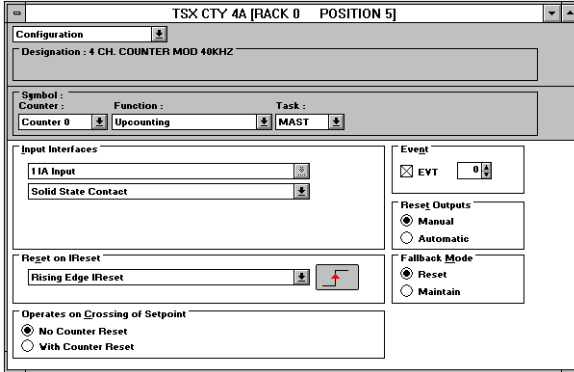
### List of privileges linked to application-specific functions



### 5.3 Event processing

Event processing is described in Reference manual, section 1.6-5, part A. This section summarizes the operations for setting up an application-specific function.

#### Choosing the input to trigger the task :



This choice is made on the channel configuration screen.

- Check the event box.
- Choose the number of the associated event-triggered task (by default, the first free event number is proposed).

Having opened the event-triggered task and created the corresponding program, the list of updated I/O data (at the start (inputs) and at the end (outputs) of the event-triggered task) is created automatically from the I/O objects used in the event-triggered task. This exchange list appears in the program documentation file.

This exchange list is limited depending on the processor and the type of module.

#### TSX 37 :

Maximum number of event-triggered tasks :

- 8 for TSX 37-10 PLCs,
- 16 for TSX 37-21/22 PLCs

It is possible to declare the exchanges of 2 input modules and 2 output modules as a maximum.

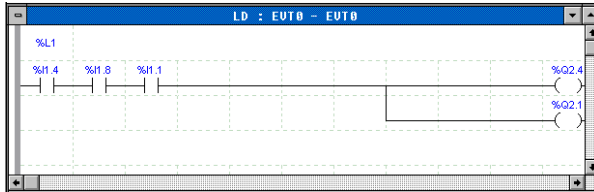
#### TSX 57 :

Number of exchanges which can be used by processors in EVT tasks	P57-10 (32 EVT's)	P57-20/25 (64 EVT's)	P57-30/35 (64 EVT's)	P57-40/45 (64 EVT's)
Max. no. of discrete exchanges	32	128	128	128
Max. no. of ANA exchanges	8	16	16	16
Max. no. of other app-spec exchanges	4	16	16	16

**Reminder :** data on the inputs associated with the channel which triggered the interrupt is updated systematically.

Examples :

- Using discrete module objects in event 0



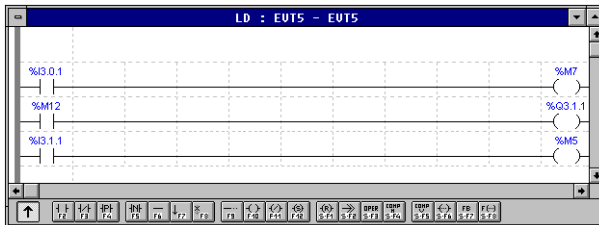
The list of I/O data is as follows (16 inputs and 8 outputs)

Group of 8	%I1.1 and I1.4	%I1.0	%I1.8	%I1.8	%Q2.1 and %Q2.4	%Q2.0
		%I1.1		%I1.9		%Q2.1
		%I1.2		%I1.10		%Q2.2
		%I1.3	Group of 8	%I1.11	Group of 8	%Q2.3
		%I1.4		%I1.12		%Q2.4
		%I1.5		%I1.13		%Q2.5
		%I1.6		%I1.14		%Q2.6
		%I1.7		%I1.15		%Q2.7

**Comment :**

For discrete I/O, there is only one exchange per group of 8 channels.

- Using input objects from channel 0 of a TSX CTY 4A module in position 3 and I/O objects from channel 1 of the same module in the event-triggered task; this task is activated by an event generated by channel 3 of a TSX CTY 4A module in position 4.



The exchange list established in the documentation is as follows :

Inputs                      Outputs



---

The following objects are updated (module in position 3) :

<b>Channel 0 :</b>	inputs	%I3.0	%IW3.0.2	%ID3.0
		%I3.0.1	%IW3.0.3	%ID3.0.4
		%I3.0.2		
		...		
		%I3.0.15		
<b>Channel 1 :</b>	inputs	%I3.1	%IW3.1.2	%ID3.1
		%I3.1.1	%IW3.1.3	%ID3.1.4
		%I3.1.2		
		...		
		%I3.1.15		
	outputs	%Q3.1	%QW3.1	
		%Q3.1.1	%QW3.1.1	
		%Q3.1.2		
		...		
		%Q3.1.15		

**Comment :**

Channel 3 input objects of the module located in position 4 are updated implicitly. Moreover, the exchange relating to this channel is not included when counting the maximum number of exchanges.

**Comments :**

- If a modification is made when programming an event or the configuration, the exchange list can be modified.  
If the maximum number of exchanges is reached, PL7 indicates this with the message "exchange list too large".
- The exchange list cannot be modified in online mode. Therefore it is not possible to make any modification to the EVT program which adds or removes an exchange from the EVT exchange list.  
It is, however, possible to remove or add an I/O channel as long as the exchange list is not modified.
- Objects %I and %Q for remote modules on a FIPIO link can be accessed in event processing, but are not updated in realtime during this processing.



In the case of discrete I/O (for the whole range) or analog I/O (TSX 37), the use of an I/O object in the event-triggered task updates all the I/O objects of a single group of channels.

In the case of remote I/O, only the I/O images are updated in realtime.

During a cold start or a warm restart, current and waiting events are lost. When the PLC changes from RUN --> STOP, waiting events are lost.

**Reminder** : data on the inputs associated with the channel which triggered the interrupt is updated automatically.

The event-triggered task is programmed in Instruction list language, structured text language or Ladder language. The program must remain short so that it does not disrupt the main task or contain explicit exchange instructions.

#### **Programming the main task :**

From a main task (master or fast), it is possible to manage the events :

- Using the MASKEVT and UNMASKEVT instructions (see Reference manual, section 1.5-6 part B). The calls are memorized during masking.
- Using bit %S38 for enabling/inhibiting event-triggered processing. The calls are lost during inhibition.

---

## 5.4 Operating modes

---

Processing the operating modes is described in Reference manual, section 1.4, part A.

- **cold start** : the initial adjustment parameters replace the current parameters (except when setting %S0 to 1 via the program or the terminal, in which case the parameters of the I/O modules retain their current values).
  - **warm restart** : the parameters of the I/O modules retain their current values.
  - **RUN/STOP** : when the PLC stops,  
The outputs are set to fallback position (1), the inputs are updated.  
The Grafcet chart is frozen in the current situation.  
The events are lost.
  - **%S8** : the system bit is only managed with the TSX 37. It is used to test the output wiring :
    - %S8 = 1 : the outputs are forced to 0,
    - %S8 = 0 : the outputs can be modified via the terminal.
  - **stop on task (breakpoint)** : the outputs associated with the task change to fallback mode (TSX 57),
  - **%S9** : in normal operation this system bit causes the outputs to change to fallback mode (1).
  - **%SW8** : controls acquisition of inputs for the tasks :  
Used to inhibit the input acquisition phase for each task :  
%SW8:X0 = 1 inhibited in the master task  
%SW8:X1 = 1 inhibited in the fast task
  - **%SW9** : controls updating outputs for the tasks :  
Used to inhibit the update phase for the outputs in each task :  
%SW9:X0 = 1 inhibited in the master task  
%SW9:X1 = 1 inhibited in the fast task
- (1) Output fallback mode :
- retains the current value,
  - fallback to a given value depending on the configuration selected (default value = 0) for the TSX 57,
  - fallback to 0 for the TSX 37.

---

## 5.5 Processing application-specific faults by program

---

It is possible to process faults by program : the following objects can be used :

- **%S10** :
  - TSX 37 : this system bit indicates if an I/O fault is detected in the main rack or in the extension rack.
  - TSX 57 : this system bit indicates if an I/O fault is detected in one of the racks or on the FIPIO bus.

Reading system bit %S4i (rack number i) indicates a fault in rack i.  
 Each bit in the group of words **%SW128 to %SW143** indicates the status of the connection point. The presence of a bit at 0 indicates a connection point fault (with %SW128:X0=address 0 ... %SW143:X15=address 255).
- **%S16**: this system bit indicates if an I/O fault is detected in the current task (on TSX 57).
- **%S118** : this system bit indicates a general FIPIO I/O fault (on TSX 57).
- **%S119** : this system bit indicates a general I/O fault (on TSX 57).
- **%lxy.MOD.ERR** : this bit indicates if an I/O fault is detected on the module located in position y of rack x (1).
- **%lxy.i.ERR** : this bit indicates a channel fault in channel i of the module located in position y of rack x (1).
- **%SW16** : this system word indicates a FIPIO dialog fault between the processor and a remote I/O module in the current task (on TSX 57).
- **%SW128 to %SW143** : each bit in this group of words indicates the status of the connection point. The presence of a bit at 0 indicates a connection point fault (on TSX 57).

To obtain an additional level of detail, it is possible to read the status words explicitly via the program :

Example :

- **%MWxy.MOD.2** : module status words
- **%MWxy.i.2** : channel status word  
and so on
- **%S39** :  
Saturation of event processing. This bit is set to 1 by the system to indicate to the application that one or several events have been lost as a result of a stack overflow.

(1) For the TSX 37, only one rack is possible. The rack address (0) does not appear, only the position (y) in the rack appears.

## 6.1 Status words

The status word bits contain information on the operating status of the module (or the channel):

Fault at module level %MWxy.MOD.2 : Xj

Bit	Function	Meaning
0	Module faulty	Internal fault, module off
1	Operational fault	Fault : process, faulty channel or application program (one of the command, adjustment or configuration values is not accepted by the application-specific function).
2	Terminal block fault	The terminal block is not connected to the module.
3	Self-tests running	
4	Reserved	
5	Hardware or software configuration fault	The module present is not the one declared at configuration.
6	Module absent	
7	Reserved	

Fault at channel level %MWxy.i.2 : Xj

Bit	Function	Meaning
0	External fault	(1)
1	External fault	(1)
2	Terminal block fault	The terminal block is not connected to the module.
3	External fault	(1)
4	Internal fault	Module off, missing or self-test is running.
5	Configuration fault	The module present is not the one declared at configuration.
6	Communication fault	CPU-module exchange fault (TSX 57)
7	Application fault	Fault with the configuration parameters of the application-specific function

(1) The interpretation of each of the 3 bits depends on the application-specific function (example : bit 0 corresponds to range overshoot in the case of analog modules)

### Notes :

- The other bits of number 2 channel level status word or other channel level status words are specific to each application-specific function.
- In the case of remote FIPIO I/O, with a base module followed by an extension module, it is only the status word of the base module which is significant and which can be accessed. Its low order byte is assigned to the base module, and its high order byte is assigned to the extension module.

## 6.2 Printing the parameters of I/O modules

The documentation editor can be used to print the symbols and parameters associated with the different I/O for each module or integrated interface.

In the Application Browser, double-click on the following icons :  
Station, Documentation File, Station folder, Configuration, Hardware configuration,  
Module configuration then Module parameters. Select the **File/Print** command.

Example : TSX AEY 1600 module

TSX AEZ 1600 [RACK 0 POSITION 4]						
<b>Identification of the module</b>						
<b>Product ref. :</b>	TSX AEY 1600	<b>Designation :</b>	16ANA.INP.HIGH LEVEL			
<b>Position :</b>	4	<b>Symbol :</b>				
<b>Common parameters</b>						
<b>Type :</b>	Inputs	<b>Terminal block detection :</b>	active			
<b>Channel parameters</b>						
ChannelSymbol	Range	Scale	Min	Max	Filter	Task
0	0..10V	User	0	100	2	MAST
1	0..20mA	%..	0	10000	6	MAST
2	4..20mA	%..	0	10000	2	MAST
3	+/-10V	%..	-10000	100000	0	MAST
4	+/-10V	%..	-10000	100000	0	MAST
5	+/-10V	%..	-10000	100000	0	MAST
6	+/-10V	%..	-10000	100000	0	MAST
7	+/-10V	%..	-10000	100000	0	MAST
8	+/-10V	%..	-10000	100000	0	MAST
9	+/-10V	%..	-10000	100000	0	MAST
10	+/-10V	%..	-10000	100000	0	MAST
11	+/-10V	%..	-10000	100000	0	MAST
12	+/-10V	%..	-10000	100000	0	MAST
13	+/-10V	%..	-10000	100000	0	MAST
14	+/-10V	%..	-10000	100000	0	MAST
15	+/-10V	%..	-10000	100000	0	MAST

**A**

Adjustment	2/5
Analog inputs	3/5
Analog output	3/5
Application-specific faults	5/8
Application-specific instructions	4/1
Application-specific screens	2/3

**C**

Communication interface	3/5
Configuration	2/4

**D**

Debugging	2/6
Declaration of an I/O module	2/1
Diagnostics	2/6
Documentation editor	6/2

**E**

Events	5/3
Explicit exchange objects	3/1
Explicit exchanges	3/13
Explicit restore of the initial adjustment parameters	3/18
Explicit save of adjustment parameters	3/18

**F**

Faults	2/6
--------	-----

**I**

Implicit exchange objects	3/1
Implicit exchanges	3/12
Integrated application-specific interfaces	3/5

**L**

Library	4/1
---------	-----

**M**

Module type	2/2
-------------	-----

**O**

Object addressing	3/2
Operating modes	5/7

**P**

Printing	6/2
Processing application-specific faults	5/8

**R**

READ_PARAM	3/17
READ_STS	3/16
Reading status words	3/16
Restore Parameters	2/5
RESTORE_PARAM	3/18

**S**

Save Parameters	2/5
SAVE_PARAM	3/18
Setup	5/1
Status	6/1
Status words	6/1

**T**

Terminal port	3/5
---------------	-----

**U**

Up/Down counter channels	3/5
--------------------------	-----

**W**

WRITE_CMD	3/16
WRITE_PARAM	3/17
Writing command words	3/16





<b>Section</b>	<b>Page</b>
<b>1 Configuring the discrete function</b>	<b>1/1</b>
1.1 Introduction	1/1
1.2 Reminder of the configuration editor	1/2
1.2-1 Accessing the configuration editor	1/2
1.3 Configuring in-rack discrete modules	1/3
1.3-1 Choosing the modules	1/3
1.3-2 Accessing the channel parameter settings of a discrete module	1/4
1.4 Configuring distributed discrete modules	1/6
1.4-1 Accessing the FIPIO configuration screen	1/6
1.4-2 Selecting a FIPIO connection point	1/6
1.4-3 Selecting the device to be connected	1/7
1.4-4 Accessing the channel parameters	1/8
1.4-5 Using the OTHER_FRD reference	1/9
<b>2 Setting the channel parameters on a discrete I/O module</b>	<b>2/1</b>
2.1 Presentation	2/1
2.2 Displaying channel parameters	2/2
2.3 Modifying channel parameters	2/3
2.4 Input modules	2/4
2.4-1 In-rack discrete input parameters	2/4
2.4-2 TBX distributed discrete input parameters	2/5
2.4-3 Momentum distributed discrete input parameters	2/6
2.5 Modifying input parameters	2/7
2.5-1 Types of task	2/7
2.5-2 Functions	2/8
2.5-3 Filtering	2/9

Section	Page
2.5-4 External power supply fault monitoring	2/9
2.5-5 Latching	2/10
2.5-6 Wiring check	2/10
<b>2.6 Output modules</b>	<b>2/11</b>
2.6-1 In-rack discrete output parameters	2/11
2.6-2 TBX distributed discrete output parameters	2/12
2.6-3 Momentum distributed discrete output parameters	2/13
<b>2.7 Modifying output parameters</b>	<b>2/14</b>
2.7-1 Types of task	2/14
2.7-2 Fallback mode	2/15
2.7-3 Reactivating the outputs	2/16
2.7-4 External power supply fault monitoring	2/17
2.7-5 Wiring check	2/17
<b>2.8 Deconfiguring and reconfiguring groups of channels</b>	<b>2/18</b>
<b>2.9 Setting the parameters of the RUN/STOP input</b>	<b>2/19</b>
<b>2.10 Confirming the configuration</b>	<b>2/20</b>
2.10-1 Confirming after modification	2/20
2.10-2 Global reconfiguration	2/20
<b>3 Debugging discrete modules</b>	<b>3/1</b>
<b>3.1 Introduction to the Debug function</b>	<b>3/1</b>
<b>3.2 Description of the debug screen</b>	<b>3/2</b>
<b>3.3 Displaying the module diagnostics</b>	<b>3/4</b>
<b>3.4 Canceling forcing on the channels of a module</b>	<b>3/5</b>
<b>3.5 Accessing channel commands</b>	<b>3/5</b>
3.5-1 Displaying detailed channel diagnostics	3/6

<b>Section</b>	<b>Page</b>
3.5-2 Forcing or unforcing channels	3/7
3.5-3 Masking or unmasking events	3/8
3.5-4 Reactivation command (discrete outputs)	3/9
3.5-5 Applied outputs (discrete outputs)	3/9
3.5-6 Write command (discrete outputs)	3/9
<b>4 Bits and words associated with the discrete function</b>	<b>4/1</b>
4.1 In-rack discrete I/O module object addressing	4/1
4.2 Addressing distributed discrete I/O modules	4/2
4.3 Language objects associated with discrete I/O	4/3
4.3-1 Implicit exchange objects	4/3
4.3-2 Explicit exchange objects	4/4
4.4 Reading the status word and writing the command word	4/6
4.4-1 Reading the status word	4/6
4.4-2 Writing the channel command word	4/6
<b>5 Setting up the reflex discrete I/O module</b>	<b>5/1</b>
5.1 Presentation	5/1
5.1-1 Area of application of the module	5/1
5.1-2 Module functions	5/1
5.1.3 List of reflex function blocks	5/2
5.2 Configuration and parameter entry	5/3
5.2-1 Selecting the module	5/3
5.2-2 Accessing the module parameters	5/3
5.2-3 Entering the input parameters	5/4
5.2-4 Entering the output parameters	5/5
5.3 Configuring the reflex functions	5/7
5.3-1 Language objects associated with the reflex module	5/7
5.3-2 The reflex function Ladder editor	5/10

---

Section	Page
5.4 Function blocks	5/14
5.4-1 Direct	5/14
5.4-2 Combinational function	5/14
5.4-3 On-delay timer function block	5/15
5.4-4 Off-delay TIMER function block	5/15
5.4-5 On-delay / off-delay TIMER function block	5/16
5.4-6 2-value on-delay TIMER function block	5/16
5.4-7 Selectable on-delay / off-delay TIMER function block	5/18
5.4-8 Retriggerable Monostable function block	5/19
5.4-9 Time-delayed monostable function block	5/20
5.4-10 2-value monostable function block	5/21
5.4-11 OSCILLATOR function block	5/22
5.4-12 COUNTER OUTPUT D function block	5/23
5.4-13 COUNTER OUTPUT T function block	5/24
5.4-14 Double threshold COUNTER function block	5/25
5.4-15 Single electronic cam function block	5/26
5.4-16 INTERVAL COUNTER function block	5/27
5.4-17 BURST function block	5/28
5.4-18 PWM function block	5/29
5.4-19 Slow speed detection function block	5/30
5.4-20 Speed monitoring function block	5/31
5.4-21 Type 1 Command/Control function block	5/33
5.4-22 Type 2 Command/Control function block	5/34
5.4-23 Control/Counting function block	5/35
5.4-24 Fault indication function block	5/36
5.5 Configuring the function blocks	5/37
5.5-1 Adjustment via the terminal in offline mode	5/37
5.5-2 Modifying the parameters via the program	5/38
5.5-3 Explicit exchange language objects	5/38
5.6 Debugging	5/39
5.6-1 Accessing debugging	5/39
5.7 Diagnostics	5/41
<b>6 Index</b>	<b>6/1</b>

---

## 1.1 Introduction

---

This part concerns :

- discrete rack-mounted I/O modules
- distributed discrete I/O modules connected on the FIPIO bus.

To access the latter, the configured processor must be a processor with an integrated FIPIO link.

To create an application program, the physical operating context in which it will be executed should be defined : i.e. the rack and the modules located in it, the power supply, the processor and the discrete and application-specific modules, the connection points and the devices connected to any FIPIO link. The channels of each module must also be configured.

Each discrete I/O module has default parameters which can be modified individually or in groups of channels according to type. To do this, PL7 software is equipped with a **configuration** editor which fulfills this requirement either :

- in offline mode or,
- in online mode; in this case only certain parameters can be modified.

In online PLC operation, it also offers a debugging function which can be used to :

- display the status of the channels,
- modify the status of the channels (forcing, set to 1, set to 0),
- access module and channel diagnostics.

**Note** : the functions in online mode cannot be accessed for distributed I/O modules.

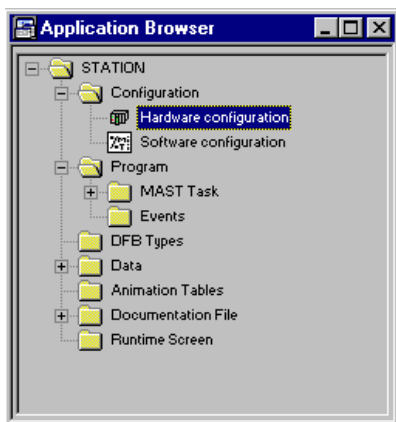
---

## 1.2 Reminder of the configuration editor

---

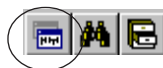
### 1.2-1 Accessing the configuration editor

Use the Application Browser to select the Station folder and then the Configuration folder, then double-click on the "Hardware configuration" icon.



If the Application Browser is not displayed :

- click on the Application Browser icon

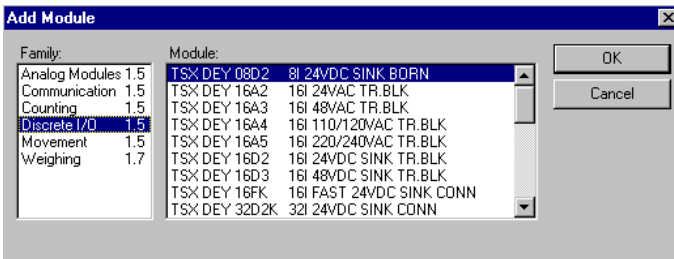


- or select **Tools/Application Browser**

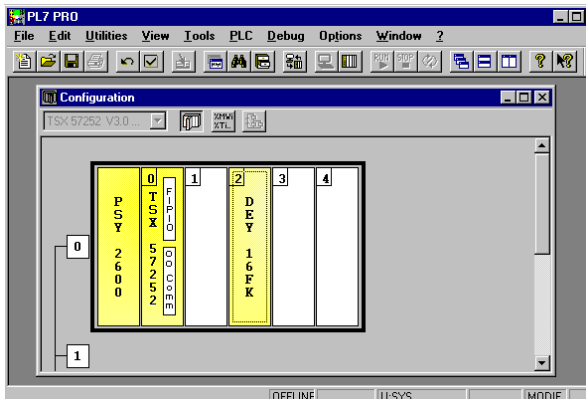
## 1.3 Configuring in-rack discrete modules

### 1.3-1 Choosing the modules

To add a module to the PLC configuration, **double-click** on the position in the rack where it is to be installed (for example, position 2); which displays the following selection list :

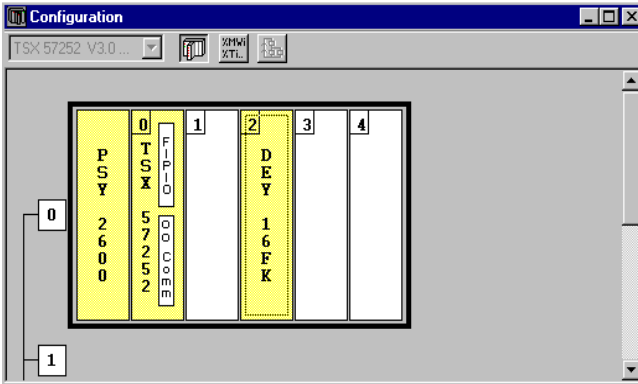


In the **Family** field, select the "discrete" family, then, in the **Module** field, the module reference, (for example TSX DEY 16FK). Once the choice has been confirmed with **OK**, the module is displayed in the configuration ; the module reference appears on a shaded background in position 2.



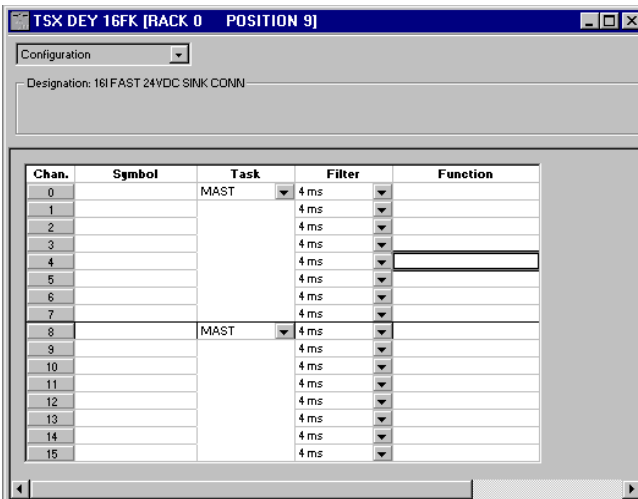
### 1.3-2 Accessing the channel parameter settings of a discrete module

To set the channel parameters of a module, double-click on the position of the selected module in the rack, or select the **Utilities/Open Module** command.



Procedure :

- 1 Click on the position of the module for which parameters are to be set (2 in the example). The following parameter screen will then appear :

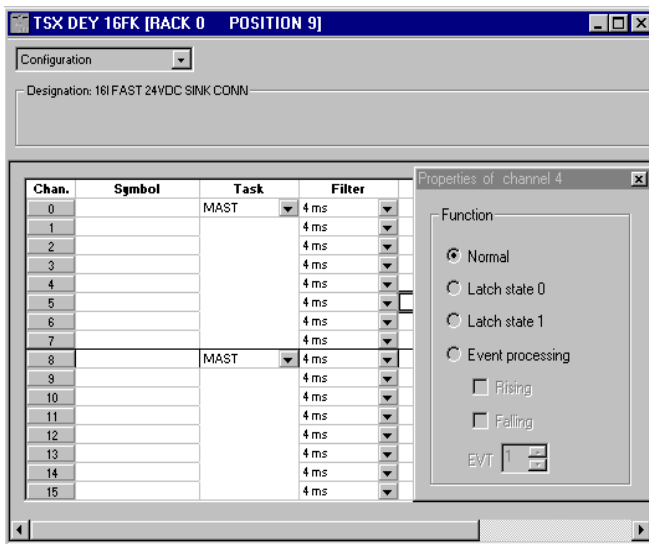




- 2 Set the parameters for each of the channels. For some modules, a dialog box appears from which additional parameters can be selected.

This dialog box can be accessed by :

- clicking with the right mouse button on the table cell which corresponds to the channel for which parameters are to be set and by selecting Properties from the pull-down list or,
- double-clicking with the left mouse button on the table cell which corresponds to the channel for which parameters are to be set or,
- selecting the Function cell for the channel for which parameters are to be set and pressing ENTER.



- 3 Confirm the configuration by closing the parameter settings window.

To confirm, click on the icon :



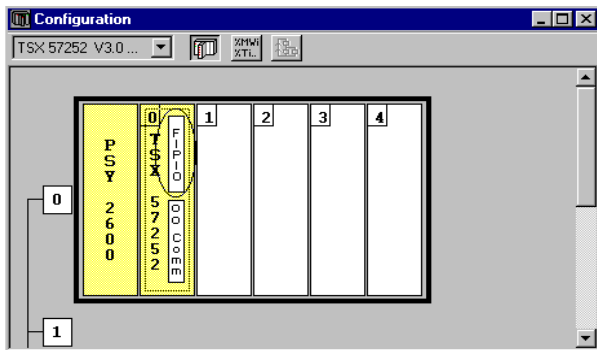
or select the **Edit / Confirm** command  
Ctrl+W

or select **Confirm** from the shortcut menu.

## 1.4 Configuring distributed discrete modules

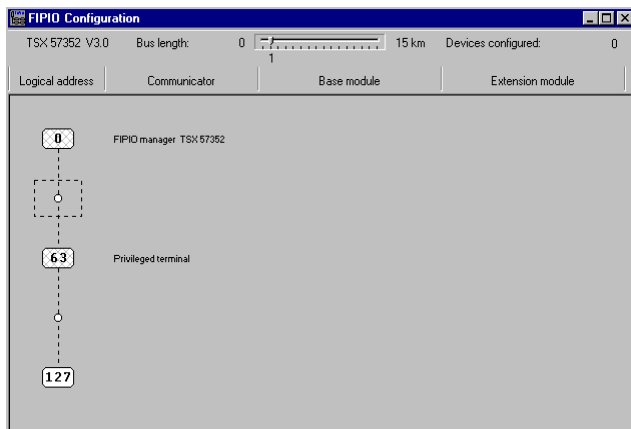
### 1.4-1 Accessing the FIPIO configuration screen

To access the FIPIO configuration screen, **double-click** on the FIPIO zone of the processor (this must be a processor with an integrated FIPIO link) :



### 1.4-2 Selecting a FIPIO connection point

The screen shows the FIPIO bus, giving the occupied addresses in the Logical address column. Initially, if no devices are configured, only the first and last connection points and connection point 63 are displayed (addresses 0 and 63 are reserved by the system).



### 1.4-3 Selecting the device to be connected

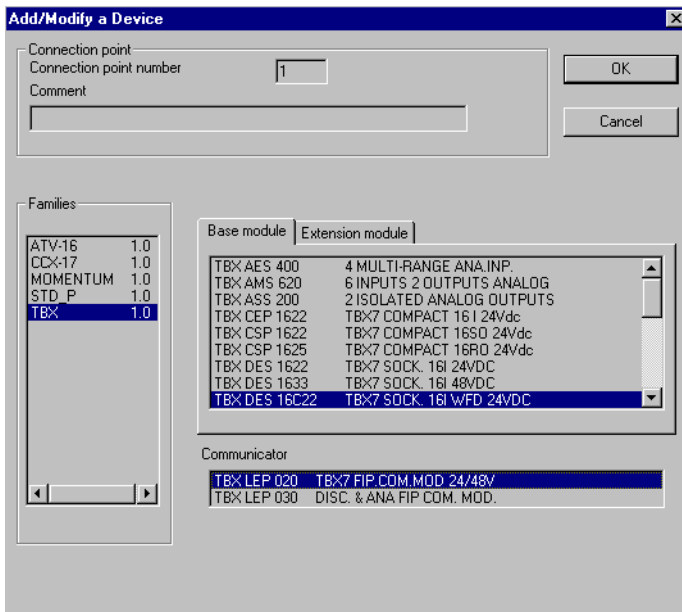
**Double-click** in the "Logical address" column at the position where the device should be connected.

Enter the address of the connection point (1), enter a comment (optional).

To select a device, first select the family (TBX or MOMENTUM), the base module and the extension module, and then the communicator.

For MOMENTUM, if the base unit reference used is not yet offered, select OTHER FRD (see section 1.4-5).

**Note** : the communicator and the extension module are not supported for compact TBXs. The extension module is not supported for Momentum modules.



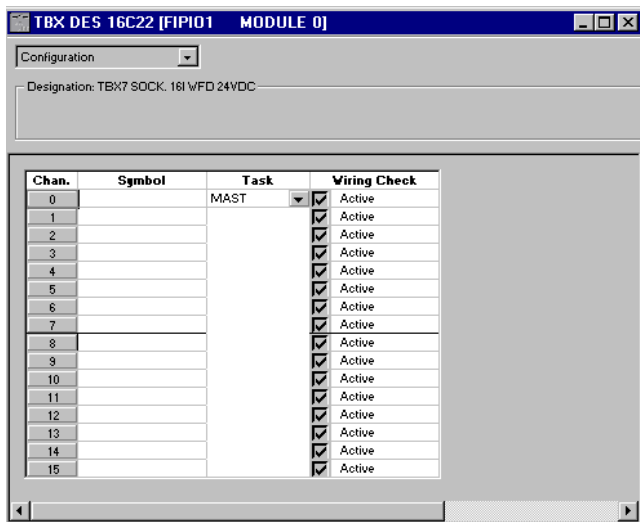
(1) The maximum number of connection points per device is summarized in the table below :

Type	compact TBX	16-channel TBX	32-channel TBX	Momentum
No.	31	78	88	99

For more information, please refer to the communication manual : FIPIO part.

## 1.4-4 Accessing the channel parameters

To set the channel parameters of a module, double-click on the position of the selected module on the bus, or select the **Edit/Open Module** command.



Procedure :

- 1 Click on the position of the module for which parameters are to be set (2 in the example). The parameter screen will then appear.
- 2 Set the parameters for each of the channels.
- 3 Confirm the configuration screen by closing the parameter window.

To confirm, click on the icon :

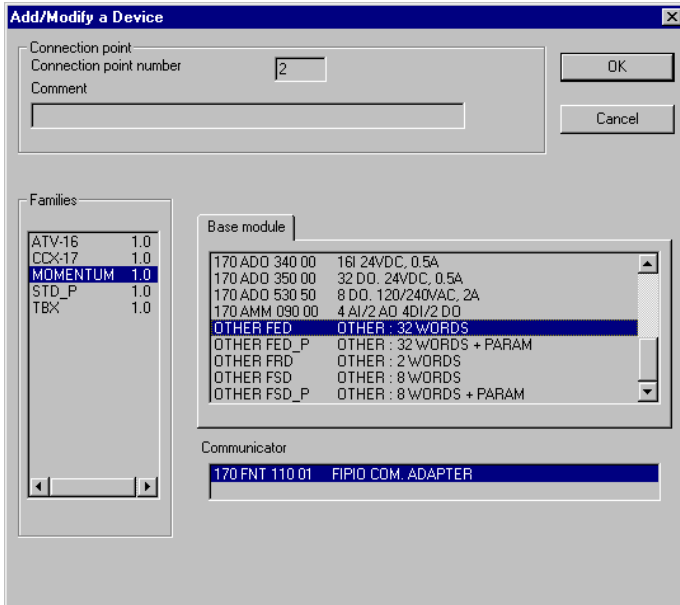


or select the **Edit / Confirm** command  
Ctrl+W

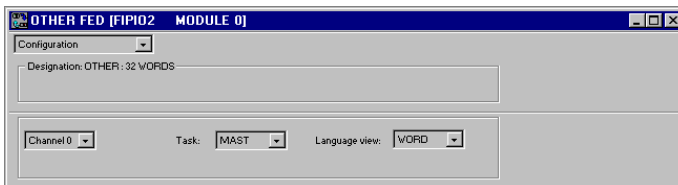
or select **Confirm** from the shortcut menu (accessed by clicking with the right mouse button on the screen background).

### 1.4-5 Using the OTHER\_FRD reference

If the base unit reference used is not yet offered by the configuration tool, select OTHER\_FRD.



The task associated with the module can be selected from the screen which appears.





## 2 Setting the channel parameters on a discrete I/O module

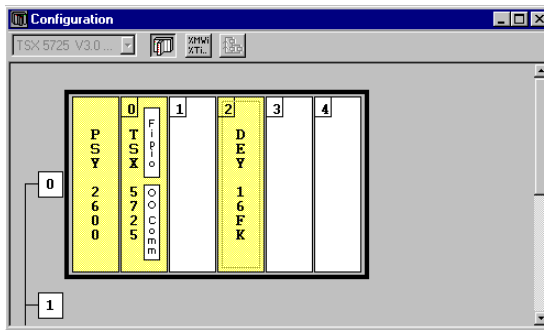
## 2.1 Presentation

The I/O channels on a module have configuration parameters which can be displayed and modified in the **configuration** editor. These parameters offer the following functions :

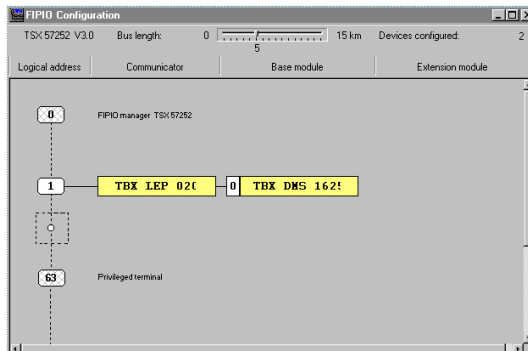
- assignment of channels to a task,
- filter times for fast inputs,
- fallback mode for outputs on a fault,
- etc.

**For in-rack I/O modules :**

Access the channel parameters screen for a module by double-clicking on the rack module to be configured.

**For distributed I/O modules :**

Access the channel parameters screen for a module by double-clicking on the FIPIO bus module to be configured.

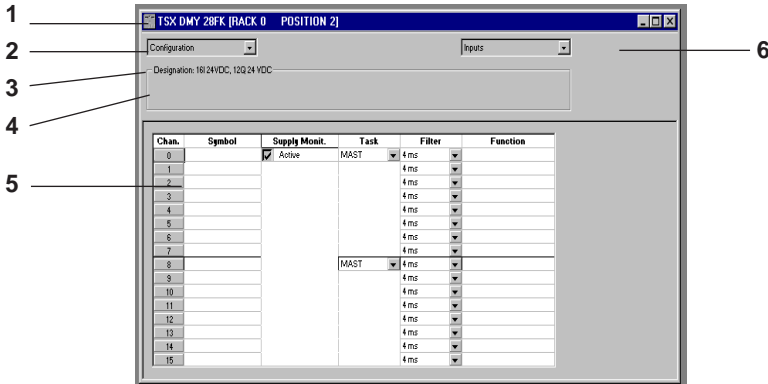


If the parameters are not modified, the channels are configured according to the **default** settings.

## 2.2 Displaying channel parameters

The configuration screen for the module selected in the rack or on the FIPIO bus displays the parameters associated with the input or output channels.

This screen accesses the parameter **display** and **modification** functions in offline mode and the **Debug** function in online mode.



### Description

The **title bar** (1) indicates the reference of the selected module and its physical position, as well as the rack number for in-rack modules or the FIPIO connection point for distributed I/O.

Current operating mode: **Configuration** (2); in online mode the window also gives access to the **Debug** function (diagnostics).

**Designation** (3) of the selected module :

Example :

- number and type of channels :
  - 4 INP : designates the number of inputs for the selected module; in the example given, the module has 64 inputs,
- characteristics of the module power supply :
  - 24 VDC Sink : 24VDC positive logic,
- type of connection : CONN : HE10 type connector.

The display of the module zone (4) is optional. Access is via the **View/Module Zone** command.

**Channel selection boxes** (5) :

All the **channels** and associated **symbols**. The name (symbol) is defined by the user in the variables editor. The scroll bar to the right is used to display all the module channels at the top and bottom of the list.

The pull-down list box (6) is offered for modules with both inputs and outputs. It is used to select either the inputs or the outputs, or the program for programmable TBX modules (TBX DMS 16P22).



## 2.3 Modifying channel parameters

The configuration editor has a number of functions for facilitating the entry and modification of module parameters.

### Shortcut menus

These can be accessed by clicking with the right mouse button, providing fast access to the main commands.

in a table cell :

Copy Parameters
Paste Parameters
Properties

in the module zone (outside a table) :

Undo
Confirm
Animate

### Selecting a channel

A channel can be selected in its entirety by clicking on the number of the required channel in the channel column.

### Multiple channel selection

- To select a group of consecutive channels :  
Select the 1st channel,  
Hold down SHIFT and click on the last channel.
- To select a group of channels which are not consecutive :  
Select the 1st channel,  
Hold down CTRL and click on each of the channels individually.

### Selecting a channel parameter

Click on the associated cell.

### Selecting several consecutive cells

Click on the 1st cell and, holding the mouse button down, drag the mouse upwards or downwards, releasing the button when the last cell is reached.

### Copy/paste

Copy : select the cell or channel to be copied, then select "Copy Parameters" from the shortcut menu.

Paste : select the cell(s) or channel(s) to be pasted, then select "Paste Parameters" from the shortcut menu.

## 2.4 Input modules

### 2.4-1 In-rack discrete input parameters

Discrete input modules have parameters for each channel, for each group of 8 consecutive channels or for each group of 16 consecutive channels.

Module references	Associated task (group of 8 channels)	Functions (per channel)	Filtering (per channel)	Supply monitoring (group of 16 channs)
<b>TSX DEY 08D2</b> 8 inputs	<b>Mast / Fast</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16A2</b> 16 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16A3</b> 16 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16A4</b> 16 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16A5</b> 16 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16D2</b> 16 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16D3</b> 16 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 32D2K</b> 32 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 32D3K</b> 32 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 64D2K</b> 64 inputs	<b>Mast / Fast / None</b>	-	-	<b>Active/Inactive</b>
<b>TSX DEY 16FK</b> 16 inputs	<b>Mast / Fast / None</b>	<b>Normal</b> or 0,1 or RE,FE,RE and FE	0.1 to 7.5 ms <b>4ms</b>	<b>Active/Inactive</b>
<b>TSX DMY 28FK</b> inputs	<b>Mast / Fast / None</b>	<b>Normal</b> or 0,1 or RE,FE,RE and FE	0.1 to 7.5 ms <b>4ms</b>	<b>Active/Inactive</b>
<b>TSX DMZ 16DTK</b> inputs	<b>Mast / Fast / None</b>	-	0.1 to 7.5 ms <b>4ms</b>	<b>Active/Inactive</b>
<b>TSX PAY 262</b> <b>TSX PAY 282</b> inputs	<b>Mast / Fast / None</b>	-		
<b>TSX DMY 28RFX</b> inputs	<b>Mast / Fast / None</b>		0.1 to 7.5 ms <b>4ms</b>	<b>Active/Inactive</b>

#### Notes :

- The parameters in bold correspond to the default parameters.
- The first group of module channels (addresses 0 to 7) is always assigned to a Mast or Fast task. The following groups also have the option : None (no task assigned to a group of unused channels).

### 2.4-2 TBX distributed discrete input parameters

TBX distributed discrete input modules have parameters for each channel and for all channels.

The parameters can be configured according to the options proposed in the following table :

TBX module references	Associated task (for the module)	Filtering (per channel)	Latching	Wiring check (per channel)
<b>TBX CEP 1622</b> 16 inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DES 1622</b> 16 inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DES 1633</b> 16 inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX EEP 1622</b> 16 inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DMS 1025</b> module inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DMS 1625</b> module inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DES 16S04</b> 16 inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DMS 16S44</b> module inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DMS 16P22</b> (1) module inputs	<b>Mast / Fast</b>	-	-	-
<b>TBX DES 16C22</b> 16 inputs	<b>Mast / Fast</b>	-	-	<b>Active / inactive</b>
<b>TBX EEP 08C22</b> 16 inputs	<b>Mast / Fast</b>	-	-	<b>Active / inactive</b>
<b>TBX DMS 16C22</b> module inputs	<b>Mast / Fast</b>	-	-	<b>Active / inactive</b>
<b>TBX DMS 16C222</b> module inputs	<b>Mast / Fast</b>	-	-	<b>Active / inactive</b>
<b>TBX DES 16F22</b> 16 inputs	<b>Mast / Fast</b>	<b>Normal / fast</b>	Active / <b>inactive</b>	-

**Note :** The parameters in bold correspond to the default parameters.

(1) For TBX DMS16P22 modules, the type of channel (input or output) for each of the channels 8 to 15 is selected in the "Type" column which appears on selecting "Programmable" from the pulldown list menu6 (see section 2.2).

---

### 2.4-3 Momentum distributed discrete input parameters

With Momentum distributed discrete input modules, the task can only be selected for all the module channels.

The parameters can be configured according to the options proposed in the following table :

<b>TIO module references</b>	<b>Number of channels</b>	<b>Associated task (for the module)</b>
<b>170 ADI 340 00</b>	16 inputs	<b>Mast</b> / Fast
<b>170 ADI 350 00</b>	32 inputs	<b>Mast</b> / Fast
<b>170 ADM 350 10</b>	16 inputs, 16 outputs	<b>Mast</b> / Fast
<b>170 ADM 390 30</b>	10 inputs, 8 outputs	<b>Mast</b> / Fast
<b>170 ADM 370 10</b>	10 inputs, 8 outputs	<b>Mast</b> / Fast

**Note** : The parameters in bold correspond to the default parameters.

## 2.5 Modifying input parameters

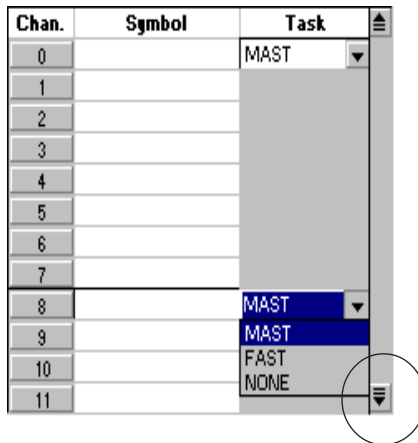
### 2.5-1 Types of task

This parameter defines the task in which the inputs will be read.

The "Task" parameters can be accessed via pulldown menus in the "Task" column.

To access other groups (such as in-rack discrete modules with more than 16 channels), click on the scroll bar pointing to the bottom of the list of channels, as in the example below :

Chan.	Symbol	Task
0		MAST
1		
2		
3		
4		
5		
6		
7		
8		MAST
9		MAST
10		FAST
11		NONE



#### In-rack discrete input modules

Each pulldown list box defines, for 8 consecutive channels, the task at the beginning of which the inputs will be read :

- **Fast** task
- or **Mast** task
- or **None** if the group of channels is not used.

#### Distributed discrete input modules

The pulldown list defines, for all the module channels, the task at the beginning of which the inputs will be read :

- **Fast** task
- or **Mast** task

**Note** : Tasks can only be modified in offline mode.

## 2.5-2 Functions

The fast inputs of **TSX DEY 16 FK** and **TSX DMY 28 FK** modules (inputs) are event-triggered.

The channels of these modules have parameters operating in exclusive mode :

- normal (no event associated with the channel),
- channel by channel latching (at 0 or 1),
- channel by channel event processing :
  - event triggered on rising edge (RE),
  - event triggered on falling edge (FE),
  - event triggered on rising and falling edge.

Event inputs are associated with a processing **number (Evti)**. These numbers range from :

- **0 to 31** for a TSX/PMX/PCX 5710 processor,
- **0 to 63** for a TSX/PMX/PCX 572•/3•/4• processor.

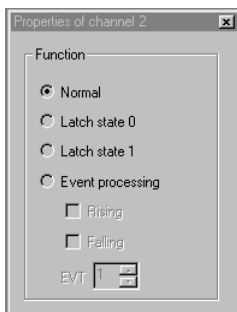
If both types of edge are selected on a channel, a single event number is associated with the channel.

Processing of event 0 (Evti) has the **highest priority**, and this can only be assigned to **channel 0**.

Double-clicking on the "Function" column of the channel for which parameters are to be set opens the Properties dialog box.

Four selection buttons determine the function supported exclusively by the channel :

- Normal,
- Latch state 0,
- Latch state 1,
- Event processing on a rising or falling edge or **both** with the associated number.



### Notes :

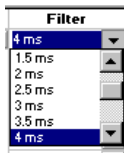
- The proposed event number is the first one available in the list. A number entered manually which is out of range will not be accepted on confirmation.
- Event numbers can only be added, deleted or changed in online mode.

### 2.5-3 Filtering

This box enables the filtering time of the selected channel to be configured.

#### In-rack discrete input modules

A pulldown list box for each channel enables the filtering time of the selected channel to be configured in increments or decrements of 0.5 ms from 0.1 to 7.5 ms.

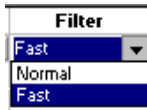


**Note** : Filtering can only be modified in online mode.

#### TBX distributed discrete input modules

One pulldown list box for each channel enables the filtering time of the selected channel to be configured :

- Normal : typical filtering of 5.7 ms
- Fast : typical filtering of 0.7 ms



### 2.5-4 External power supply fault monitoring

This is activated from the module configuration screen by means of a check box which makes it possible to monitor the sensor voltage of a group of 16 consecutive channels.

To access other groups (such as modules whose number of channels is greater than 16), click on the scroll bar pointing to the bottom of the list of channels, as shown in the example earlier.

Monitoring is active by default (box checked).

**Warning** : Deactivating monitoring may cause safety problems for the people using the application.

For discrete module versions later than V2.0 (the version number is given on the label on the side of the module), it is not possible to deactivate external power supply fault monitoring : leave this function active (default configuration). If this monitoring is deactivated by mistake, the user is warned by the Diagnostics function after transfer and connection, and this selection can be modified online.

**Note** : Monitoring can only be modified in online mode.

## 2.5-5 Latching

This function is used to select whether a positive pulse with a duration less than that of the task period should be taken into account or not.

A check box for each channel is used to activate or inhibit the latching function.

Channe	Symbol	Task	Filter	Latching
0		MAST	Fast	<input checked="" type="checkbox"/> Active
1				<input checked="" type="checkbox"/> Active
2				<input checked="" type="checkbox"/> Active
3				<input checked="" type="checkbox"/> Active
4				<input checked="" type="checkbox"/> Active
5				<input checked="" type="checkbox"/> Active
6				<input checked="" type="checkbox"/> Active
7				<input checked="" type="checkbox"/> Active
8			Fast	<input checked="" type="checkbox"/> Active
9				<input checked="" type="checkbox"/> Active
10				<input checked="" type="checkbox"/> Active
11				<input checked="" type="checkbox"/> Active
12				<input checked="" type="checkbox"/> Active
13				<input checked="" type="checkbox"/> Active
14				<input checked="" type="checkbox"/> Active
15				<input checked="" type="checkbox"/> Active

## 2.5-6 Wiring check

This function continuously checks the quality of the link between the TBX module sensors and inputs.

A check box for each channel is used to activate or inhibit the wiring check.

Channe	Symbol	Task	Wiring Check
0		MAST	<input checked="" type="checkbox"/> Active
1			<input checked="" type="checkbox"/> Active
2			<input checked="" type="checkbox"/> Active
3			<input checked="" type="checkbox"/> Active
4			<input checked="" type="checkbox"/> Active
5			<input checked="" type="checkbox"/> Active
6			<input checked="" type="checkbox"/> Active
7			<input checked="" type="checkbox"/> Active



## 2.6 Output modules

### 2.6-1 In-rack discrete output parameters

Discrete output modules have parameters for each channel or for each group of 8 channels as shown in the following table :

Reference	Group of 8 channels				Chann./Chann.
	Task	Reactivation	Fallback mode	Supply monit.	Fallback value
<b>TSX DSY 08R4D</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DSY 08R5A</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DSY 16S5</b> 16 outputs	<b>Mast/Fast/</b> None	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DSY 08S5</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DSY 08T2</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 08T22</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 08T31</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 16T2</b> 16 outputs	<b>Mast/Fast/</b> None	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 16T3</b> 16 outputs	<b>Mast/Fast/</b> None	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 32T2K</b> 32 outputs	<b>Mast/Fast/</b> None	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 64T2K</b> 64 outputs	<b>Mast/Fast</b> None	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DSY 08R5</b> 8 outputs	<b>Mast/Fast</b>	-	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DSY 16R5</b> 16 outputs	<b>Mast/Fast/</b> None	-	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DSY 16S4</b> 16 outputs	<b>Mast/Fast/</b> None	-	<b>Fallback/</b> Maintain	-	<b>0/1</b>
<b>TSX DMY 28FK</b> outputs	<b>Mast/Fast</b> None	<b>Programmed/</b> Automatic(1)	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DMZ 16DTK</b> outputs	<b>Mast/Fast</b> None	<b>Programmed/</b> Automatic(1)	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1</b>
<b>TSX DMY 28RFK</b> outputs	<b>Mast/Fast</b> None	<b>Programmed/</b> Automatic(1)	<b>Fallback/</b> Maintain	<b>Active/Inactive</b>	<b>0/1, Continuous</b>
<b>TSX PAY 262</b> <b>TSX PAY 282</b> outputs	<b>Mast/Fast</b> None	-	-	-	-

**Notes :**

- The parameters in bold correspond to the default parameters.
  - The first group of module channels (addresses 0 to 7) is always assigned to a Mast or Fast task.  
The following groups also have the option : None (no task assigned to a group of unused channels).
- (1) Reactivation is selected globally for the 12 output channels.

**2.6-2 TBX distributed discrete output parameters**

TBX output modules have parameters for each channel, for each group of 8 channels or for all module channels. The parameters can be configured according to the options proposed in the following table :

Reference	Module	Group of 8 channels		Channel by channel	
		Reactivation	Fallback mode	Fallback value	Wiring check
<b>TBX CSP 1625</b> 16 inputs	<b>Mast/Fast</b>	-	-	-	-
<b>TBX DSS 1622</b> 16 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	-
<b>TBX ESP 1622</b> 16 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	-
<b>TBX DSS 16C22</b> 16 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	<b>yes/no</b>
<b>TBX DMS 16C22</b> 16 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	<b>Yes/No</b>
<b>TBX DMS 16C222</b> 16 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	<b>Yes/No</b>
<b>TBX ESP 08C22</b> 8 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	<b>Yes/No</b>
<b>TBX CSP 1622</b> 16 outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	-	-	-
<b>TBX DSS 1235</b> 12 outputs	<b>Mast/Fast</b>	-	<b>Fallback/</b> Maintain		<b>0/1</b> -
<b>TBX DSS 1625</b> 16 outputs	<b>Mast/Fast</b>	-	<b>Fallback/</b> Maintain		<b>0/1</b> -
<b>TBX DMS 1025</b> module outputs	<b>Mast/Fast</b>	-	<b>Fallback/</b> Maintain	<b>0/1</b>	-
<b>TBX DMS 1625</b> module outputs	<b>Mast/Fast</b>	-	<b>Fallback/</b> Maintain	<b>0/1</b>	-
<b>TBX DMS 16S44</b> module outputs	<b>Mast/Fast</b>	-	<b>Fallback/</b> Maintain	<b>0/1</b>	-
<b>TBX DMS 16P22</b> module outputs	<b>Mast/Fast</b>	<b>Programmed/</b> Automatic	<b>Fallback/</b> Maintain	<b>0/1</b>	-

**Note :** The parameters in bold correspond to the default parameters.

**Notes**

1. For discrete TBX modules, it is impossible to reactivate tripped outputs from the application-specific screen in online mode.
2. For TBX DMS16P22 modules, the type of channel (input or output) for each of the channels 8 to 15 is selected in the "Type" column which appears on selecting "Programmable" from the pulldown list menu6 (see section 2.2).
3. For discrete TBX DMS 16P22 modules, it is impossible to reconfigure the type of discrete channel in online mode (a channel which is programmed as an input cannot be reconfigured as an output).

**2.6-3 Momentum distributed discrete output parameters**

With Momentum distributed discrete input modules, the task can only be selected for all the module channels.

The parameters can be configured according to the options proposed in the following table :

<b>Momentum module references</b>	<b>Number of channels</b>	<b>Associated task (for the module)</b>
<b>170 ADO 340 00</b>	16 outputs	<b>Mast</b> / Fast
<b>170 ADO 350 00</b>	32 outputs	<b>Mast</b> / Fast
<b>170 ADO 530 50</b>	8 outputs	<b>Mast</b> / Fast
<b>170 ADM 350 10</b>	16 inputs, 16 outputs	<b>Mast</b> / Fast
<b>170 ADM 390 30</b>	10 inputs, 8 outputs	<b>Mast</b> / Fast
<b>170 ADM 370 10</b>	10 inputs, 8 outputs	<b>Mast</b> / Fast

**Notes :**

- The parameters in bold correspond to the default parameters.
- If the required reference does not appear in the list above, OTHER must be selected (see Standard Profile documentation).

## 2.7 Modifying output parameters

### 2.7-1 Types of task

This parameter defines the task in which the outputs will be updated.

The "Task" parameters can be accessed via pulldown menus in the "Task" column.

To access other groups (such as in-rack discrete modules with more than 16 channels), click on the scroll bar pointing to the bottom of the list of channels, as shown in the example below :

Chan.	Symbol	Task	Reactivate	Fall. Mode	Fall. Value
0		MAST	Programmed	Fallback	0
1					0
2					0
3					0
4					0
5					0
6					0
7					0
8		MAST	Programmed	Fallback	0
9		MAST			0
10		FAST			0
11		NONE			0
12					0
13					0
14					0
15					0

#### In-rack discrete output modules

Each pulldown list box defines, for 8 consecutive channels, the task at the end of which the outputs will be updated :

- **Fast** task
- or **Mast** task
- or **None** if the group of channels is not used.

#### Distributed discrete output modules

The pulldown list defines, for all the module channels, the task at the beginning of which the outputs will be updated :

- **Fast** task
- or **Mast** task

**Note** : Tasks can only be modified in offline mode.

**2.7-2 Fallback mode**

The "Fallback" parameters can be accessed via pulldown menus in the "Fallback" column.

These selection boxes are used to define the fallback mode taken by the outputs when the PLC changes to STOP, or on a processor fault, a rack fault or an inter-rack cable fault.

- **Fallback**: the channels are set to 0 or 1 depending on the fallback value configured for the corresponding group of 8 channels (0 by default),
- **Maintain**: the outputs remain in the state in which they were prior to the change to STOP.

Chan.	Symbol	Task	Reactivate	Fall. Mode	Fall. Value
0		MAST	Programmed	Fallback	0
1					0
2					1
3					0
4					0
5					0
6					0
7					0
8		MAST	Programmed	Fallback	0
9					0
10					0
11					0
12					0
13					0
14					0
15					0

After selecting the fallback mode, the user can define the value for each of the channels in the groups. Select the channel in the "Fallback value" column, then select 1 in the pulldown list box for "Fallback to 1" or 0 for "Fallback to 0".

**Note** : Fallback mode can only be modified in online mode (for in-rack discrete I/O modules).

### 2.7-3 Reactivating the outputs

The "Reactivation" parameters can be accessed via pulldown menus in the "Reactivate" column.

The configuration editor offers a reactivation box for each group of 8 channels.

The reactivation modes are :

- "**Programmed**" indicates programmed mode : this is the default parameter taken by the outputs. Reactivation will be executed via a PLC application command or via the debug screen. In order to avoid repetitive reactivations in close succession, the module automatically ensures a time delay of 10 seconds between two reactivations.
- "**Automatic**" designates automatic mode : reactivation is carried out automatically every 10 seconds until the fault disappears.

Chan.	Symbol	Task	Reactivate	Fall. Mode	Fall. Value
0		MAST	Programmed	Fallback	0
1			Programmed		0
2			Automatic		0
3					0
4					0
5					0
6					0
7					0
8		MAST	Programmed	Fallback	0
9					0
10					0
11					0
12					0
13					0
14					0
15					0

Reactivation does not have any effect on an inactive channel or a channel without a fault.

#### Notes :

- Monitoring can only be modified in online mode (for in-rack discrete I/O modules).
- For the TSX DMY 28FK module, reactivation is selected globally for the 12 output channels.
- For discrete TBX modules, it is impossible to reactivate tripped outputs from the application-specific screen in online mode.

### 2.7-4 External power supply fault monitoring

A check box for each group of 16 channels, situated in the "Supply Monit." column, is used to activate or inhibit external supply fault monitoring.

Chan.	Symbol	Supply Monit.	Task	Reactivate	Fall. Mode	Fall. Value
16		<input checked="" type="checkbox"/> Active	MAST	Programmed	Fallback	0
17						0
18						0
19						0
20						0
21						0
22						0
23						0
24			MAST	Programmed	Fallback	0
25						0
26						0
27						0

**Note :** Monitoring can only be modified in online mode.

### 2.7-5 Wiring check

This function continuously checks the quality of the link between the TBX module actuators and outputs.

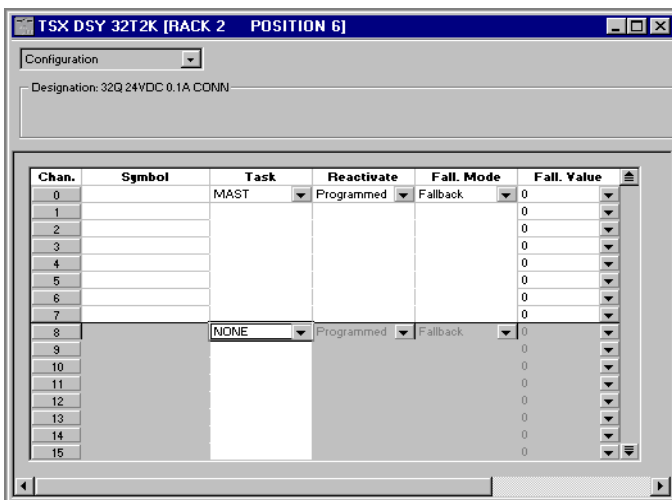
A check box for each channel is used to activate or inhibit the wiring check.

Chan.	Symbol	Task	Reactivate	Fall. Mode	Fall. Value	Wiring Check
0		MAST	Automatic	Fallback	0	<input checked="" type="checkbox"/> Active
1					0	<input checked="" type="checkbox"/> Active
2					0	<input checked="" type="checkbox"/> Active
3					0	<input checked="" type="checkbox"/> Active
4					0	<input checked="" type="checkbox"/> Active
5					0	<input checked="" type="checkbox"/> Active
6					0	<input checked="" type="checkbox"/> Active
7					0	<input checked="" type="checkbox"/> Active
8			Automatic	Fallback	0	<input checked="" type="checkbox"/> Active
9					0	<input checked="" type="checkbox"/> Active
10					0	<input checked="" type="checkbox"/> Active
11					0	<input checked="" type="checkbox"/> Active
12					0	<input checked="" type="checkbox"/> Active
13					0	<input checked="" type="checkbox"/> Active
14					0	<input checked="" type="checkbox"/> Active
15					0	<input checked="" type="checkbox"/> Active

## 2.8 Deconfiguring and reconfiguring groups of channels

Deconfiguration is carried out via the configuration screen for all the groups of channels other than the 0 to 7 group, by assigning the "Task" type to None.

After confirming the modification, the deconfiguration operation applies to the group itself and the following groups.



Reconfiguration is carried out via the same configuration screen, by re-assigning the "Task" to Mast or Fast.

After confirming the modification, the reconfiguration of a group also reconfigures the preceding groups. Parameters are therefore found in the state in which they were previously programmed.

### Note :

It is recommended that groups of unused channels are not configured in the application. Therefore, even if no connector is connected, the module will not signal a fault.

### Notes :

- Deconfiguration or reconfiguration cannot be executed in online mode.
- Distributed discrete I/O do not offer the option of deconfiguring groups of channels.

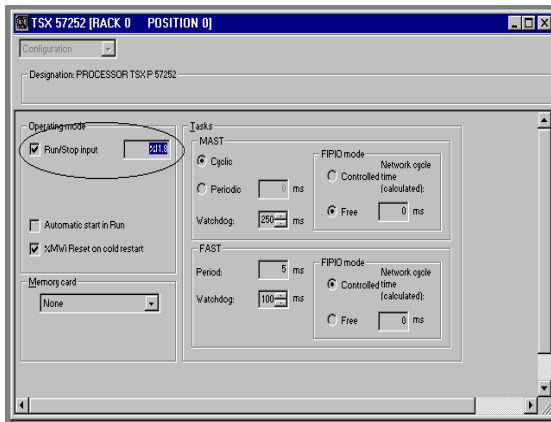


## 2.9 Setting the parameters of the RUN/STOP input

The RUN/STOP function can be assigned to an input of an in-rack discrete module. It enables the execution of the application program to be started (RUN) or stopped (STOP).

### Note :

A STOP command from the physical input assigned to the RUN/STOP input has priority over a RUN command from a terminal or network.



This input must be configured in order to perform these functions.

### Procedure :

- Double-click on position 0 or 1 (processor slot) on the configuration screen,
- Check the **Run/Stop Input** box,
- Enter one of the input bits in the data entry window.

Two Run/Stop icons in the module configuration indicate the number of the Run/Stop channel.



### Warning

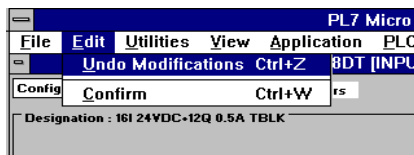
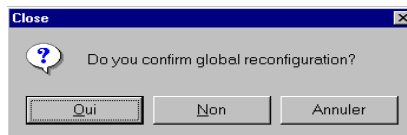
When a channel is configured as a RUN/STOP input, it is not advisable to modify the module configuration in online mode, as this stops the PLC.

## 2.10 Confirming the configuration

### 2.10-1 Confirming after modification

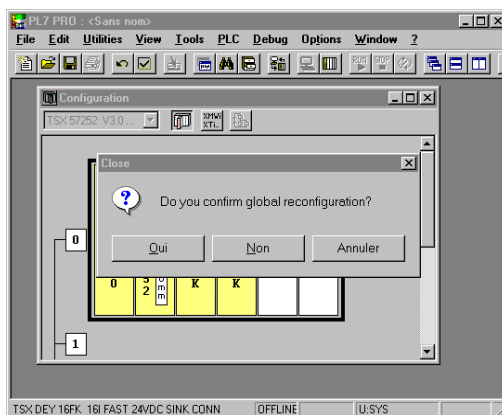
When quitting the function after modifying the channel configuration parameters of a module, the new configuration must be confirmed. This can be done in several ways :

1. Confirm using the toolbar by clicking on the corresponding icon or via the Confirm command in the shortcut menu.
2. Quit the function without confirming the parameters. This displays a dialog box which enables the user to confirm the new configuration.
3. Pull down the PL7 **Edit** menu and select **Confirm**.



### 2.10-2 Global reconfiguration

Quitting the configuration editor after modifying all the channel configuration parameters of all the modules requires a global reconfiguration. When the editor is closed, a dialog box enables this global reconfiguration to be confirmed.



---

Global reconfiguration must be performed offline so that the modifications confirmed for each module are taken into account in the application.

This reconfiguration is carried out :

- either via the "Confirm" icon or by using the **Edit/Confirm** command or the Confirm command in the shortcut menu,
- or by closing the configuration editor, without having carried out a global confirmation, and confirming the global reconfiguration.



### 3.1 Introduction to the Debug function

This function can only be accessed in online mode (**PLC** menu, **Connect** command, or click on the corresponding icon). For each discrete I/O module of the application, it displays the parameters of each of the channels (channel status, filter value, etc) and accesses diagnostics and adjustment of the selected channel (channel forcing, channel masking, etc).

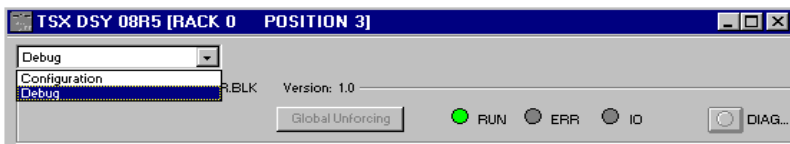
The function also gives access to module diagnostics in the event of a fault.

The discrete I/O module **Debug** function can be accessed by double-clicking on the Station, Configuration, and Hardware configuration icons in the Application Browser and then on the position of the module in the rack.

By default, the **Debug** function is selected in online mode, and the pulldown dialog box in the command zone can be used to return to the **Configuration** function.



online mode



#### Notes

Version PL7 < V3.3 : Momentum and TBX modules are not accessible in online mode.

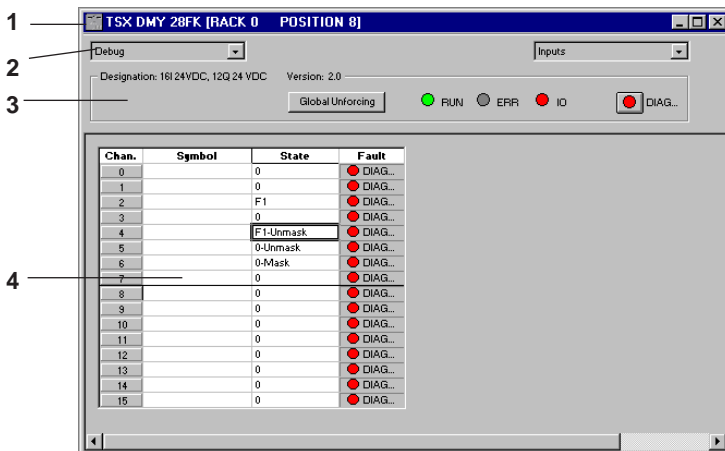
Version PL7  $\geq$  V3.3 with CPU V $\geq$ 3.3 and IOB V3.3 : Momentum and TBX modules are accessible in online mode with up to 4 screens open.

#### Warning

Maximum user rights are required in order to access the Force/Unforce and Write commands.

## 3.2 Description of the debug screen

This screen shows the selected module and displays the state of each of its channels in realtime. It is also used to access the channel commands (forcing the input or output value, reactivating outputs, etc).



- 1 This line displays the catalog reference and the location of the module in the PLC (rack and position).
- 2 This command zone displays the current function (**Debug** function) and enables the **Configuration** function to be selected from the pulldown list box.
- 3 This "module" zone contains the module description and version. It also has a copy of the module status indicator lamps (RUN, ERR, I/O) and two command buttons which are used to :
  - access the module diagnostics in the case of a module fault; this is signaled by the indicator lamp integrated in the diagnostics access button, which turns red.
  - globally remove all channel forcing,
- 4 This "channel" zone displays the value and the state of each of the module channels in realtime :
  - **Channel** : number of the input or output channel,
  - **Symbol** : symbol defined by the user and associated with the language object for the channel. If the channel has no associated symbol, this field is empty,
  - **State** : channel state
    - 0 or 1
    - indication of channel forcing : F0 if the channel is forced to 0 or F1 if the channel is forced to 1
    - indication of masking/unmasking
    - Set or Reset

- 
- **Fault** : access to channel diagnostics
  - **Reactivation** : access to output reactivation
  - **Applied outputs** : output applied by the PLC program or in fallback mode

**Notes**

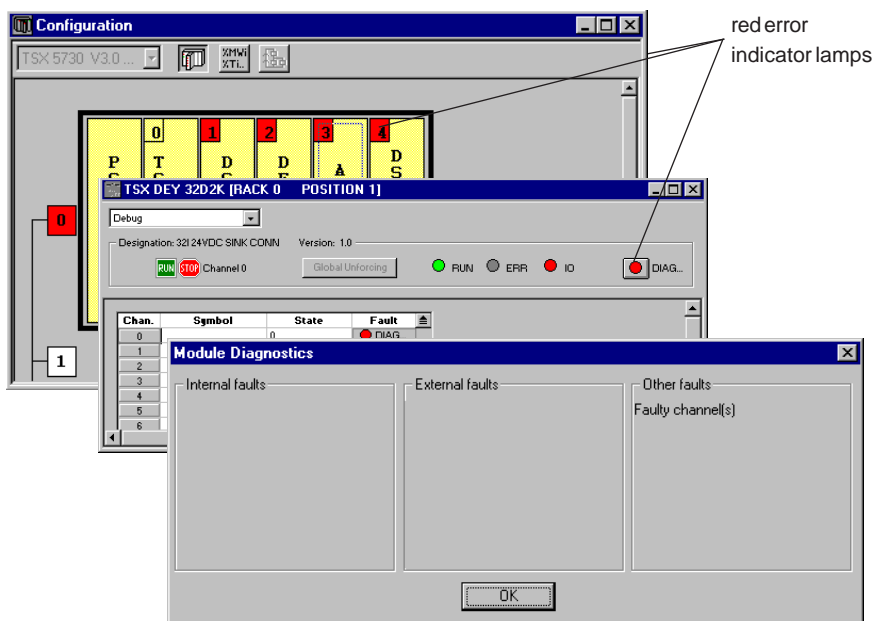
1. For discrete TBX modules, it is impossible to reactivate tripped outputs from the application-specific screen in online mode.
2. For TBX DMS16P22 modules, the type of channel (input or output) for each of the channels 8 to 15 is selected in the "Type" column which appears on selecting "Programmable" from the pulldown list menu **6** (see section 2.2).
3. For discrete TBX DMS 16P22 modules, it is impossible to reconfigure the type of discrete channel in online mode (a channel which is programmed as an input cannot be reconfigured as an output).

### 3.3 Displaying the module diagnostics

When a module is faulty, some of the indicator lamps which can be accessed in the configuration editor screen turn red :

- indicator lamp showing the position of the module on the screen in which it is displayed (first screen in the configuration editor),
- copy of the ERR and I/O module indicator lamps, in the "module" zone,
- indicator lamp integrated in the DIAG command button, also in the "module" zone.

Activating the DIAG command button also gives access to the **Module Diagnostics** screen which shows the current module faults, classified according to their category : internal faults, external faults or other faults.



#### List of module faults

- **Internal faults** : Module failure
- **Other faults** : Faulty channel(s) (details in the channel diagnostics)  
Terminal block fault, self-test in progress, configuration fault, module absent or not powered-up

#### Note

When there is a configuration fault or a module is absent, the module diagnostics screen cannot be accessed. The following message will appear on the screen : "The module is absent or different from the one configured in this position!"



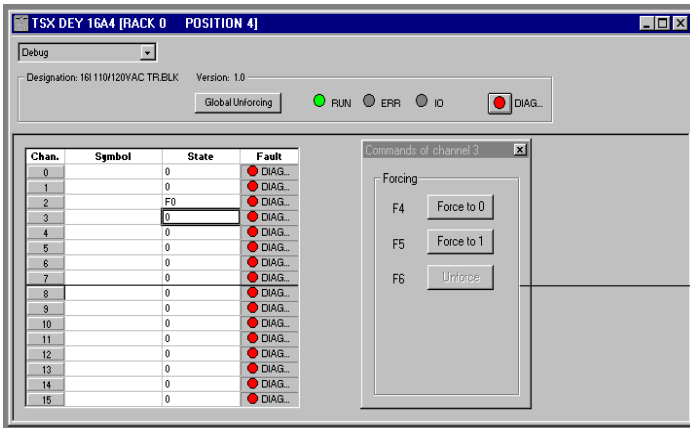
### 3.4 Canceling forcing on the channels of a module

The **Global Unforcing** command button is used to cancel forcing on all channels of a module.



### 3.5 Accessing channel commands

The commands for a channel can be accessed by double-clicking on a channel number. The Properties dialog box **(1)** appears, which can be used to control the selected channel.



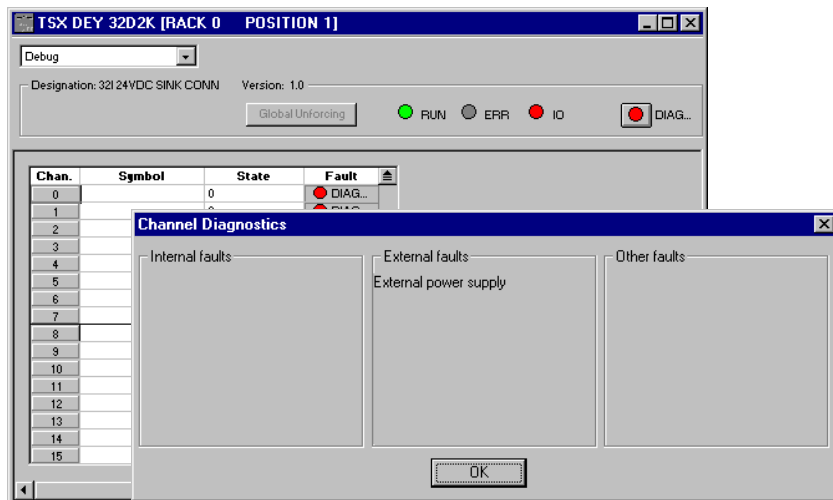
Selecting a new channel in the display zone accesses the commands in the Properties dialog box **(1)** which displays the number of the selected channel in realtime.

The Properties dialog box is accessed using one of the following methods :

- by clicking with the right mouse button on the table cell corresponding to the channel to be controlled and selecting Properties in the pulldown list
- by double-clicking with the left mouse button on the table cell corresponding to the channel to be controlled.

### 3.5-1 Displaying detailed channel diagnostics

When a channel is faulty, the DIAG button in the **ERR** column becomes active. Activating this button accesses a "channel" diagnostics screen (identical to the "module" diagnostics screen) which shows the channel faults, classified according to their category : internal faults, external faults or other faults.



#### List of channel faults

- **Internal faults** : Module failure
- **External faults** : Sensor link fault  
Sensor supply fault
- **Other faults** : Terminal block fault  
Configuration fault  
Communication fault

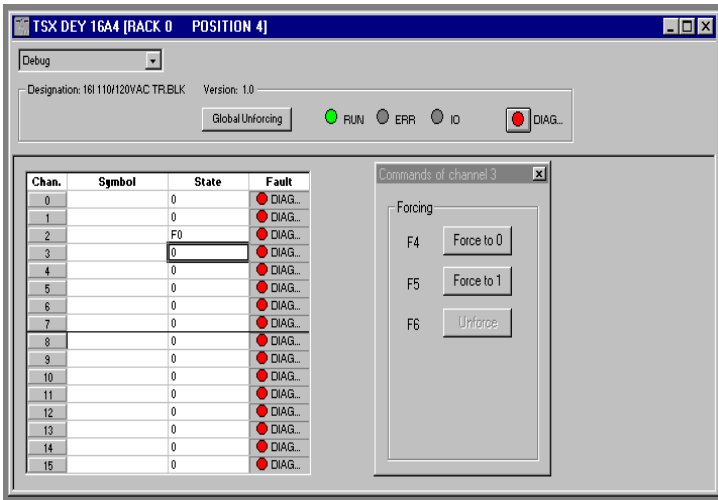
#### Note

The channel diagnostics can also be accessed by program (READ\_STS instruction).

### 3.5-2 Forcing or unforcing channels

Three buttons in the Properties dialog box are used to perform these functions :

- **Force to 0** using the F4 shortcut key (state F0),
- **Force to 1** using the F5 shortcut key (state F1),
- **Unforce** using the F6 shortcut key.



**Note :**

Forcing can be carried out on a multiple selection.

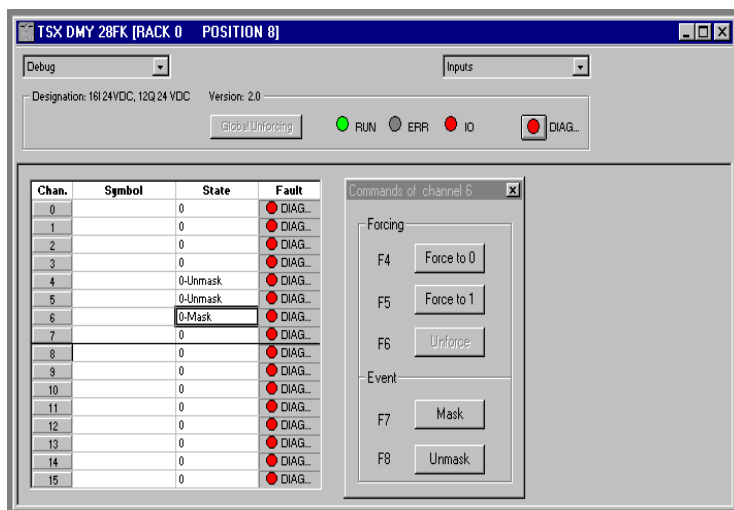
The **Unforce** command is accessible if the selected channel is forced.

The **Global Unforcing** command (module zone) is accessible as soon as a module channel is forced.

### 3.5-3 Masking or unmasking events

Two buttons in the Properties dialog box are used to perform these functions :

- **Mask** masks events detected on the configured channel during event processing, using the F7 shortcut key, (Mask state)
- **Unmask** inhibits the masking of events detected on the configured channel during event processing, using the F8 shortcut key (Unmask state).

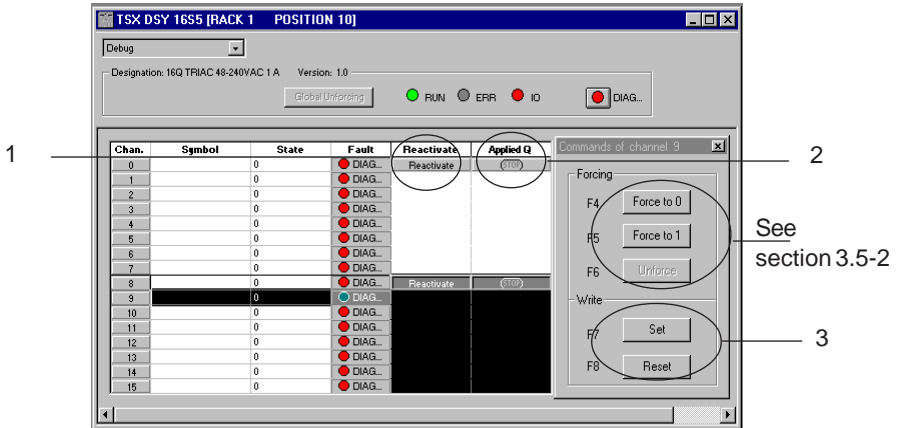


#### Notes:

- Masking or unmasking can be carried out on a multiple selection.
- The mask/unmask functions are offered for TSX DEY 16FK and TSX DMY 28FK fast input modules, and the 4 virtual outputs of the TSX DMY 28 RFK module.
- Global masking is performed by :
  - the PL7 MASKEVT() instruction
  - the PL7 UNMASKEVT() instruction
  - system bit %S38
  - or the event activation/deactivation button in the CPU debug screen

### 3.5-4 Reactivation command (discrete outputs)

The reactivation command (1) is carried out for groups of 8 channels. Each group displayed has its own button. The shortcut keys for the two channel groups are F2 and F3.



#### Notes

Reactivation is selected globally for the 12 output channels of the TSX DMY 28FK and TSX DMY 28 RFK modules.  
 For discrete TBX modules, it is impossible to reactivate tripped outputs from the application-specific screen in online mode.

### 3.5-5 Applied outputs (discrete outputs)

A red STOP indicator (2) is displayed when the outputs are in the fallback state. This indicator lamp is not active when the outputs are applied correctly by the PLC.

### 3.5-6 Write command (discrete outputs)

Two buttons (3) in the Properties dialog box are used to perform these functions :

- **Set** to write the value 1 using the F7 shortcut key,
- **Reset** to write the value 0 using the F8 shortcut key.

#### Note :

Writing to 1 or 0 can be carried out on a multiple selection. If the user makes a multiple selection, the Properties window shows the operations which can be performed on all the selected channels.



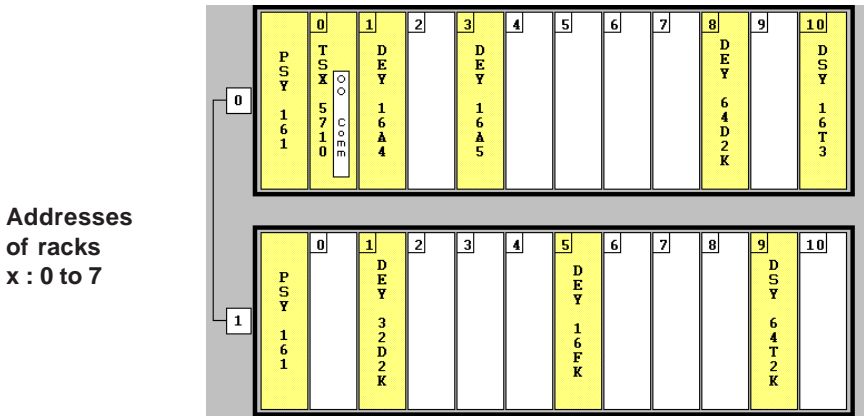
### 4.1 In-rack discrete I/O module object addressing

This explains the principle of addressing I/O image bits for TSX Premium PLC in-rack discrete modules.

Channel addressing is geographical, in other words, it depends on :

- the rack number (address),
- the physical position of the module in the rack,
- the module channel number.

**Positions of modules y : 00 to 10**



**Addresses of racks x : 0 to 7**

Address syntax for discrete I/O :

%	I or Q	address rack x	position module y	.	Channel no. i
Symbol	<b>Object type</b> I = input Q = output	<b>x = 0 to 7</b>	<b>y = 00 to 10</b>	Point	i = 0 to 63

**Example :**

%I101.5 : image bit of input 5 of the module in position 1 in rack address 1.

%Q10.3 designates image bit of output 3 of the module in position 10 in rack 0.

**Note :**

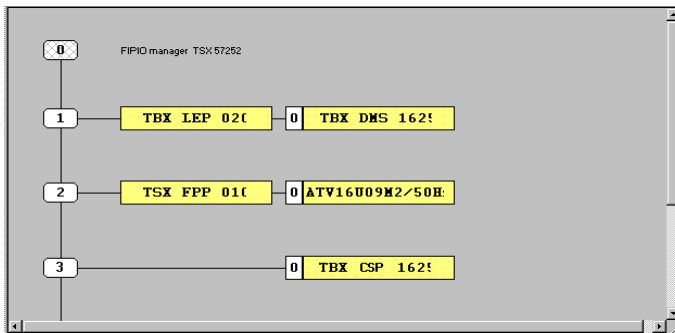
Discrete modules can address other types of object : %IW, %QW, %MW. The complete syntax for these objects appears in the part entitled "Common features of application-specific functions".

## 4.2 Addressing distributed discrete I/O modules

This explains the principle of addressing I/O image bits for TSX Premium PLC distributed discrete modules.

Channel addressing is geographical, in other words, it depends on :

- the connection point,
- the type of module : base or extension,
- the channel number.



Address syntax for distributed discrete I/O :

<b>%</b>	<b>I or Q</b>	<b>\</b>	<b>p.2.c</b>	<b>\</b>	<b>m</b>	<b>.</b>	<b>i</b>
Symbol	Object type I = input Q = output		Module/channel address and connection point p= 0 or 1 processor address 2= FIPIO interface c= connection point no. 1 to 127		Module number 0 for base 1 for extension		Channel no. 0 to 31

### Example :

%I\0.2.6\0.5 : image bit of input 5 of the distributed input base module at connection point 6 on the FIPIO bus.

%Q\0.2.8\1.7 : image bit of input 7 of the distributed output extension module at connection point 8 on the FIPIO bus.

### Note :

Discrete modules can address other types of object : %IW, %QW, %MW. The complete syntax for these objects appears in the part entitled "Common features of application-specific functions".



---

### 4.3 Language objects associated with discrete I/O

---

A discrete I/O module configured at a given position automatically generates a series of language objects for programming and diagnosing this I/O module.

**Note :**

Configuration constants for discrete I/O modules

These are organized in groups of eight channels with the address of the first channel for each group, for example :

`%KW@module.0.r`: where  $r = 0, 1$  or  $2$  for channels 0 to 7

`%KW@module.8.r` : where  $r = 0, 1$  or  $2$  for channels 8 to 15 and so on.

These objects can only be accessed in **read** mode.

where `@module` = module address

---

#### 4.3-1 Implicit exchange objects

These are objects which are exchanged automatically on each scan of the task in which the module channels are configured :

**Bits :**

- `%I@module.i` : bit : input channels. At state 1, it indicates that input channel  $i$  of the module in position  $y$  that the output of the sensor controlling the input is activated.
- `%Q@module.i` : bit : output channels. At state 1, it indicates that the output channel  $i$  of the module in position  $y$  is activated.
- `%I@module.i.ERR` : bit : channel fault. At state 1, it indicates that input channel  $i$  of the module in position  $y$  is faulty.
- `%I@module.MOD.ERR` : bit : module fault. At state 1, it indicates that the module in position  $y$  is faulty.

where `@module` = module address :

- `xy` for in-rack modules
- `\p.2.c\m` for distributed modules

**Note :**

All the discrete module I/O objects can be indexed (except for TBX DMS 16P22 programmable module I/O objects).

For in-rack TSX modules and TBX modules, `%I` and `%Q` objects are indexed independently.

For Momentum modules, indexing on `%I` objects takes place on `%Q` objects in the same module.

---

---

**Event status associated with the channel : (1)**


---

Address	Meaning
%IW@module.i:X0	Rising edge
%IW@module.i:X1	Falling edge
%Qw@module.i:X0	Mask/Unmask - Masks the event associated with the channel

(1) The event (EVT) status is only updated when the Evt occurs.

---

**4.3-2 Explicit exchange objects**

Explicit exchange objects are updated on request.

**Words :**

- **%MW@module.i.0**: channel exchange in progress

Address	Meaning
%MW@module.i.0:X0	Status parameter exchange in progress
%MW@module.i.0:X1	Command parameter exchange in progress

- **%MW@module.i.1**: channel report

Address	Meaning
%MW@module.i.1:X0	Status parameter exchange report
%MW@module.i.1:X1	Command parameter exchange report

- **%MW@module.i.2:Xj** : channel status, where j : 0 to 15.

Address	Meaning
%MW@channel.i.2:X0	External fault : Trip
%MW@channel.i.2:X1 (1)	External fault : Fuse
%MW@channel.i.2:X2	Terminal block fault
%MW@channel.i.2:X4	Internal fault : Module inoperative
%MW@channel.i.2:X5	Hardware or software configuration fault
%MW@channel.i.2:X6	Communication fault
%MW@channel.i.2:X7	Reserved
%MW@channel.i.2:X8	External fault : Short-circuit
%MW@channel.i.2:X9	External fault : Line fault
%MW@channel.i.2:X10 to X15	Reserved

(1) Momentum I/O : for a minor fault outside the base unit, signaling will depend on the base unit selected (see Momentum documentation).

- **%MW@module.MOD.2:Xj** : module status

Address	Meaning	Module type
<b>%MW@module.MOD.2:X0</b>	Internal fault : Module inoperative	Base
<b>%MW@module.MOD.2:X1</b>	Operational fault (extension or com.)	Base
<b>%MW@module.MOD.2:X2</b>	Terminal block fault	Base
<b>%MW@module.MOD.2:X3</b>	Self-test in progress	Base
<b>%MW@module.MOD.2:X4</b>	Reserved	Base
<b>%MW@module.MOD.2:X5</b>	Hardware or software configuration fault	Base
<b>%MW@module.MOD.2:X6</b>	Module absent	Base
<b>%MW@module.MOD.2:X7</b>	FIPIO extension module fault	Base
<b>%MW@module.MOD.2:X8</b>	Internal fault : Module inoperative	FIPIO extension
<b>%MW@module.MOD.2:X9</b>	Operational fault (extension or com.)	FIPIO extension
<b>%MW@module.MOD.2:X10</b>	Terminal block fault	FIPIO extension
<b>%MW@module.MOD.2:X11</b>	Self-test in progress	FIPIO extension
<b>%MW@module.MOD.2:X12</b>	Reserved	FIPIO extension
<b>%MW@module.MOD.2:X13</b>	Hardware or software configuration fault	FIPIO extension
<b>%MW@module.MOD.2:X14</b>	Module absent	FIPIO extension
<b>%MW@module.MOD.2:X15</b>	Reserved	FIPIO extension

**Note** : For FIPIO distributed I/O with a base module connected to an extension module, only the status word of the base module is significant. Its low order byte is assigned to the base module, its high order byte is assigned to the extension module.

- **%MW@channel.i.3:X0**: channel command where i = first channel in a group of channels (0, 8, 16, etc)

Specific addressing of output modules with reactivation

Address	Meaning
<b>%MW@channel.3:X0</b>	Reactivation of tripped outputs (protected outputs)
<b>%MW@channel.3:X1</b>	Inhibit external supply monitoring
<b>%MW@channel.3:X2</b>	Enable external supply monitoring
<b>%MW@channel.3:X3 to 15</b>	Reserved

---

## 4.4 Reading the status word and writing the command word

---

### 4.4-1 Reading the status word

The **READ\_STS %CHx.i** instruction associated with the channel or module enables explicit reading in the I/O module of the status word associated with the channel or module. This read operation updates **status word %MWx.i.2** or **%MWx.MOD.2**.

The status word contains information on the operating status of the module. It can be used for program diagnostics.

**Syntax :** **READ\_STS%CH** module address.channel (or MOD for module)

---

### 4.4-2 Writing the channel command word

The **WRITE\_CMD %CHx.i** instruction is for the explicit writing in the input or output module of the **channel command** word associated with the channel.

**Syntax :** **WRITE\_CMD%CH** module address.1st channel in the group

**Note :**

The **READ\_STS** and **WRITE\_CMD** instructions are executed per group of channels. To update the status of all the channels in a group of 8 channels, only one instruction is necessary (write the address of the first channel in the group).

See the examples in the part entitled "Common features of application-specific functions".

## 5.1 Presentation

### 5.1-1 Area of application of the module

The TSX DMY 28 RFK module comprises :

- 16 fast discrete inputs
- 12 fast discrete outputs

It also has 4 virtual outputs.

The purpose of this module is to provide a solution for specific applications which require, for example :

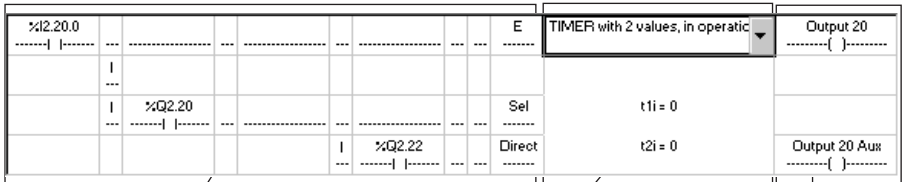
- a response time faster than the Fast task or the event-triggered task
- reaction of an output to a simple logic command less than 500  $\mu$ s
- speed control of a moving part and cessation of movement if the speed falls below a certain limit
- motion control
- time delays with a 0.1 ms time base
- generation of a continuous oscillation with a fixed frequency but a variable cyclic ratio
- etc.

### 5.1-2 Module functions

This module is used to perform control system functions executed at module level, independently of the PLC task, and using the following as input variables :

- the physical inputs of the module (%I)
- the states of the module physical and auxiliary outputs (%I)
- the commands from the PLC processor program (%Q)
- the channel fault data

Each function is programmed in configuration mode. The configuration screen for each output comprises three main parts :



Simplified Ladder rung layout (four lines of four contacts) to achieve a combinational logic function based on the module objects.

Part representing the function used which can either be direct control of the output using the configured combinational logic, or one of the function blocks listed overleaf.

Part for assigning the module physical and auxiliary outputs.

### 5.1.3 List of reflex function blocks

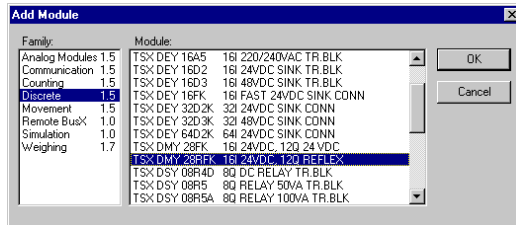
Function block name	Function description
Direct	Standard discrete outputs
Combinational logic	Control of the output by the combinational logic defined in the Ladder rung
On-delay TIMER	On-delay type timer function block
Off-delay TIMER	Off-delay type timer function block
On-delay/off-delay TIMER	On-delay/off-delay type timer function block
2-value TIMER function	On-delay type timer function block with 2 selectable values
Selectable on-delay/off-delay TIMER	Selectable on-delay / off-delay timer function block
Retriggerable MONOSTABLE	Retriggerable monostable function block
Time-delayed MONOSTABLE	Time-delayed non-retriggerable monostable function block
2-value MONOSTABLE	Monostable function block with 2 selectable values
OSCILLATOR	Oscillator function block
COUNTER OUTPUT D	Counter output D function block, edge latching
COUNTEROUTPUTT	Counter output T function block, division by 2
Double threshold COUNTER	Counter function block with 2 selectable thresholds
Single Electronic CAM	Double threshold counter function block for producing a single cam output
Single threshold INTERVAL COUNTER	Interval counter function block used to measure time or length
BURST	Function block used to generate a defined number of oscillator periods
PWM generation	Function block used to generate a continuous oscillation with a fixed frequency but a variable cyclic ratio
Slow speed detection 1	Slow speed selection detection function block
Slow speed detection 2	Speed monitoring function block
Type 1 Command/Control function	Command/control function blocks used to control an action and to check that it has been executed correctly after a certain period of time (1 single command)
Type 2 Command/Control function	Command/control function blocks used to control an action and to check that it has been executed correctly after a certain period of time (2 commands FWD and REV)
Command/Counting	Function block command during a number of counting points (standard positioning),
Fault indication	Fault indication function block

## 5.2 Configuration and parameter entry

### 5.2-1 Selecting the module

To add a reflex module to the PLC configuration :

1. **Double-click** on the position in the rack where the module is to be installed.
2. Select **Discrete** in the Family list then the module reference from the Module field : TSXDMY 28 RFK
3. Confirm with **OK**.



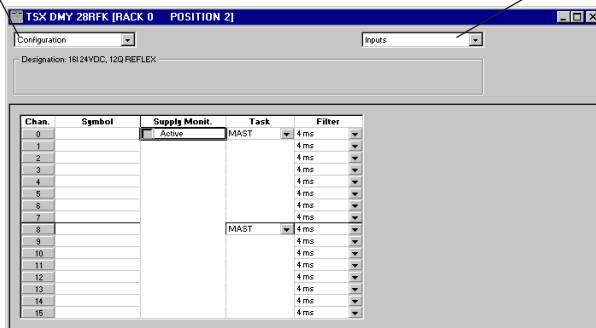
### 5.2-2 Accessing the module parameters

To enter the parameters of the module channels, double-click on the module position or execute the **Utilities/Open Module** command. The parameter entry screen is displayed.

This screen is used to enter the I/O parameters.

Access to the configuration or parameter adjustment

Selection of inputs or outputs



### 5.2-3 Entering the input parameters

See section 2.3 for information on how to use the configuration editor to enter or modify the parameters.

Chan.	Symbol	Supply Monit.	Task	Filter
0		<input checked="" type="checkbox"/> Active	MAST	4 ms
1				4 ms
2				4 ms
3				4 ms
4				4 ms
5				4 ms
6				4 ms
7				4 ms
8			MAST	4 ms
9				4 ms
10				4 ms
11				4 ms
12				4 ms
13				4 ms
14				4 ms
15				4 ms

#### External power supply fault monitoring

This function is used to disable or activate monitoring of external power supply faults on the input connector.

Check the box to activate monitoring.

#### Task

This parameter defines the PLC processor task (1) in which the image bits (%I) of the physical input channels of the module are updated.

Select the **Mast** or **Fast** task.

Note : the selection is made in groups of 8 channels.

(1) The update is performed in real time in the module Ladder diagram.

#### Input filtering

This is used to select the filtering value for each channel from among the 16 values available (possible values from 0 to 7.5 ms, default value : 4 ms).

Select the filtering value.



### 5.2-4 Entering the output parameters

See section 2.3 for information on how to use the configuration editor to enter or modify the parameters.

Chan.	Symbol	Supply Monit.	Task	Reactivate	Fall. Value	Function	Event
16		<input checked="" type="checkbox"/> Active	MAST	Programmed	Fallback 0	OSCILLATOR	
17					Fallback 1	Direct	
18					Fallback 0	TIMER in operation	
19					Fallback 0	Direct	
20					Fallback 0	Direct	
21					Fallback 0	TIMER with selectable oper	
22					Fallback 0	Direct	
23					Fallback 0	Direct	
24			MAST	Programmed	Fallback 0	Direct	
25					Fallback 0	Combinational	
26					Fallback 0	Direct	
27					Fallback 0	Direct	
28 v					Fallback 0	TIMER in operation/Idle	
29 v					Fallback 0	Direct	
30 v					Fallback 0	Direct	
31 v					Fallback 0	Direct	

#### External power supply fault monitoring

This function is used to disable or activate monitoring of external power supply faults on the output connector.

Check the box to activate monitoring.

#### Task

This parameter defines the PLC processor task (1) in which the output command bits %Q and status bits %I are updated.

Select the **Mast** or **Fast** task.

Note : the selection is made in groups of 8 channels.

(1) The update is performed in real time in the module Ladder diagram.

#### Reactivating outputs

The reactivation modes are :

- **Programmed**: The outputs take the default parameters. The outputs are reactivated by a PLC application command or via the debug screen. To avoid frequent, repeated reactivations, the module automatically provides a 10-second time delay between two reactivations.
- **Automatic** : The outputs are reactivated automatically every 10 seconds until the fault disappears.

Select **Programmed** or **Automatic**.

### Fallback mode, fallback value

These are used to define the fallback mode for each physical output of the module (1) when the PLC stops, or when there is a processor, rack or inter-rack cable fault.

Select :

**Fallback to 0** : the output is forced to 0

**Fallback to 1** : the output is forced to 1

**Maintain** : the output is maintained in its state

**Continuous** : the reflex function remains active.

(1) The auxiliary outputs do not have fallback values, as they are not real outputs.

### Function block

This can be used to define the reflex function of each channel : see section 5.3.

### Event, Virtual output

This associates an event with a virtual output.

Virtual outputs correspond to language objects to which a reflex function (which is not connected to a physical output) is applied. These outputs are identified by the symbol V next to the channel number (channels 28V to 31V).

Double-click on the cell associated with the channel in the "Event" column to select parameter entry.

The following parameters are possible :

- normal (no event associated with the channel)
- event processing channel by channel :
  - event triggered on a rising edge (RE)
  - event triggered on a falling edge (FE)
  - event triggered on a rising and falling edge

Events are associated with a processing **number (Evti)**.

These numbers range from :

- **0 to 31** for a TSX/PMX/PCX 5710 processor
- **0 to 63** for a TSX/PMX/PCX 572\*/3\*/4\* processor



If both types of edge are selected on one channel, only one event number is associated with the channel.

Performance :

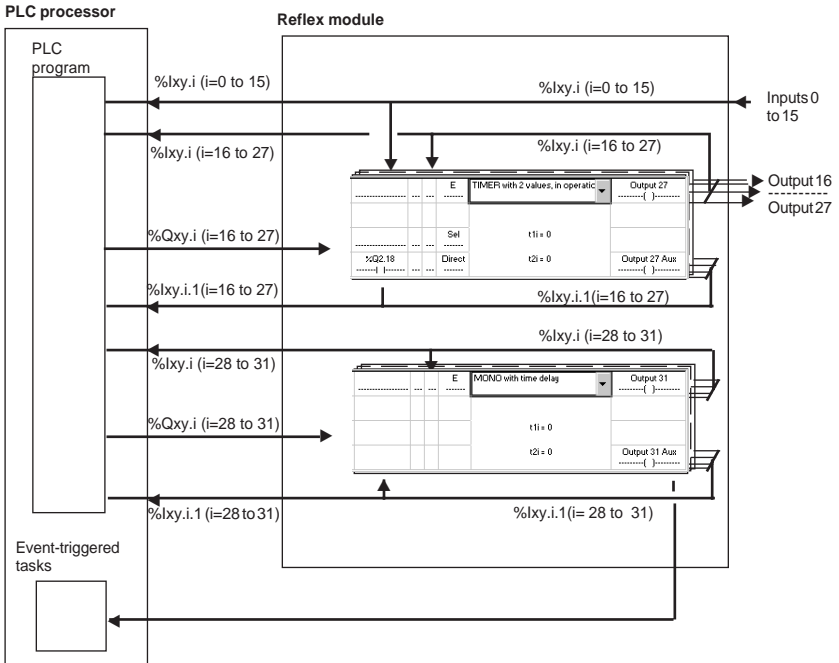
- Maximum event frequency : 1 kHz / Number of outputs programmed for event
- Maximum number of events in burst : 100 events per 100ms

### 5.3 Configuring the reflex functions

#### 5.3-1 Language objects associated with the reflex module

The language objects associated with the reflex module can be used :

- in the Ladder diagram for the module
- in the PLC processor program



#### • Module physical inputs

Syntax	Role	Use in the module Ladder diagram	Use in the PLC processor program
%Ixy.i ( i= 0 to 15)	image bits of the state of the module physical inputs	yes (read-only) updated in real time by the module	yes (read-only) updated cyclically at the rate of the task (1)

(1) Task in which the channels are configured (see sections 5.2-4 and 5.2-5).

## • Module outputs

Channels 16 to 27 correspond to the real outputs of the module.

Channels 28 to 31 correspond to the virtual outputs of the module. The virtual outputs are not physical module outputs, they operate on module internal status bits and can be associated with events. A virtual output can therefore initiate a PLC processor event-triggered task.

Syntax	Role	Use in the module Ladder diagram	Use in the PLC processor program
%Qxy.i (i= 16 to 31)	module output command bits	yes (read-only) (2)	yes (read/write) transmitted cyclically at the rate of the task (1)
%lxy.i (i= 16 to 31)	image bits of the state of the module outputs	yes (read-only) updated in real time by the module	yes (read-only) updated cyclically at the rate of the task (1)
%lxy.i.1(i= 16 to 31)	image bits of the state of the module auxiliary outputs	yes (read-only) updated in real time by the module Note : the Aux outputs are internal to the module	yes (read-only) updated cyclically at the rate of the task (1)

(1) Task in which the channels are configured (see sections 5.2-4 and 5.2-5).

(2) These command bits only control the corresponding physical outputs if the Direct function is selected. In other cases their use is optional.

## • Events

Events are activated by virtual outputs 28 to 31.

Syntax	Role	Use in the module Ladder diagram	Use in the PLC processor program
%IWxy.i (i= 28 to 31) • %IWxy.i:X0 • %IWxy.i:X1	event status word • 1=rising edge • 1=falling edge	no	yes (read-only) updated cyclically at the rate of the task (1)
%QWxy.i(i= 28 to 31) • %QWxy.i:X0	Event masking • 1=Mask • 0=Unmask	no	yes (write-only) updated cyclically at the rate of the task (1)

(1) Task in which the channels are configured (see sections 5.2-3 and 5.2-4).

---

- **Faults**

<b>Syntax</b>	<b>Role</b>	<b>Use in the module Ladder diagram</b>	<b>Use in the PLC processor program</b>
<b>ERRi</b> (i= 16 to 27)	module physical output short-circuit fault bits	yes (read-only)	no
<b>ERR28</b>	input external power supply fault bits	yes (read-only)	no
<b>ERR29</b>	output external power supply fault bits	yes (read-only)	no
<b>%lx.y.i.ERR</b> ( i= 0 to 31)	channel fault bits	no	yes (read-only) updated cyclically at the rate of the task (1)
<b>%lx.y.mod.ERR</b>	module fault bits	no	yes (read-only) updated cyclically at the rate of the task (1)

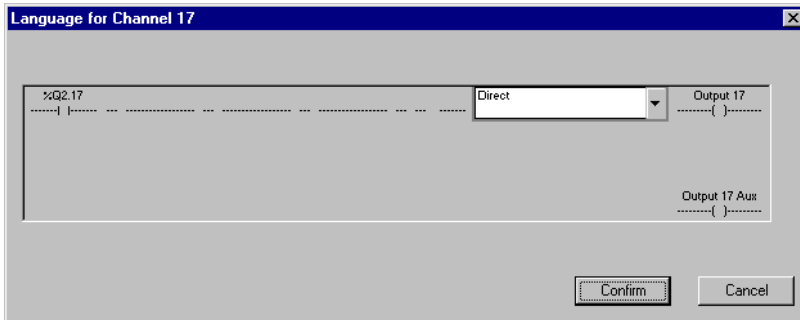
(1) Task in which the channels are configured (see sections 5.2-4 and 5.2-5).

Note: Explicit exchange objects are described in section 5.5-3. They can only be used in the PLC processor program with the help of explicit exchange instructions.

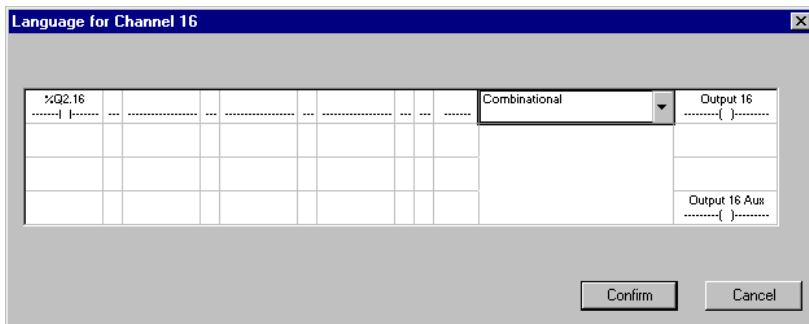
### 5.3-2 The reflex function Ladder editor

#### Accessing the editor

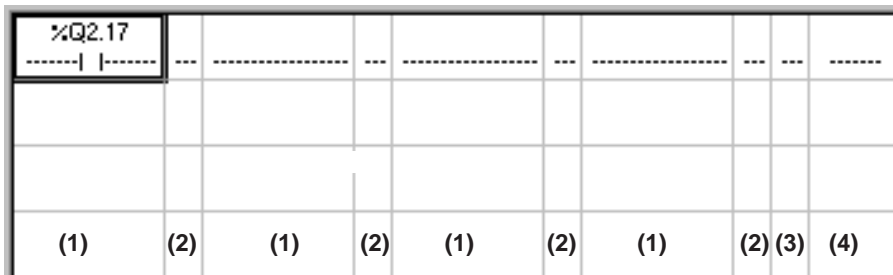
Double-click on the cell associated with the channel in the "Function" column. If the channel is not configured the "Direct" function is displayed by default.



To access the entry of the combinational logic, select the function block first.



The reflex configuration screen comprises a grid for entering the Ladder combinational logic and selecting the function block.



- (1) Columns for entering contacts
- (2) Column for entering links between contacts and divergences
- (3) Column for setting function block inputs to 1 or entering their link with the combinational logic
- (4) Column displaying the inputs relating to the selected function block
- (5) Column displaying the type of internal parameter used by the block
- (6) Column for entering the type of output coil.

.....	Combinational	▼	Output 17 -----{ }-----
(4)	(5)		(6)
			Output 17 Aux -----{ }-----

### Selecting the function block

Click on the drop-down selection list

... ..	E	TIMER idle	▼	Output 17 -----{ }-----
		Direct	▲	
		Combinational		
		TIMER in operation		
		TIMER idle	▼	
		TIMER in operation/idle		
		TIMER with 2 values, in operatic		Output 17 Aux -----{ }-----

- **Direct**

By default there is no combinational logic and no function associated with the output channel. In this case, "Direct" is displayed in the cell of the main configuration screen. The output behaves like a standard output (%Qxy.i.17 directly controls output 17).

- **Combinational logic**

This enables the user to associate the output directly with the combinational logic.

- **Reflex function block**

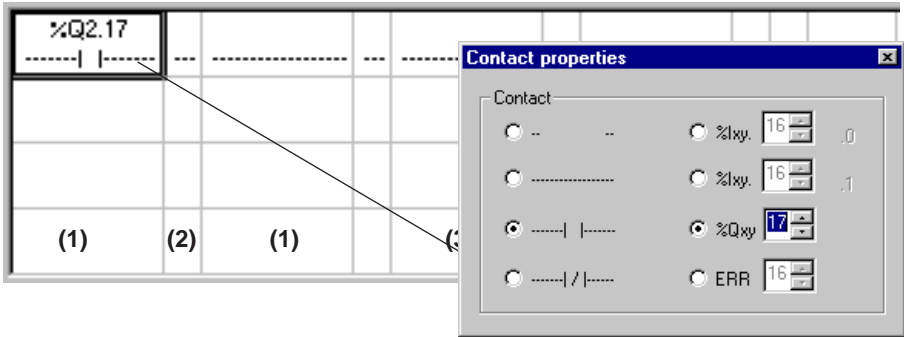
Select the function block from the list (example, off-delay TIMER).

The function block is displayed in column (5) of the grid.

To enter the parameters of the function block, change to Adjust mode (see section 5.5)

## Entering contacts

Click on the required cell in one of the columns (1).



The Properties dialog box is displayed, for selecting :

- a blank field
- a connection wire
- a normal contact
- an inverted contact

The Properties dialog box can also be used to define the associated language object (see section 5.3-1)

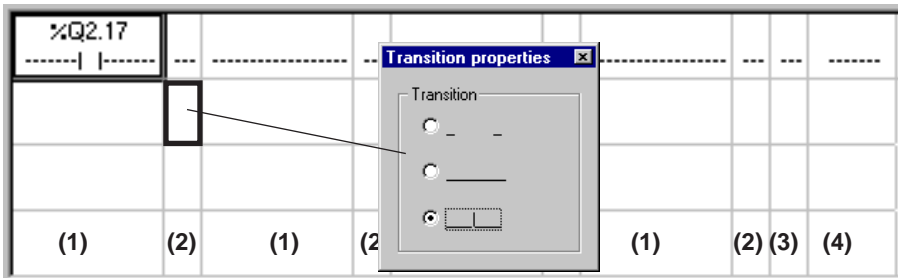
- %lx.y.i.0
- %lx.y.i.1
- %Qxy.i
- ERRi

1. Select the contact.
2. Select the variable.
3. Select the address.

## Entering links

To enter horizontal or vertical wires

1. Click on the required cell (column (2)).
2. Select the horizontal or vertical transition.

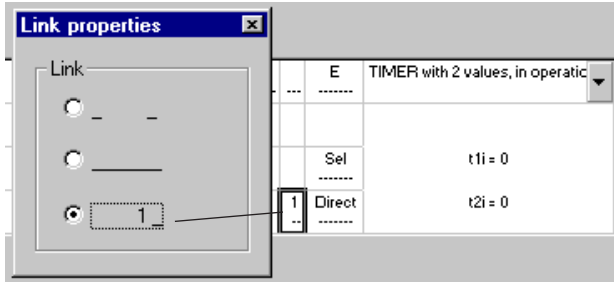




### Forcing function block inputs

Function block inputs can be forced to 1.

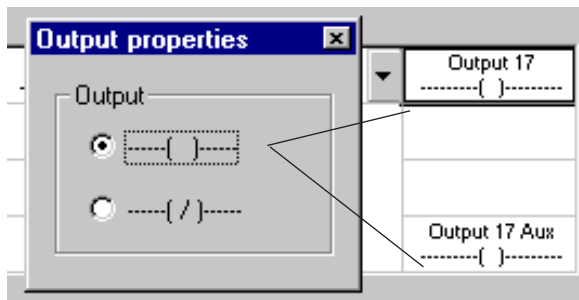
- 1 Click on the cell containing the input to be forced (column (3)).
2. Select the value 1.



(3)

### Entering output coils

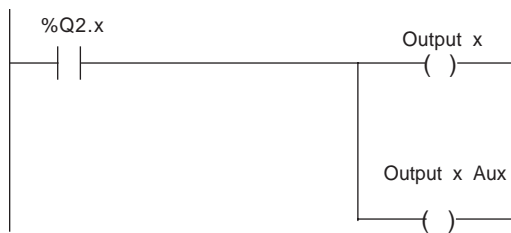
1. Click on the output cell.
2. Select :
  - a single coil
  - an inverse coil



## 5.4 Function blocks

### 5.4-1 Direct

Physical output x is directly controlled by command %Qxy.i from the PLC processor.



Note : This default function on an output channel is not a reflex function in that the command object is provided by the PLC processor.

### 5.4-2 Combinational function

#### Application

Performs a reflex combinational function on an output (without using any specific function block).

#### Operation

-----	Combinational logic	Output x ----- ( ) -----
		Output x Aux ----- ( ) -----

The combinational logic equation entered is applied directly to outputs **x** and **x Aux**.

Note : This function can be used when the combinational logic of a function is too unwieldy. Using the associated status bit %lxy.i as an intermediate result the combinational logic can be programmed on a number of "Ladders".

**5.4-3 On-delay timer function block**

**Application**

Applies a delay to the triggering of an action.

**Operation**

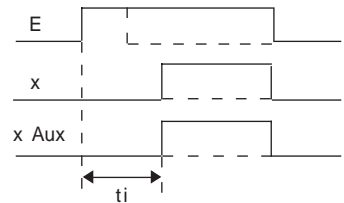
Launches a time delay **ti** (time base of 0.1 ms) on a rising edge of input **E**.

Output **x** changes to 1 at the end of the time delay.

If the duration of the high state of input **E** is less than time delay **ti**, output **x** remains at 0.

Output **x** changes to 0 when input **E** changes to 0.

E ----- -----	On-delay TIMER	Output x ----- ( ) -----
	<b>ti = 100</b>	
		Output x Aux ----- ( ) -----



Note : The operation of output **x Aux** is identical to that of output **x**.

**5.4-4 Off-delay TIMER function block**

**Application**

Applies a delay to the stopping of an action.

**Operation**

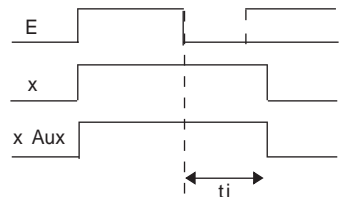
Launches a time delay **ti** (time base of 0.1 ms) on a falling edge of input **E**.

Output **x** is set to 1 when input **E** changes to 1.

Output **x** changes to 0 at the end of the time delay.

If the duration of the low state of input **E** is less than time delay **ti**, output **x** remains at 1.

E ----- -----	Off-delay TIMER	Output x ----- ( ) -----
	<b>ti = 100</b>	
		Output x Aux ----- ( ) -----



Note : The operation of output **x Aux** is identical to that of output **x**.

### 5.4-5 On-delay / off-delay TIMER function block

#### Application

Applies a delay to the starting and stopping of an action.

#### Operation

A time delay **tri** (delaying the start of an action) is launched on a rising edge of input **E**. Output **x** rises at the end of time delay **tri** (time base of 0.1 ms).

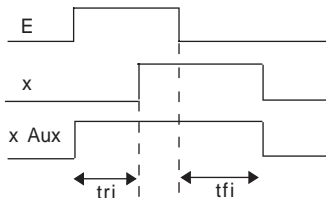
A time delay **tfi** (delaying the end of an action) is launched on a falling edge of input **E**. Output **x** falls at the end of time delay **tfi**.

If the duration of the high state of input **E** is less than **tri**, output **x** will not rise.

During time delay **tfi**, if input **E** changes to 0 for a period shorter than **tfi**, output **x** remains at 1.

Output **x Aux** remains at 1 as long as input **E** or output **x** is at 1.

E ----- -----	On-delay / off-delay TIMER	Output x ----- ( ) -----
	<b>tri</b> = 100	
	<b>tfi</b> = 150	Output x Aux ----- ( ) -----



### 5.4-6 2-value on-delay TIMER function block

#### Application

Applies a delay (**t1i** or **t2i**) to the starting of an action.

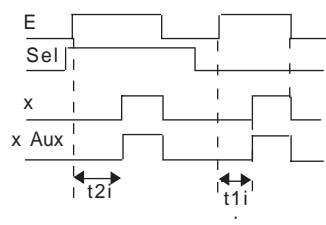
#### Operation

A time delay (time base of 0.1 ms) which corresponds to the state of the **Sel** input (selection of time delay **t1i** or **t2i**) is launched on a rising edge of input **E** :

- if **Sel** = 0 time delay **t1i** is selected
- if **Sel** = 1 time delay **t2i** is selected

If the duration of the high state of input **E** is less than the time delay selected, output **x** does not rise.

E ----- -----	2-value on-delay TIMER	Output x ----- ( ) -----
	<b>t1i</b> = 100	
<b>Sel</b> ----- -----	<b>t2i</b> = 120	Output x Aux ----- ( ) -----
<b>Direct</b> ----- -----		



Note : The operation of output **x Aux** is identical to that of output **x**.

**Increasing the number of time delays which can be selected (direct input)**

To increase the number of time delays which can be selected to more than two, several blocks must be linked so that output **x** of the first becomes input **E** of the next.

The **Direct** input can be used to select the block used.

When the Direct input is at 1, output **x** exactly follows input **E**.

To select a time delay :

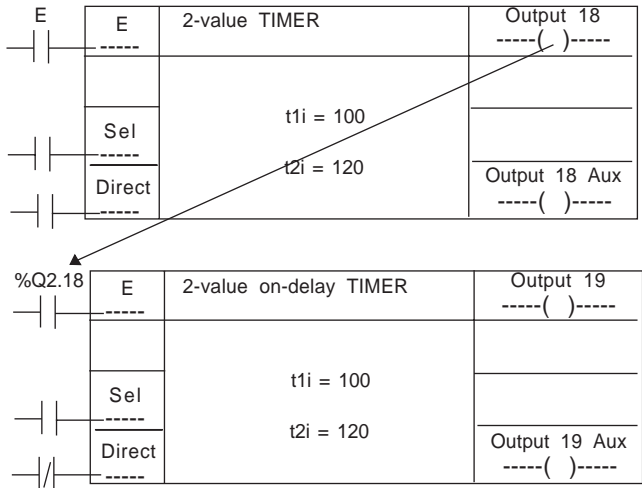
- the Direct input of the block containing the selected time delay must be at 0
- all other blocks must have their Direct input at 1

Output **x** will begin to fall again immediately with input **E**.

**Important**

When several function blocks of this type are linked, the states of the **Sel** and **Direct** inputs must only be changed when input **E** is at state 0.

Example : linking of two 2-value on-delay timer function blocks



**Notes :**

It is possible to use virtual outputs for intermediate blocks.

## 5.4-7 Selectable on-delay / off-delay TIMER function block

### Application

Used to independently select on-delay/off-delay using a **Sel** input.

The on-delay/off-delay function block offers a choice of two delays which can be applied on a rising and falling edge of output Q.

### Operation

Time delays **t1i** or **t2i** (time base 0.1 ms) can be selected using the **Sel** input or a zero time delay can be selected using the **Direct** input :

- on a rising edge of input **E** for an on-delay on **x**
- on a falling edge of input **E** for an off-delay on **x**

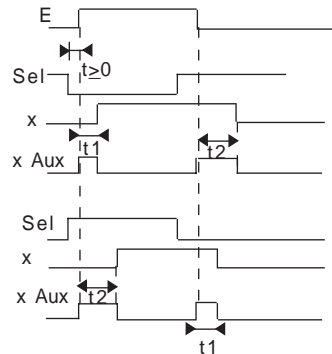
If **Sel** = 0, time delay **t1i** is selected. If **Sel** = 1, time delay **t2i** is selected.

If the duration of the high state of input **E** is less than the selected time delay, output **x** will not rise.

If the duration of the low state of input **E** is less than the selected time delay, output **x** will not fall.

Output **x Aux** is at 1 while the time delays are running.

E -----	Selectable on-delay/off-delay TIMER	Output Q ----- ( ) -----
Sel -----	t1i = 50	
Direct -----	t2i = 80	Output x Aux ----- ( ) -----



### Increasing the number of time delays which can be selected (Direct input)

See section 5.4-6.

**5.4-8 Retriggerable Monostable function block**

**Application**

Launches an action of duration **ti** with the possibility of restarting it at any time for the same duration.

**Operation**

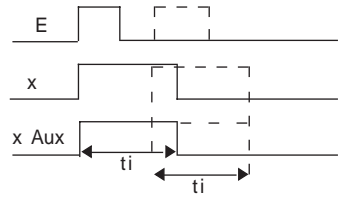
On a rising edge of input **E**, output **x** changes to 1 for time **ti** (time base of 0.1 ms).

When time **ti** has elapsed output **x** changes to 0.

The high state of output **x** is extended by the duration **ti** if a new rising edge occurs on input **E** during the time delay.

Output **x Aux** behaves in the same way as output **x**.

E -----	Retriggerable MONO	Output x ----- ( ) -----
	ti = 100	
		Output x Aux ----- ( ) -----



### 5.4-9 Time-delayed monostable function block

#### Application

Applies a delay to a monostable.

#### Operation

Time delay  $t1i$  (time base 0.1 ms) is launched on a rising edge of input **E**.  
 If the duration of the high state of input **E** (following a rising edge) is less than time delay  $t1i$ , output **x** will not rise.

The monostable time delay  $t2i$  is launched after the time delay  $t1i$  has elapsed and output **x** rises.

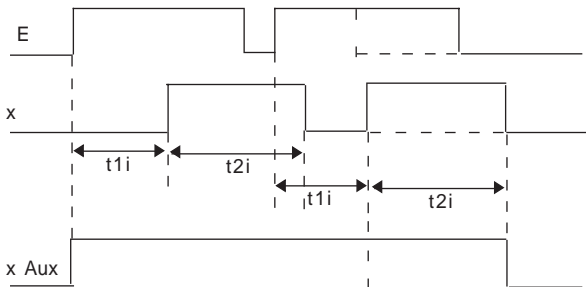
Once time delay  $t2i$  has elapsed, output **x** changes to low state.

If there is a rising edge of input **E** during the high state of output **x**, operation is identical to that described previously.

In all cases, the duration of the monostable is  $t2i$ .

Output **x Aux** is at 1 as long as input **E** or output **x** is at 1.

E	Time-delayed MONO	Output x ----- ( ) -----
	$t1i = 80$	
	$t2i = 120$	Output x Aux ----- ( ) -----





**5.4-10 2-value monostable function block**

**Application**

Launches a monostable of duration **t1i** or **t2i** (time base 0.1 ms), selected via the state of the **Sel** input.

The **Direct** input is used to perform a logic OR on the various monostable blocks of this type.

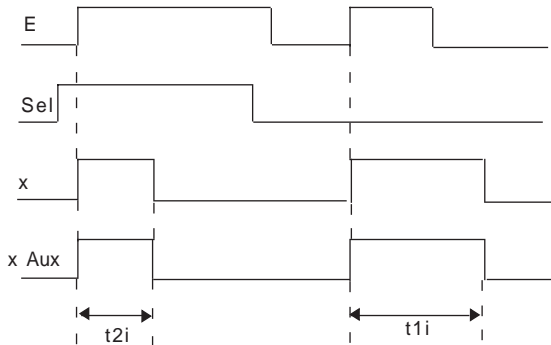
**Operation**

A time delay which corresponds to the state of the **Sel** input is launched on a rising edge of input **E**. Output **x** rises immediately.

- if the **Sel** input =0, time delay **t2i** is selected
- if the **Sel** input =1, time delay **t1i** is selected

In all cases the duration of the monostable is **t1i** or **t2i**.

E -----	2-value MONO	Output x ----- ( ) -----
Sel -----	t1i = 80	
Direct -----	t2i = 120	Output x Aux ----- ( ) -----



Note : The operation of output x Aux is identical to that of output x.

**Increasing the number of time delays which can be selected (Direct input)**

See section 5.4-6.

## 5.4-11 OSCILLATOR function block

### Application

Provides an oscillator for which the duration of the signal at state 0 and at state 1 can be independently defined.

Creates a time base for counter modules or sequences event-triggered tasks.

### Operation

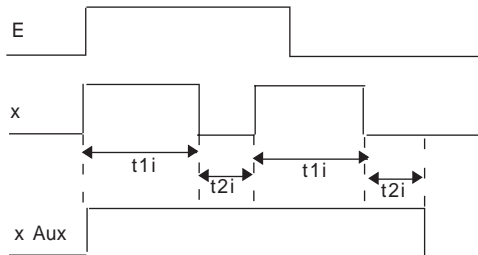
Output **x** changes to high state on a rising edge of input **E**.

The high state has a duration of **t1i** and the low state **t2i** (time base 0.1 ms).

A falling edge on input **E** during time **t1i + t2i** will stop oscillation at the end of that period.

Output **x Aux** is at 1 during oscillation.

E -----	OSCILLATOR	Output x ----- ( ) -----
	$t1i = 100$	
	$t2i = 20$	Output x Aux ----- ( ) -----



**5.4-12 COUNTER OUTPUT D function block**

**Application**

Creates sequential logic functions : edge latching, etc.

**Operation**

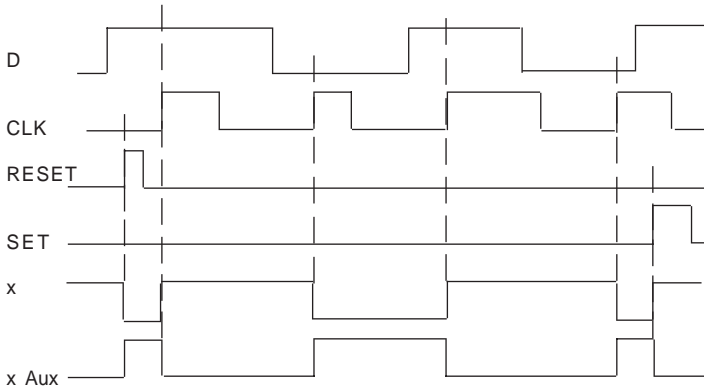
On a rising edge of input **CLK** output **x** takes the state of input **D** (output **x Aux** takes the inverse value).

Input **SET** at 1 sets output **x** to 1 and output **x Aux** to 0.

Input **RESET** at 1 sets output **x** to 0 and output **x Aux** to 1.

Input **RESET** takes priority over input **SET**.

D -----	COUNTER OUTPUT D	Output x ----- ( ) -----
CLK -----		
SET -----		
RESET -----		Output x Aux ----- ( ) -----



## 5.4-13 COUNTER OUTPUT T function block

### Application

Creates a divisor by 2 or other complex sequential functions.

### Operation

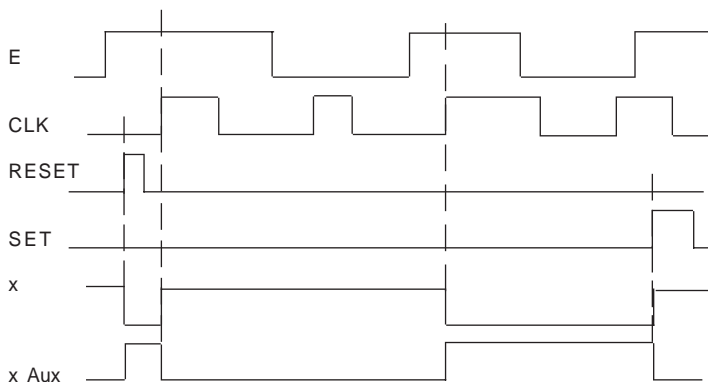
On a rising edge of input **CLK**, if input **E** is at 1, output **x** takes the inverse of its current state (output **x Aux** takes the inverse value of **x**).

On a rising edge of input **CLK**, if input **E** is at 0, outputs **x** and **x Aux** remain in their current state.

Input **SET** at 1 sets output **x** to 1 and output **x Aux** to 0.

Input **RESET** at 1 sets output **x** to 0 and output **x Aux** to 1.

E -----	COUNTER OUTPUT T	Output x ----- ( ) -----
CLK -----		
SET -----		
RESET -----		Output x Aux ----- ( ) -----



**5.4-14 Double threshold COUNTER function block**

**Application**

By counting the edges on an input, this function triggers an output when the threshold selected from the two available thresholds is reached.

Comment : The maximum performance level of the counter is 2 KHz (with input Up controlled directly by the physical input (with no filtering)).

**Operation**

On a rising edge of the **Reset** input :

- output **x** changes to low state
- the counter is reset to 0
- the threshold is selected using the **Sel** input.

If the **Sel** input = 0, threshold **th1** is selected.

If the **Sel** input = 1, threshold **th2** is selected.

The counter is incremented on a rising edge of the **Up** input.

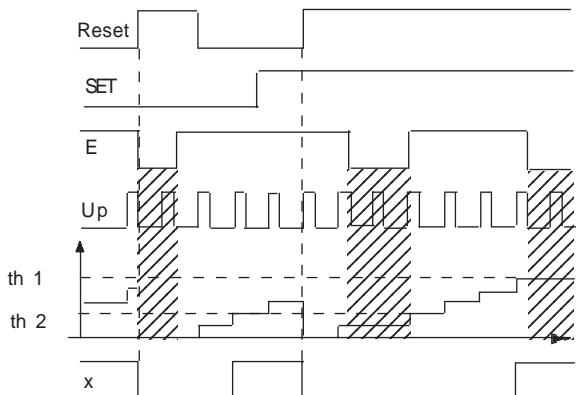
Input **E** at 0 freezes counting.

When the threshold is reached, output **x** changes to high state.

**Important**

Initializing a threshold at zero corresponds to the maximum number of points (ie. 65536 points). A reset is required to define the value of the threshold to be reached.

E ----- -----	Double threshold COUNTER	Output x ----- ( ) -----
Up ----- -----	th1 = 120	
Reset ----- -----		
Sel ----- -----	th 2 = 60	Output x Aux ----- ( ) -----



Note : The operation of output x Aux is identical to that of output x.

## 5.4-15 Single electronic cam function block

### Application

Counting the rising edges on a counter input (**Up**) activates an output when the number of pulses is between 2 threshold values.

Application example, cam in position : perform an action when the part is between two predefined positions.

### Operation

A rising edge on input **Reset 1** results in the resetting to zero of the counter and output **x Aux** changing to 1. If input **Reset 0** is at 1, output **x** is forced to 0.

The counter is incremented on a rising edge of input **Up**.

Input **E** at 1 enables counting, the rising edges of the signal on input **Up** are taken into account and the counter is incremented.

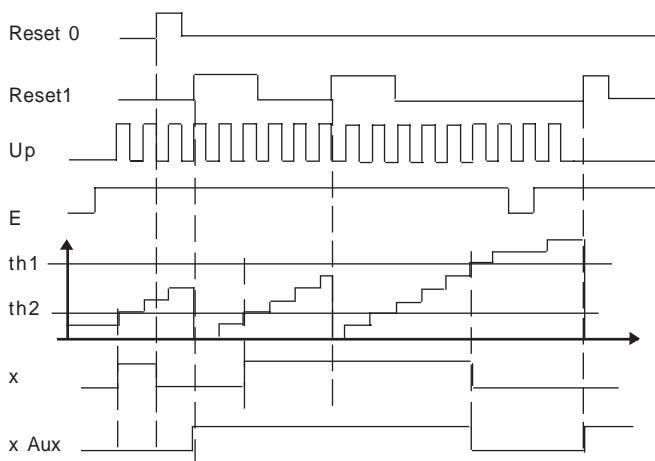
When threshold **th1** is reached, output **x** changes to 1. It remains at 1 until the counter reaches threshold **th2**.

Output **Aux** changes to 0 when threshold 2 is crossed and changes to 1 on a rising edge of input **Reset 1**.

### Important

If the counter is not reset to 0, when it reaches  $2^{16}+1$  (modulo +1), it will return to 0,1,2 etc. In many applications, it is advisable to inhibit counting (**E**=0) using output **x Aux** in series with input **E**.

E ----- -----	Single electronic CAM	Output x ----- ( ) -----
Up ----- -----	th1 = 60  th2=120	
Reset 0 ----- -----		
Reset 1 ----- -----		Output x Aux ----- ( ) -----



**5.4-16 INTERVAL COUNTER function block**

**Application**

The maximum counting level is 6.5 s with a precision of 0.1 ms. This function can be used to reach higher rates by changing the time base.

**Operation**

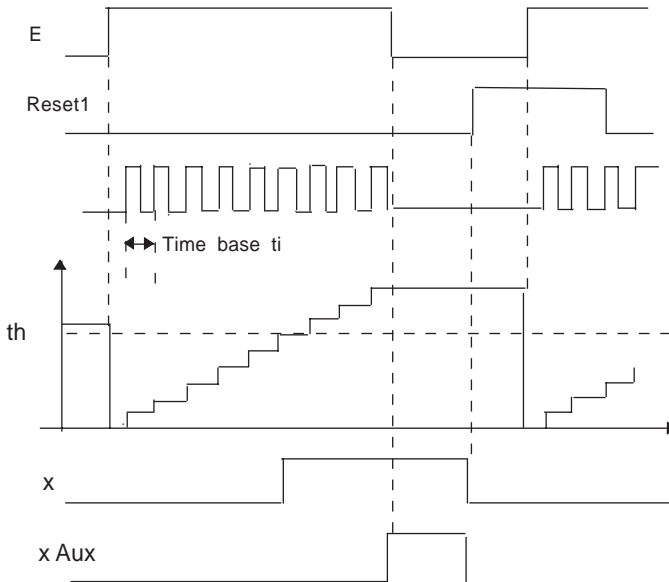
A rising edge on input **E** sets the outputs to 0 and resets the counter to 0.

The counter increments on each time base period **ti** until it reaches threshold **th**. Output **x** then changes to high state.

A rising edge on input **Reset1** resets the outputs to 0.

If input **E** changes to low state while output **x** is in high state, output **x Aux** will rise.

E -----	Single threshold INTERVAL COUNTER	Output x ----- ( ) -----
	$t_i = 20$	
Reset1 -----	$t_h = 120$	Output x Aux ----- ( ) -----



## 5.4-17 BURST function block

### Application

Generates a defined number of oscillator periods.

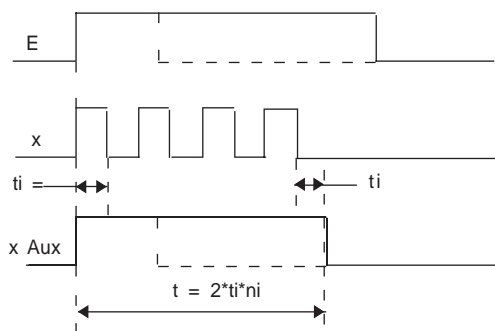
### Operation

On a rising edge of **E**, "Burst" is launched : **ni** impulses of duration  $2 \times t_i$  are generated. If input **E** falls before the end of the Burst, oscillation will stop at the low state of output **x**.

Output **x Aux** is a control output which changes the Burst time to high state.

Output **x Aux** changes to 0 either at the end of oscillation periods **ni**, or if input **E** changes to 0.

E -----	BURST	Output x ----- ( ) -----
	$t_i = 20$	
	$n_i = 4$	Output x Aux ----- ( ) -----





**5.4-18 PWM function block**

**Application**

Generates a continuous oscillation with a fixed frequency but a variable cyclic ratio.

**Operation**

On a rising edge of input **E**, launches oscillation on a high state of output **x**.

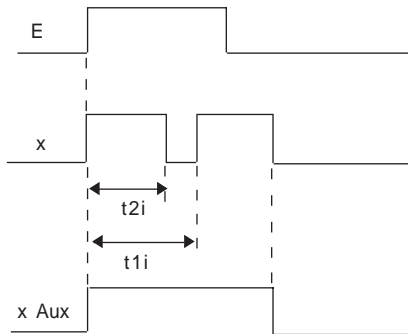
**t1i** is the signal duration, **t2i** is the duration of the high state.

Output **x Aux** is a control output which falls back at the end of oscillation.

Oscillation ends when input **E** and output **x** are in a low state.

If **t2i** ≥ **t1i** output **x** will remain in a high state.

E -----	PWM generation	Output x ----- ( ) -----
	t1i = 100	
	t2i = 80	
		Output x Aux ----- ( ) -----



## 5.4-19 Slow speed detection function block

### Application

Stops the movement of a moving part if its speed falls below a certain limit (for example : overload on a conveyor belt).

### Operation:

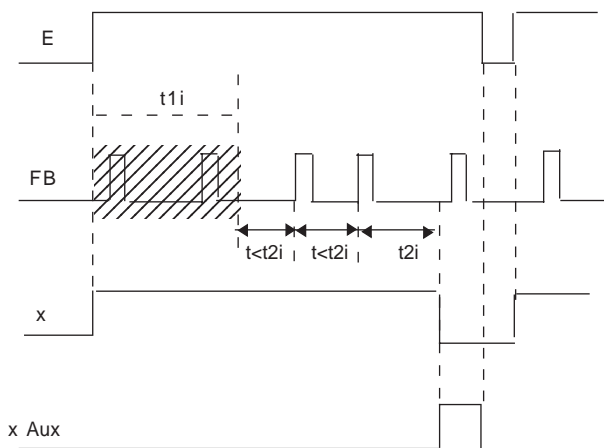
Time delay  $t1i$ , during which the state of output  $x$  is at 1, is launched on a rising edge of input  $E$  regardless of the period of the  $FB$  input signal. This allows the machine to start (masking time).

Time delay  $t2i$  is launched when time delay  $t1i$  has elapsed and then on each rising edge of the  $FB$  input.

If the rising edges of the  $FB$  input are at intervals greater than  $t2i$ , output  $x$  changes to 0 and output  $x$   $Aux$  changes to 1 (signaling that movement has stopped).

Input  $E$  changing to 0 will stop output  $x$  at any time.

E -----	Slow Speed Detection 1	Output x ----- ( ) -----
	$t1i = 0$	
	$t2i = 0$	
Speed to be controlled FB -----		Output x Aux ----- ( ) -----



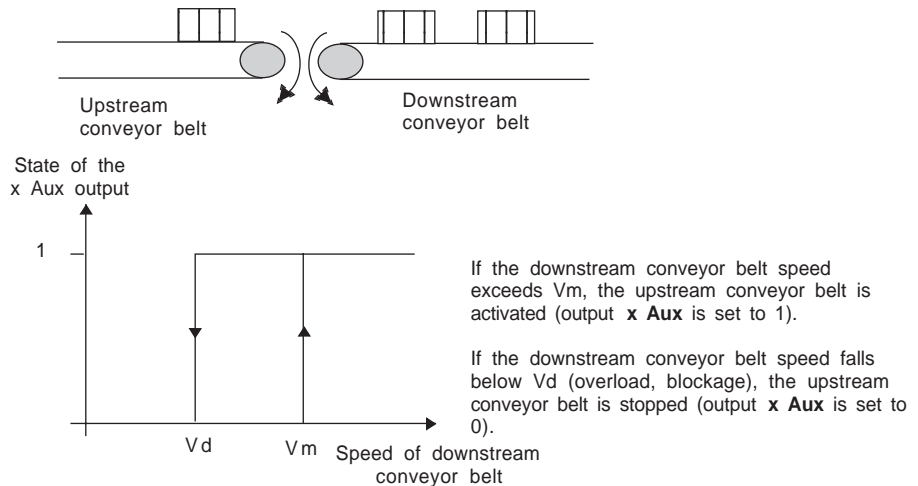
### 5.4-20 Speed monitoring function block

#### Application

Controls the movement of a moving part (sets output **x Aux** to 1) if the speed is greater than an upper limit.

Controls the stopping of a moving part (sets output **x Aux** to 0) if the speed is less than a lower limit.

Typical application : stopping an upstream conveyor belt when a downstream conveyor belt is overloaded.



#### Operation

On a rising edge of input **E**, launches time delay **t1i** and controls output **x**.

When the period of the pulses (rising edges) arriving at input **FB** is less than **t1i**, output **x Aux** changes to 1 (which corresponds to a downstream conveyor belt speed greater than  $V_m$ ).

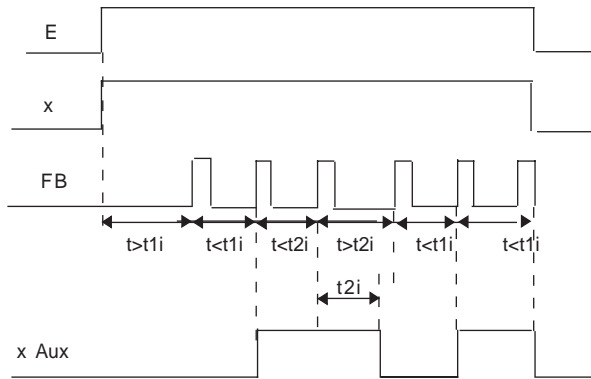
Then, if the period of the pulses arriving at input **FB** becomes greater than **t2i**, output **x Aux** returns to 0 (which corresponds to a downstream conveyor belt speed less than  $V_d$ ).

If **E** changes to low state, outputs **x** and **x Aux** will fall.

#### Important

The system will only operate correctly if the fall time is greater than the rise time **t2i > t1i** (see the diagram on the next page)

E -----	Slow Speed Detection 2	Output x ----- ( ) -----
	$t1i = 60$	
Speed to be controlled	$t2i = 80$	Output x Aux ----- ( ) -----
FB -----		



**5.4-21 Type 1 Command/Control function block**

**Application**

Controls an action and checks to see if it has been executed correctly after a certain period of time.

**Operation**

Output **x** exactly follows the **Cmd** input.

Launches the time delay **ti**, which sets the time for a check, on a rising edge of the **Cmd** input.

1. At the end of time delay **ti**, if the **Ctrl** signal has not arrived, output **x Aux** changes to 1 (type A error signal).

2. If the **Ctrl** input falls back to 0 (while **Cmd** is at 1), output **x Aux** changes to 1 (type B error signal).

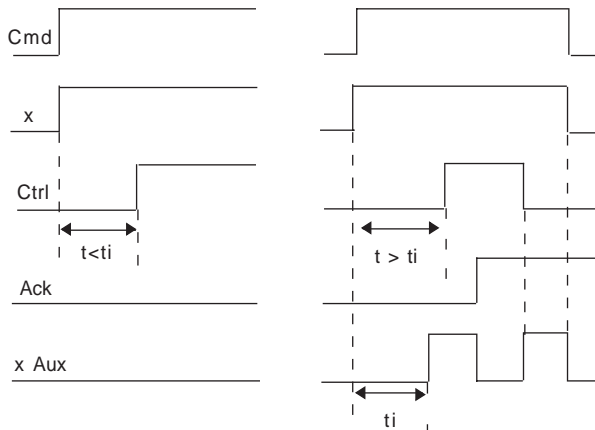
After a type A error, when the **Ack** input rises, output **x Aux** will fall. Type B errors can now be displayed.

If the **Cmd** input changes to 0, outputs **x** and **x Aux** will return to 0.

**Important**

The acknowledgment request **Ack** will only be taken into account if the **Ctrl** input is at 1.

Cmd -----	Type 1 Command/Control	Output x ----- ( ) -----
Ack -----	$t_i = 100$	
Ctrl -----		Output x Aux ----- ( ) -----



## 5.4-22 Type 2 Command/Control function block

### Application

Controls an action (for example controlling a cylinder) and checks to see if it has been executed correctly.

Stops the action and checks that it has been stopped correctly.

### Operation:

Output **x** exactly follows the **Cmd** input.

- On a rising edge of the **Cmd** input : launches time delay **t1i** which sets the time for the check.
- On a falling edge of the **Cmd** input : launches time delay **t2i** which sets the time for the check.

After this, operation, which is identical for both the rising and falling edges of the **Cmd** input, is as follows :

1. At the end of the time delay, if the **Ctrl** signal has not arrived, output **x Aux** changes to 1 (type A error signal)
2. If the **Ctrl** input falls back to 0, output **x Aux** changes to 1 (type B error signal).

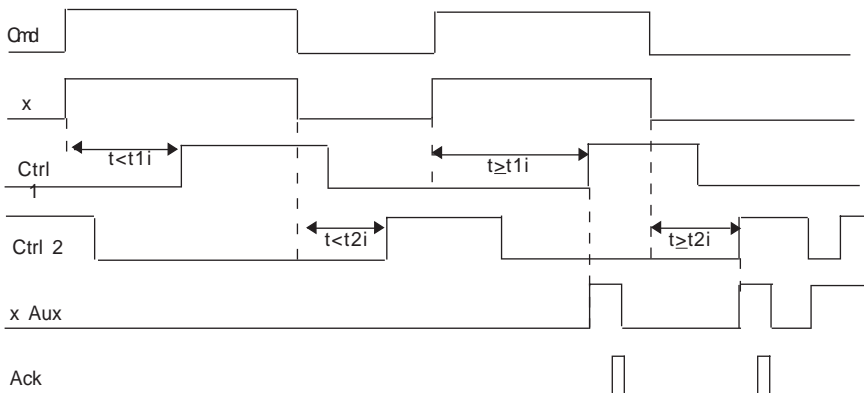
After a type A error, if the **Ack** input changes to 1, output **x Aux** will fall back. Type B errors can now be displayed.

An edge on the **Cmd** input will cause output **x Aux** to fall.

### Important

The acknowledgment request **Ack** is only taken into account if the **Ctrl** input is at 1.

Cmd -----	Type 2 Command/Control	Output x ----- ( ) -----
Ack -----	$t1i = 60$	
Ctrl 1 -----		
Ctrl 2 -----	$t2i = 40$	Output x Aux ----- ( ) -----



**5.4-23 Control/Counting function block**

**Application**

Controls a positioning action.

**Operation:**

A rising edge on the **Reset** input resets the counter to 0.

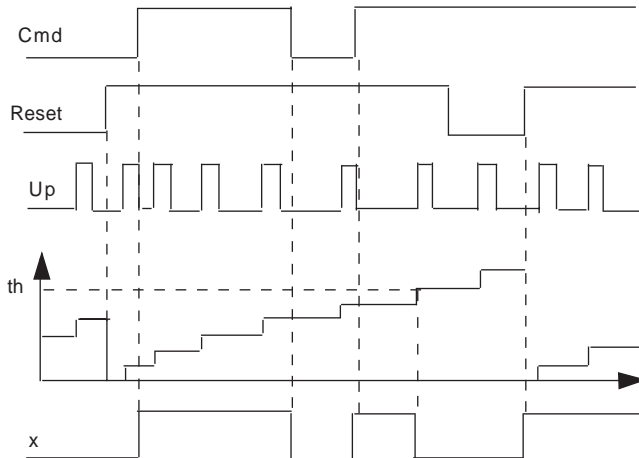
Output **x** is activated if :

- the **Cmd** input is in the high state
- the counter has not reached the threshold

The **Cmd** input does not influence counting which is performed on a rising edge of the **Up** input.

When the threshold is reached, output **x** changes to 0.

Cmd -----	Control/Counting	Output x ----- ( ) -----
	th = 100	
Reset -----		
Up -----		Output x Aux ----- ( ) -----



Note : The operation of output x Aux is identical to that of output x.

### 5.4-24 Fault indication function block

#### Application

Signals a fault with acknowledgment and clearing (2 flashing frequencies).

#### Operation :

Output **x**, which is controlled by the **Err** input, flashes if a fault occurs :

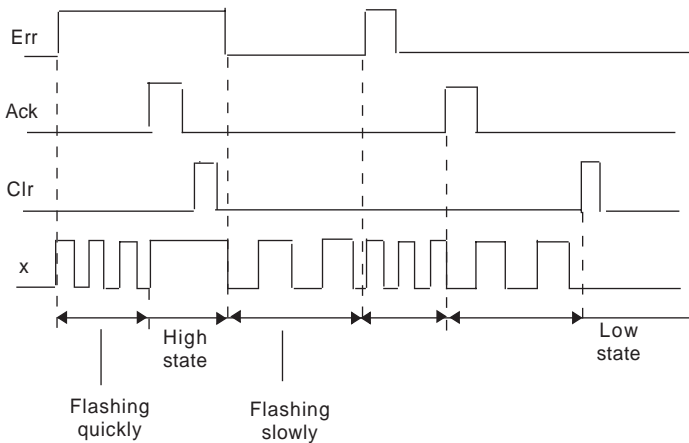
- flashing quickly : error is present and has not been acknowledged by the **Ack** input
- flashing slowly : error has been acknowledged by the **Ack** input and is no longer present
- on : error is present and has been acknowledged by the **Ack** input
- off : last error cleared using input **Clr** after acknowledgment

**t1i** : half of the quick flashing period

**t2i** : half of the slow flashing period

Note : For there to be quick and slow flashing, **t1i** must be less than **t2i**.

Err -----	Fault indication function	Output x ----- ( ) -----
	$t1i = 20$	
Ack -----		
Clr -----	$t2i = 120$	Output x Aux ----- ( ) -----



Note : This function block has no x Aux output.



## 5.5 Configuring the function blocks

### 5.5-1 Adjustment via the terminal in offline mode

To configure the parameters of the function blocks, change to Adjust mode. Adjust mode can only be accessed offline. This mode is used for entering initial parameters.

Chan.	Function block	Input 1	TM value 1	Input 2	TM value 2
16	TIMER with 2 values, in operation	t1i	0	t2i	0
17	TIMER idle	ti	0		
18	Direct				
19	Direct				
20	TIMER with 2 values, in operation	t1i	0	t2i	0
21	Direct				
22	Direct				
23	Direct				
24	Direct				
25	Direct				
26	TIMER idle	ti	0		
27	Direct				
28 v	Fault signalling	t1i	0	t2i	0
29 v	Direct				
30 v	T flip-flop				
31 v	Direct				

1. Select the cell relating to the parameter to be entered.
2. Enter the parameter.
3. Confirm with **Enter**.

The value of the parameters is between 0 and 65535.

---

## 5.5-2 Modifying the parameters via the program

- **Global modification for groups of 8 channels ( 16 to 23 or 24 to 31 )**

The **WRITE\_PARAM** (Write parameters) instruction transmits all the parameters associated with the 8 channels in a group at once.

Before transmission, the «Value 1» and/or «Value 2» parameters must be modified for each channel.

( %MW.xy.i.4 := «value»: Parameter «Value 1» of function block )

( %MW.xy.i.5 := «value»: Parameter «Value 2» of function block )

Syntax :           **WRITE\_PARAM%CHxy.i**  
                           (where i = 16 or 24)

- **Modification channel by channel**

The **MOD-PARAM** (Parameter Modification) instruction is used to modify all the parameters associated with a single channel.

Syntax : **MOD\_PARAM%CHxy.i (no., value1 , value2 , 0)**  
                           (where i = 16 or 24 and no. = index of channel in the group of 8)

Example

Modification of channel 18 parameters (value 1=10 ms, value 2=500 ms)

**MOD\_PARAM%CHxy.16 (2, 100, 5000, 0)**

The **SAVE\_PARAM**, **RESTAURE\_PARAM** and **READ\_PARAM** instructions are standard : refer to the Application-Specific Functions manual.

---

## 5.5-3 Explicit exchange language objects

- **Standard objects**

%MWxy.i.0                   Exchange in progress status

%MWxy.i.1                   Exchange report

%MWxy.i.2                   Channel status

%MWxy.i.3                   Channel command

%MWxy.MOD.2               Module status

See section 4.3-2.

- **Specific objects for the DMY28RFK module**

%MW.xy.i.4 (i = 16 to 31) : Parameter «Value 1» of function block

%MW.xy.i.5 (i = 16 to 31) : Parameter «Value2» of function block

where **xy** : position of the module (x = rack no. and y = slot no.)

**i** : channel number

## 5.6 Debugging

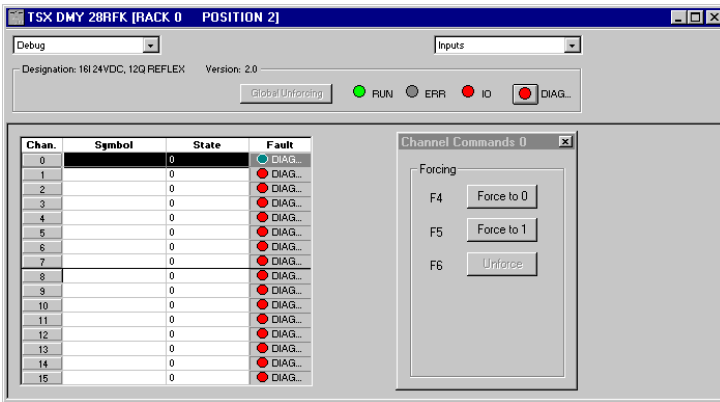
### 5.6-1 Accessing debugging

1. Change to online mode.
2. Access the hardware configuration from the Application Browser.
3. Double-click on the module concerned.

See section 3 for more information on the debug functions.

#### Input debug screen

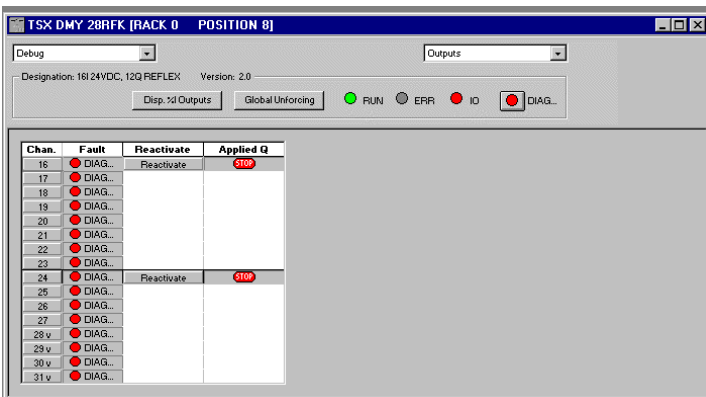
- Displays detailed diagnostics for a channel : see section 3.5-1.
- Forcing or unforcing channels : see section 3.5-2.



#### Output debug screen

By default, the user can access the following diagnostic functions :

- Displays detailed diagnostics for a channel : see section 3.5-1.
- Output reactivation command : see section 3.5-4.
- Applied outputs : see section 3.5-5.



The "**Display State**" button can be used to display the state of the variables (channel control, physical outputs, auxiliary outputs, etc) on each channel :

- **Real outputs of the module**

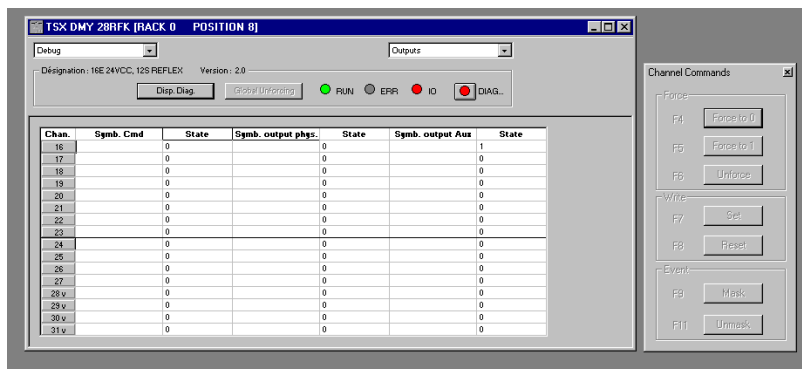
- %Q.xy.i** ( i = 16 to 27 ) : Output command bit
- %l.xy.i** ( i = 16 to 27 ) : State of the physical outputs
- %l.xy.i.1** ( i = 16 to 27 ) : State of the auxiliary outputs

- **Virtual outputs of the module**

- %Q.xy.i** ( i = 28 to 31 ) : Command bit
- %l.xy.i** ( i = 28 to 31 ) : State of the function block outputs
- %l.xy.i.1** ( i = 28 to 31 ) : State of the auxiliary outputs

where :

- xy** : position of the module (x = rack no. and y = slot no.)
- i** : channel number



This screen is also used to access :

- Masking or unmasking of events : see section 3.5-3.
- Forcing or unforcing of the output command bits **%Qxy.i** : see section 3.5-2.

Note : A physical output cannot be forced unless it is used as a standard discrete output (ie. controlled by the object %Qxy.i: direct function).

The "**Display Diag**" button is used to return to the general debug screen for the module.

---

## 5.7 Diagnostics

---

If a module is faulty, some of the indicator lamps which can be accessed in the configuration editor screen turn red :

- indicator lamp showing the position of the module on the screen in which it is displayed (first screen in the configuration editor)
- copy of the ERR and I/O module indicator lamps, in the module level zone
- indicator lamp integrated in the DIAG command button, also in the module level zone

Activating the **DIAG** command button also accesses the **Module Diagnostics** screen which shows the current module faults, classified according to their category : internal faults, external faults or other faults.

### List of module faults

- **Internal faults** : Module failure
- **Other faults** : Faulty channel(s) (details in the channel diagnostics).  
Terminal block fault, self-test in progress, configuration fault, module missing or off.



## Symboles

- 2-value monostable function block 5/21  
 2-value on-delay TIMER function block 5/16

## A

- Accessing the configuration editor 1/2  
 Addressing 4/1  
 Application Browser 1/2

## C

- Channel diagnostics 3/6  
 Channel faults 3/6  
 Choosing the modules 1/3  
 Command/Control function block 5/33, 5/34  
 Configuration screen 2/2  
 Configuring the function blocks 5/37  
 Configuring the reflex functions 5/7  
 Confirming the configuration 2/20  
 Connect 3/1  
 Control/Counting function block 5/35  
 Copy/paste 2/3

## D

- Debug screen 3/2  
 Debugging 3/1, 5/39  
 Deconfiguration 2/18  
 Diagnostics 5/41  
 Displaying parameters 2/2  
 Distributed discrete 1/6  
 Distributed discrete addressing 4/2

## E

- Electronic cam function block 5/26  
 Entering the input parameters 5/4  
 Entering the output parameters 5/5  
 Event 2/8, 3/8  
 Event output 5/6  
 Explicit exchange 4/4  
 External power supply fault 2/9, 2/17

## F

- Fallback 2/15  
 Fallback mode 5/6  
 Fast inputs 2/8  
 Fault indication function block 5/36

- Filtering 2/9, 5/4  
 FIPIO 1/6  
 FIPIO connection 1/6  
 Forcing 3/7  
 Functions 2/8

## G

- Global reconfiguration 2/20  
 Global unforcing 3/5

## I

- Implicit exchange 4/3  
 In-rack discrete 2/4, 2/11  
 In-rack discrete addressing 4/1  
 INTERVAL COUNTER function block 5/27

## L

- Language objects 4/3  
 Latching 2/8, 2/10

## M

- Masking 3/8  
 Modify the parameters 5/5  
 Modifying parameters 2/3, 2/7  
 Module diagnostics 3/4  
 Module faults 3/4, 5/41  
 MOMENTUM 1/7  
 Momentum 1/7, 2/6, 2/13  
 Monostable function block 5/19, 5/20, 5/21  
 Multiple selection 2/3

## O

- Objects associated with the reflex module 5/7  
 Off-delay TIMER function block 5/15  
 On-delay / off-delay TIMER function block 5/16  
 On-delay timer function block 5/15  
 OSCILLATOR function block 5/22  
 OTHER 1/7, 1/9  
 Output modules 2/11

---

## P

Parameter	5/3
Parameter setting	1/4, 1/8
PWM function block	5/29

## R

Reactivation	2/16, 3/9, 4/5
Reading the status word	4/6
Reconfiguration	2/18
Reflex function blocks	5/2
Reflex function Ladder editor	5/10
Retriggerable Monostable function block	5/19
RUN/STOP input	2/19

## S

Selectable on-delay / off-delay TIMER function block	5/18
Shortcut menus	2/3
Slow speed detection function block	5/30

## T

Task	2/7, 2/14
TBX	1/7, 2/5, 2/12
Time-delayed monostable function block	5/20
TSX DMY 28 RFK module	5/1
Type 1 Command/Control function block	5/33
Type 2 Command/Control function block	5/34

## U

Unforcing	3/7
Unmasking	3/8

## W

Wiring check	2/10, 2/17
Write command	3/9
Writing the command word	4/6



<b>Section</b>	<b>Page</b>
<b>1 General</b>	<b>1/1</b>
1.1 Introduction	1/1
1.2 Basic AS-i concepts	1/2
1.2-1 Structure of an AS-i slave	1/2
1.2-2 TSX SAY 100 module architecture	1/3
<b>2 Configuring the AS-i communication function</b>	<b>2/1</b>
2.1 Introduction	2/1
2.2 The configuration tool	2/1
2.2-1 Accessing the configuration tool	2/1
2.2-2 Selecting an AS-i communication module	2/2
2.3 Description of the module configuration screen	2/3
2.4 Selecting slave devices to be connected	2/4
2.4-1 Selection principle	2/4
2.4-2 Adding and modifying a profile in the catalog	2/7
2.5 Configuring slaves	2/8
2.6 General parameters	2/9
2.6-1 Automatic addressing of slaves option	2/9
2.6-2 Output fallback position for slave devices	2/10
2.7 Confirming the configuration	2/11
2.7-1 Confirming after modification	2/11
2.7-2 Global reconfiguration	2/11

<b>Section</b>	<b>Page</b>
<b>3 Debugging the AS-i communication function</b>	<b>3/1</b>
3.1 Introduction to the debug function	3/1
3.2 Description of the module debug screen	3/2
3.3 Diagnostics mode	3/3
3.4 Displaying the status of the slaves	3/4
3.5 Adjusting the parameters	3/5
<b>4 Bits and words associated with the AS-i application</b>	<b>4/1</b>
4.1 Addressing slave I/O on the AS-i bus	4/1
4.2 Language objects associated with the AS-i module	4/2
4.2-1 Configuration objects	4/3
4.2-2 AS-i channel status objects (implicit exchanges)	4/3
4.2-3 AS-i channel status objects (explicit exchanges)	4/4
4.2-4 Command objects of the AS-i communication channel	4/5
4.2-5 Adjustment objects of the AS-i communication channel	4/5
4.2-6 TSX SAY 100 module language objects	4/6
<b>5 AS-i operating mode</b>	<b>5/1</b>
5.1 General	5/1
5.2 AS-i protected mode	5/2
5.3 AS-i wiring test mode	5/2
5.4 Advanced operating modes	5/3

<b>Section</b>	<b>Page</b>
<b>6 Performance</b>	<b>6/1</b>
6.1 Performance	6/1
<b>7 Index</b>	<b>7/1</b>

---

**Section**

**Page**

---

## 1.1 Introduction

---

This part describes the software setup for the AS-i bus and the slave devices which are connected to it, on the TSX/PMX/PCX57 PLC.

Before creating an application program, the physical operating context in which it will be used must be defined, ie. the rack and the modules located in the rack : supply, processor and discrete I/O and application-specific modules (analog, communication, counting, etc).

The use of the AS-i bus means that AS-i slave devices connected to the bus must also be defined, as well as the I/O parameters of these devices.

To do this, PL7 software offers the **configuration** editor which makes it easy to perform these operations.

Once they have been configured, the I/O of the slave device can be accessed via the bit objects in the user program.

In online application operation, the **configuration** editor also offers debugging functions which can be used to :

- check the connection of devices on the bus,
- adjust certain parameters in order to best adapt them to the application,
- write or force I/O bits associated with slaves,
- detect module faults.

---

## 1.2 Basic AS-i concepts

---

The AS-i bus (Actuator Sensor Interface) is a sensor/actuator connection designed for the lowest level of automation.

The architecture design is very simple, with all the sensors/actuators on a single cable. Assembly and installation using vampire clips eliminates the risk of error, and facilitates modification.

31 slaves can be connected on one AS-i bus on a length of 100m, and each slave can manage 4 inputs and 4 outputs, making a total of 248 I/O.

---

### 1.2-1 Structure of an AS-i slave

An AS-i slave is equipped with an integrated circuit for connection to the AS-i bus. The integrated circuit comprises :

- 4 configurable data I/O,
- 4 parameter outputs.

The operating parameters, configuration data with I/O assignment, identification and address codes are stored in a non-volatile memory.

#### I/O data

The outputs are destined for control system components and are transmitted by the AS-i master to the AS-i slave. The values of the inputs are stored by the AS-i slave and made available to the master.

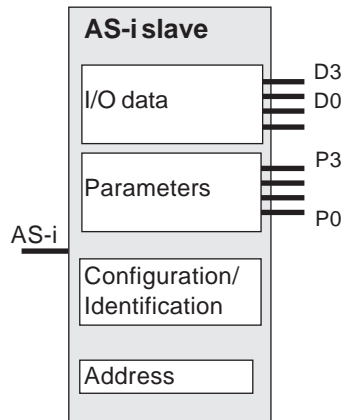
#### Parameters :

The AS-i slave parameter outputs enable the AS-i master to transmit those values which are not interpreted as I/O data.

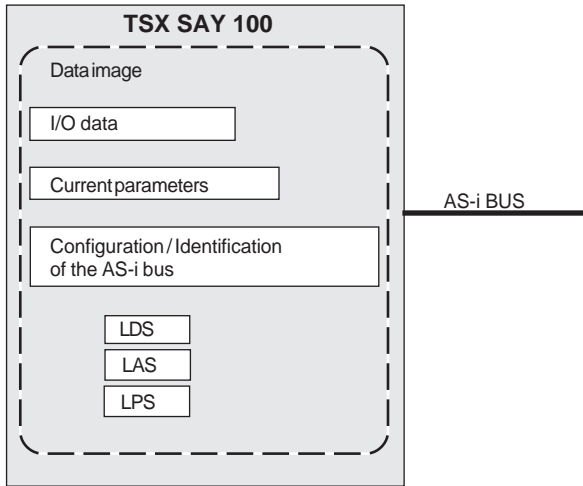
These parameters can be used for controlling and switching sensor or actuator internal modes.

For example : motor starter : rotation speed, starting, stopping, etc.

proximity sensors : sensing distance, remove background, etc.



1.2-2 TSX SAY 100 module architecture

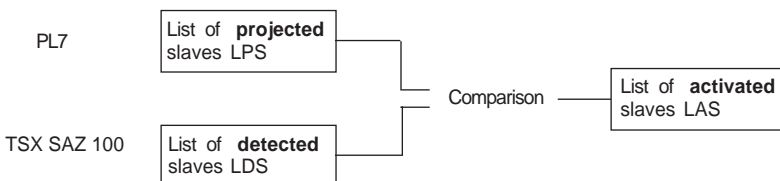


The module integrates the data fields which can be used to manage lists of slaves and the I/O data images. This information is stored in the volatile memory.

- I/O data : images of the 124 inputs and 124 outputs of the AS-i bus,
- Current parameters : image of the parameters for all the slaves,
- Configuration / identification : this field contains all the I/O codes and identification codes for all detected slaves,
- LDS : list of detected slaves on the bus,
- LAS : list of activated slaves on the bus.
- LPS : list of projected slaves on the bus configured by PL7. The master is responsible for checking the presence of slaves on the bus using this list.

**Data exchange is only possible with activated slaves.**

Operating diagram :



---

The TSX SAY 100 module operates in master/slave mode. The master alone controls exchanges on the bus. The AS-i standard defines several levels of service provided by the master.

Profile M0 : "Minimum Master" , the master offers configuration of slaves connected to the bus on power-up and I/O exchanges only.

Profile M1 : "Full Master", this profile covers all the functions defined by the AS-i standard.

Profile M2 : "Reduced Master", this profile has the functions of profile M0 with the possibility of defining the slave parameters.

<p>The TSX SAY 100 module corresponds to profile M2 with the added possibility of reading diagnostic information for slaves.</p>
--



## 2 Configuring the AS-i communication function

### 2.1 Introduction

Before creating an application program, it is necessary to define the physical operating context in which it will be executed, ie. the type of processor and the modules located in each slot.

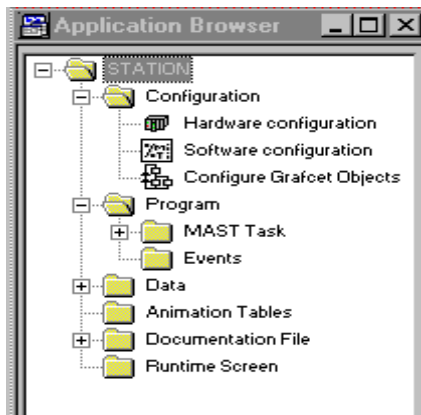
Using the AS-i communication function requires the user to define the parameters of the communication channel used and of each of the slaves.

In order to do this, the **PL7 Junior** and **PL7 Pro** software packages offer the configuration tool which simplifies these operations. In online operation, it also offers a debug screen which can be used to adjust certain parameters in order to make them more suitable for the application.

### 2.2 The configuration tool

#### 2.2-1 Accessing the configuration tool

Use the Application Browser to select the Station and then Configuration icons, then double-click on the "Hardware configuration" icon.



If the Application Browser is not displayed :

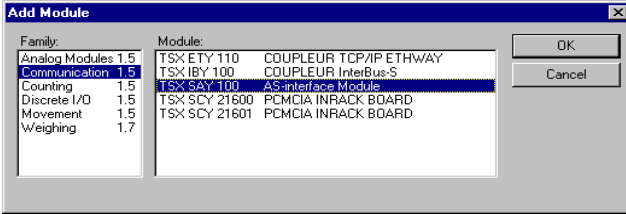
- click on the Application Browser icon



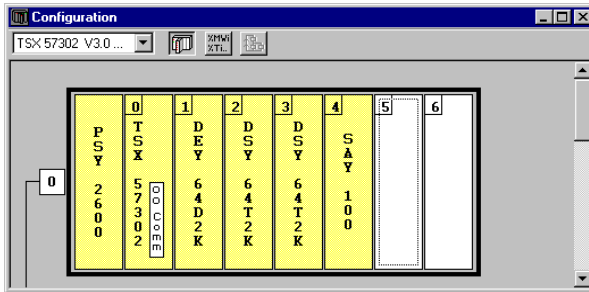
- or select **Tools/Application Browser**

## 2.2-2 Selecting an AS-i communication module

This selection is made by double-clicking on the position to be configured (for example 6). The following dialog box appears :



Select the type of module (Communication) from the **Family** field, then the reference of the module to be configured from the (TSX SAY 100) **Module** field. After confirming with **OK**, the module is declared at its position (this is framed and contains the module reference).



### Warning

The maximum number of **TSX SAY100 modules** which can be installed in one configuration is as follows :

- 2 modules maximum with a TSX/PMX/PCX 57-1• processor,
- 4 modules maximum with a TSX/PMX 57-2• processor,
- 8 modules maximum with a TSX/PMX/PCX57-3•/4• processor,

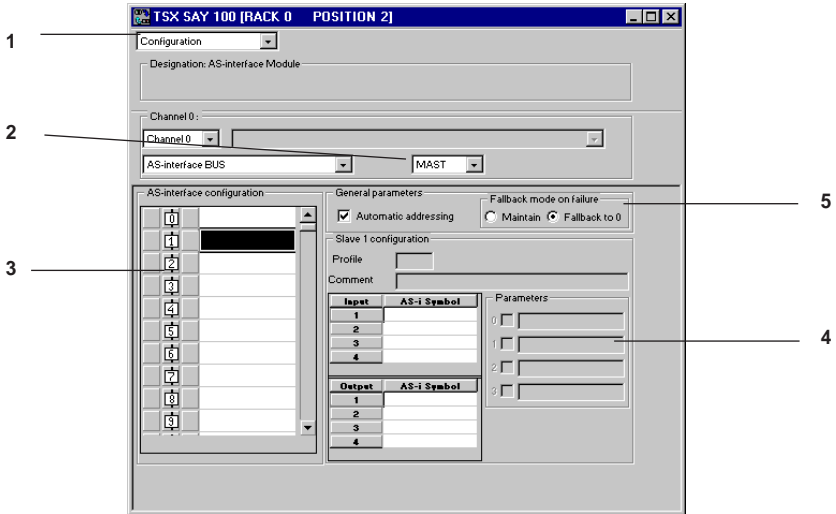
The TSX SAY 100 module can be used to manage 124 input bits and 124 output bits distributed over 31 devices.

### Note

To delete a module from its position, click on it to select it then press the <Del> key, which displays a dialog box. Then confirm deletion of the module.

### 2.3 Description of the module configuration screen

This screen provides access to the **display** and **modification** of parameters in offline mode, and to **debugging** in online mode.



- 1 This pulldown list can be used to select the operating mode : configuration (or debug).
- 2 This pulldown list can be used to select the task in which the data from the AS-i communication channel will be scanned. Select Mast or Fast task, given that the AS-i module processing cycle lasts 5 ms.
- 3 This field can be used to define the slave devices which are already connected or which are to be connected on the bus.
- 4 This field can be used to display and configure each of the slaves.
- 5 Field containing general parameters which apply to all the slaves on the bus.

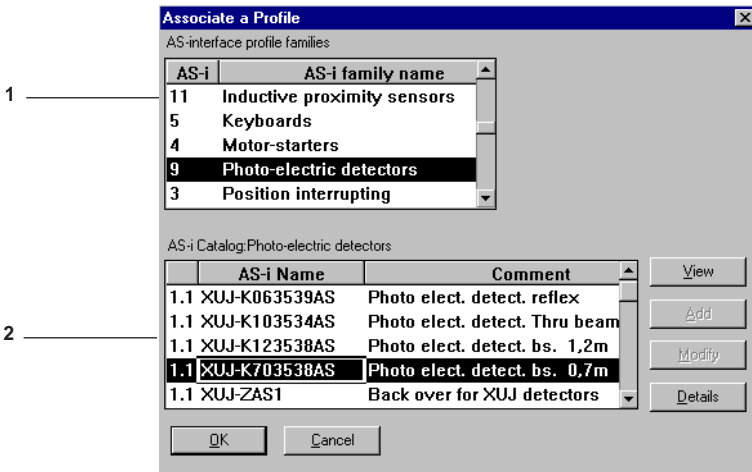
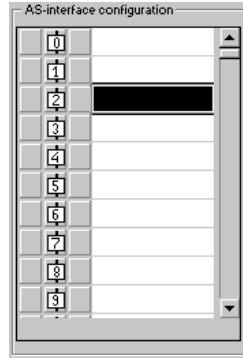
## 2.4 Selecting slave devices to be connected

### 2.4-1 Selection principle

The area of the screen entitled : "AS-interface configuration" can be used to configure all the projected slaves on the bus.

The slaves are numbered from 1 to 31 (access via the scroll bar). Address 0 is used by the system and cannot be configured.

Selecting the "**Edit/Add AS-i slave**" menu or double-clicking on the slot of the address of the slave to be configured opens an "Associate a profile" screen.



This screen gives access to the list of AS-i product families. Selecting one of the families (1) accesses the AS-i devices catalog (2).

The **OK** button can be used to confirm the selection.

#### Note :

The list of families has 2 particular elements :

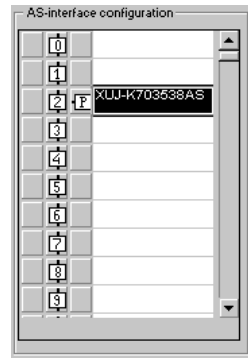
- standard profiles : selecting this family allows the user to select an AS-i profile from 240 possible profiles. The characteristics linked to this profile can be accessed by clicking on the **View** button,
- private families : a PL7 user can manage a specific AS-i device catalog file from his programming terminal (see the next section).

**Caution** : an application which uses AS-i products from the private family catalog is always linked to the use of the same private family catalog.

In the "AS-interface configuration" box, the reference for the connected device appears next to the slave number.

The following commands can be used to modify this selection :

- **"Edit / Delete AS-i Slave"** deletes the selected device.
- **"Edit / Cut AS-i Slave"** (Ctrl+X keys) deletes the selected device and stores it in the buffer memory.
- **"Edit / Copy AS-i Slave"** (Ctrl+C keys) stores the selected device in the buffer memory.
- **"Edit / Paste AS-i Slave"** (Ctrl+V keys) copies the device in the buffer memory to the selected slot.



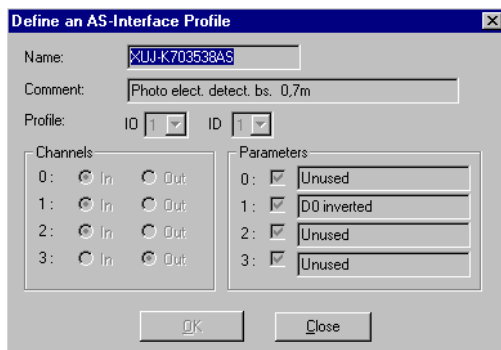
To move a device :

- 1 Select the device and execute the **Edit / Cut AS-i Slave** command.
- 2 Select the slot to which the device is to be moved and execute the **Edit / Paste AS-i Slave** command.

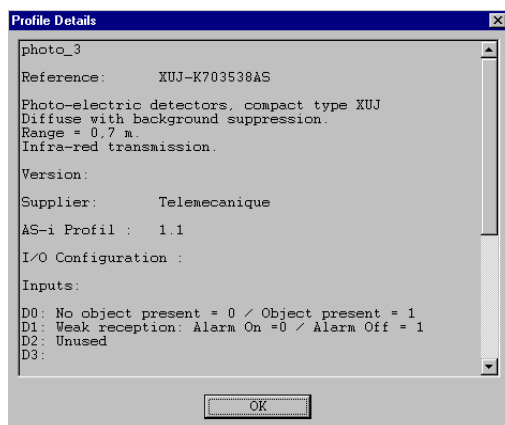
To duplicate a device :

- 1 Select the device and execute the **Edit / Copy AS-i Slave** command.
- 2 Select the slot at which the device is to be inserted and execute the **Edit / Paste AS-i Slave** command.

Selecting the **View** button in the "Associate a Profile" screen also provides a description of an AS-i slave.



It is also possible to access all the information in the catalog file by clicking the **Details** button.



### 2.4-2 Adding and modifying a profile in the catalog

From the "Associate a Profile" screen, it is possible, by selecting "Private Family" then "Add", to define the profile of a slave which is not available in the standard catalog.

To do this :

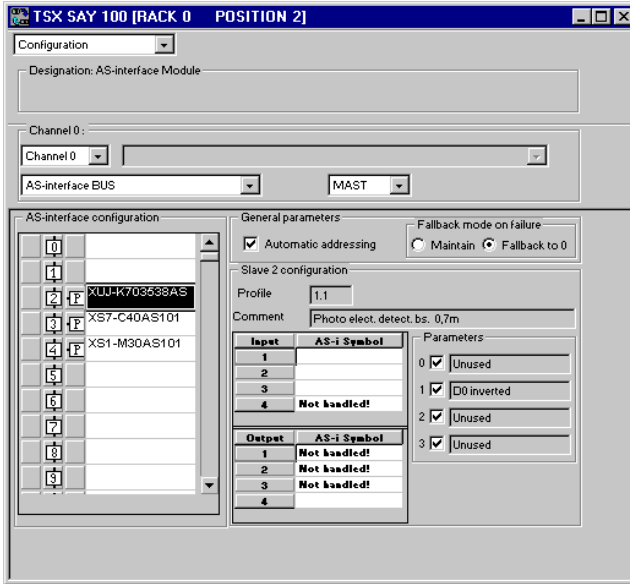
- define the name of the slave device,
- enter a comment (optional),
- define the I/O and ID data of the profile,
- select the inputs and outputs used by the slave,
- define the setting to 0 or 1 of each parameter linked to the slave device. The parameters for each slave can be programmed on 4 bits, and are specific to each slave. When a box is checked, the corresponding parameter is set to 1. By default, all of the parameters are at 1.

A slave defined in this way is added to the private family catalog.

This profile can then be used as a standard catalog profile. Pressing "**Modify**" allows the name and the comments relating to the profile of the slave to be modified. The I/O and ID data of the profile can no longer be modified once the window is confirmed.

## 2.5 Configuring slaves

The "Slave i configuration" zone in the Configuration screen displays the data associated with the slave selected in the "AS-interface configuration" zone.



The data in the "Slave i configuration" zone cannot be modified in this screen (except for the activation or deactivation of parameters).

**Profile :** consists of the I/O and ID data, it is determined by the type of device selected. It is defined by the user if the profile has been added.

**Comment :** this is determined by the type of device selected. It is defined by the user if the profile has been added and can be modified in the "Define an AS-Interface Profile" screen.

**Asi symbol :** the symbols associated with the slave I/O are defined using the variables editor.

**Parameters :** these are determined by the type of device selected. They are defined by the user if the profile has been added and can be modified in the "Define an AS-Interface Profile" screen.

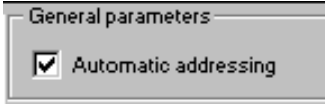
The parameters are activated and deactivated via checkboxes. By default, all the parameters are activated.



## 2.6 General parameters

### 2.6-1 Automatic addressing of slaves option

This option is selected by default in the configuration screen :



The automatic addressing option can be accessed by PL7 in AS-i protected mode (see Operating mode, section 5). It is confirmed in the configuration screen and in the "General parameters" window by checking the automatic addressing box. It is used to replace a faulty slave or to insert a new slave.

A new configuration with automatic addressing is not taken into account if one or more slaves with the address 0 are present on the bus. In this case the message "configuration refused by the module" appears on the screen.

#### Replacing a faulty slave :

In debug mode the **DIAG** buttons are red, and the P icon in the list of projected slaves is outlined in red. A faulty slave can be replaced by a slave of the same type easily without stopping the AS-i bus. If the replacement slave is programmed with the same address, using the pocket programmer, and if it has the same profile, it will be automatically inserted in the list of detected slaves (LDS) and activated.

If the new slave is brand new (address 0, new slave) and if it has the same profile, the slave automatically takes the address of the slave it replaces and is therefore in the list of detected slaves and in the list of activated slaves.

#### Minor modification of a configuration :

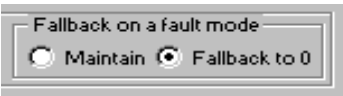
The automatic addressing option can also be used to make minor changes to the configuration without using the pocket programmer. If the slave to be inserted is projected in the PL7 configuration, if it has the profile as that which is expected, and if it has address 0 (as a new slave), then the AS-i module will program the slave with the value preset during configuration. This is only possible if **a single slave is missing from the configuration**. It is possible to modify an application by following the above procedure as many times as is necessary.

**Important :** for the insertion to be effective, it is necessary to :

- add the new slave in the configuration screen in offline mode,
- transfer the configuration to the PLC in online mode,
- physically connect the new slave with address 0 to the AS-i bus.

---

## 2.6-2 Output fallback position for slave devices



This selection enables the PLC to set the slave outputs to a defined fallback state. This fallback is activated on a change to stop or on a PLC fault.

### **Fallback to 0 position :**


The outputs of the AS-i slave present on the bus are forced to 0, then communication is stopped on the medium.

The %Q objects in the PLC are not modified.

### **Maintain position :**

The outputs of the AS-i slave present on the bus are maintained in the state which preceded the PLC fault or stop, then communication is stopped on the medium.

This mode can also be read in word %KWxy.0.19. Bit %KWxy.0.19 : X0 =1 : fallback to 0 and %KWxy.0.19 : X0 =0 : maintain state.

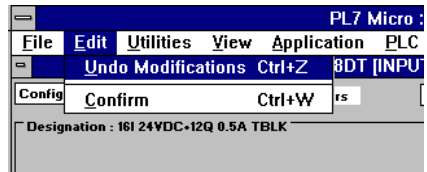
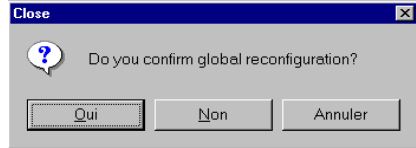
 **Caution** : the fallback mode on failure for slaves without a watchdog is not guaranteed if the AS-i bus is disconnected or if the AS-i power supply is lost. For slaves with a watchdog, the fallback position is predefined in the device.

## 2.7 Confirming the configuration

### 2.7-1 Confirming after modification

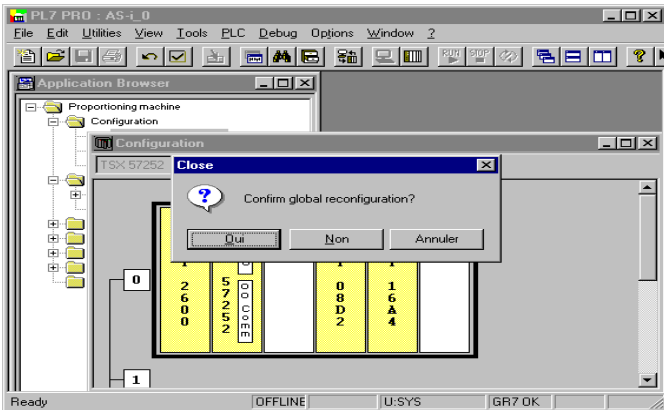
When quitting the function after modifying the module configuration parameters, the new configuration must be confirmed. To do this, several possibilities are available :

1. Confirm using the toolbar by clicking on the corresponding icon or by selecting the confirm command in the context menu.
2. Quit the function without confirming the parameters. This displays a dialog box which enables the user to confirm the new configuration.
3. Pull down the PL7 **Edit** menu and select **Confirm**.



### 2.7-2 Global reconfiguration

Quitting the configuration editor after modifying all the configuration parameters of the channels on each module makes a global reconfiguration necessary. When the editor is closed, a dialog box enables this global reconfiguration to be confirmed.



---

Global reconfiguration is required in offline mode, so that the modifications confirmed for each module are accepted by the application.

C

This reconfiguration is performed :

- using the "Confirm" icon, the **Edit/Confirm** command, or the confirm command in the context menu,
- by closing the configuration editor without global confirmation, and then confirming global reconfiguration.

### 3 Debugging the AS-i communication function

#### 3.1 Introduction to the debug function

---

This function can only be accessed online (via the **Connect** command in the **PLC** menu or by clicking on the corresponding icon).

For each AS-i communication module of the application, it displays the connection of the slaves and the parameters of each slave, and enables writing and forcing of the selected channel.

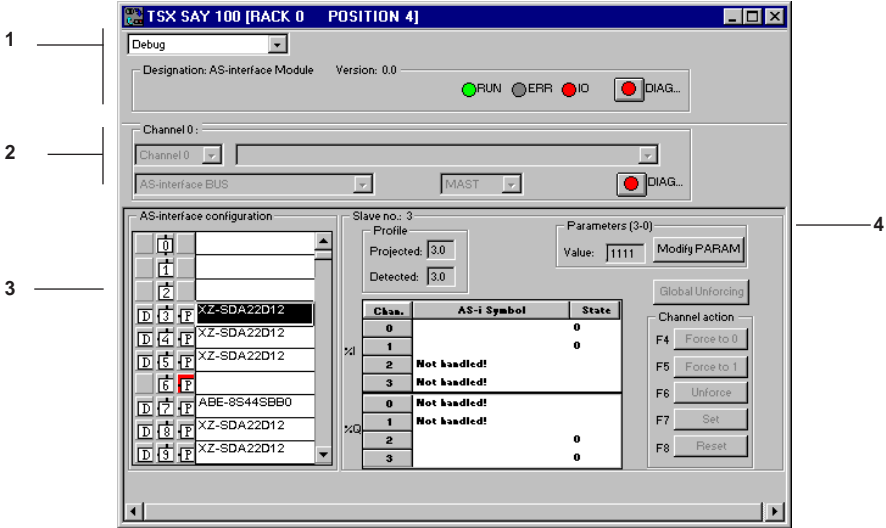
The function also gives access to module diagnostics in the event of a fault.

The AS-i communication module **Debug** function can be accessed via the Application Browser by double-clicking on the Station, the Configuration and Hardware configuration icons and then on the position of the module in the rack.

By default, the **Debug** function is selected in online mode, and the pulldown dialog box in the command field is used to return to the **Configuration** function.

## 3.2 Description of the module debug screen

This screen provides access to **Debugging** in online mode.



- 1 The pulldown list can be used to select the operating mode : debug (or configuration).
- 2 The module zone displays the status of the module RUN, I/O and ERR indicator lamps. It reports on the status of the module fault via the DIAG diagnostics button.
- 3 This zone is used to display the slave devices which are connected on the bus.
- 4 This zone is used to debug each of the slaves, using writing and forcing functions.

### 3.3 Diagnostics mode

In both these parts, the **DIAG** buttons are grayed out in normal operation, and are red if a fault is detected.



The **DIAG** button in the module zone detects module faults, and the **DIAG** button in the channel zone detects channel faults.

Clicking on the **DIAG** channel button opens the **Channel Diagnostics** window, which can be used to determine the type(s) of fault present.

The window is divided into three parts which are used to indicate the level at which the fault has occurred. For a module or a channel, faults can be internal, external, or other.

Example of the screen :



**External faults :**

- slave device faulty,
- line error (AS-i power supply switched off or terminal block error),
- difference between physical configuration and PL7 configuration,

**Internal faults :**

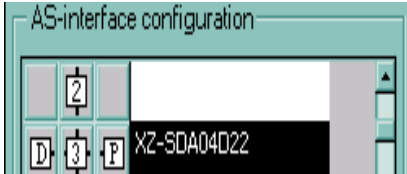
- internal software fault,
- communication fault with the processor,
- configuration, parameter definition, or control error.

### 3.4 Displaying the status of the slaves

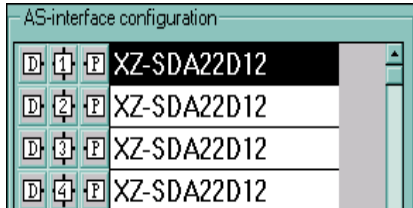
In this zone, the **P** and **D** icons are displayed either side of the slave number indicating that the slave has been projected **P** and detected **D**.

The 4 following cases may occur :

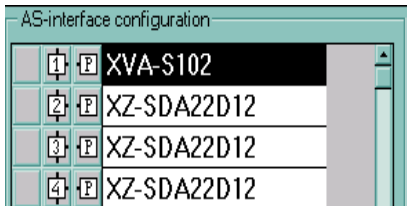
- The slave projected in the configuration and the detected slave are identical



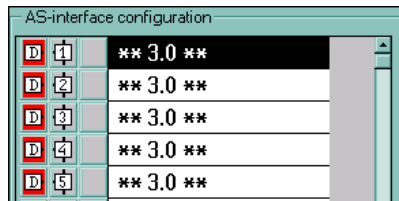
- The slave projected in the configuration and the detected slave are not identical



- A slave is projected in the configuration but no slave is detected



- An additional slave, which is not projected in the configuration but is connected on the bus, is declared faulty. When a slave is faulty, the icon is outlined in red to indicate that the slave is faulty, and the **DIAG** buttons also turn red.

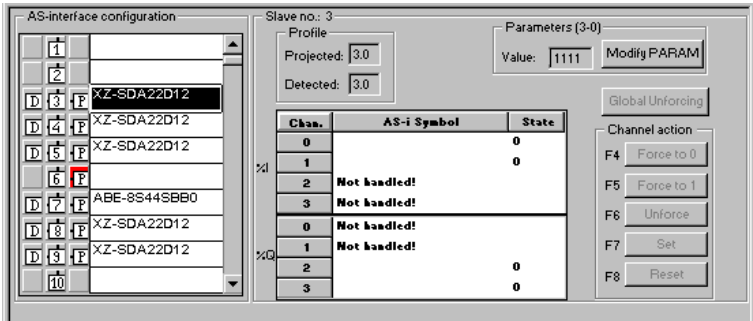


**Note :** The Profile box in the slave zone can be used to check whether the profiles of the projected slave and the detected slave are identical.



### 3.5 Adjusting the parameters

The lower part of the screen is reserved for AS-i bus diagnostics. It is used, on the left-hand side, to enter the address of the faulty slave. The corresponding P icon is then outlined in red. By selecting the address of the faulty slave, the right-hand part of the screen can be used to read the corresponding information.

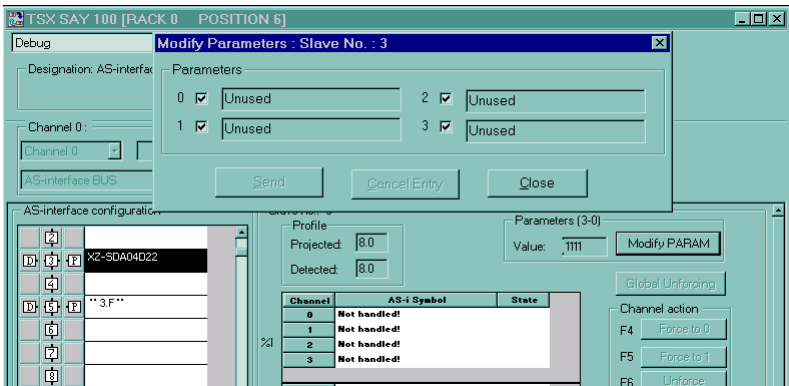


Displaying projected and detected profiles (see previous page)

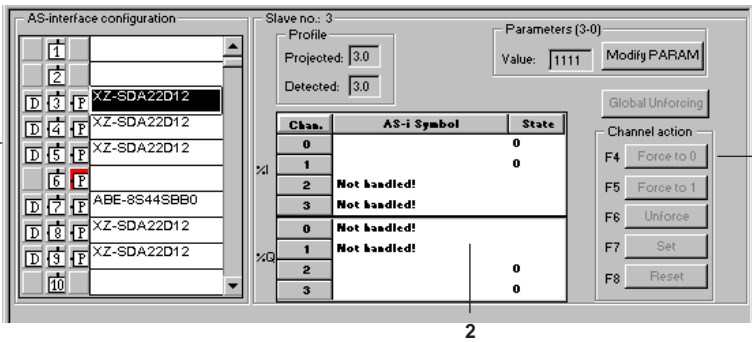
#### Modifying parameters

After selecting a slave, pressing the "Modify PARAM" button opens a window in which the parameter values can be modified (1= box checked).

Pressing the "OK" button transmits the new parameter values to the relevant device. The values are also updated in field (1) of the debug screen if the slave has accepted the parameter.



## Actions on the channels



After selecting a slave in the AS-interface configuration zone (1), select the channel to be modified in the table (2).

The buttons in the Channel Action zone (3) can be used to :

- Force the channel to 0 : 0F appears in the Status column of the table (2)
- Force the channel to 1 : 1F appears in the Status column of the table (2)
- Unforce the channel : F will disappear
- Set : set the channel to 1
- Reset : set the channel to 0

The Global Unforcing button cancel unforces all the channels linked to the slave.

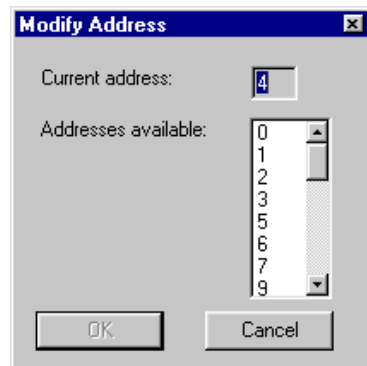
## Modifying the address

"Edit / Modify AS-i address" is used to move the selected device to another available address.

The screen for selecting a new address is displayed.

Select the address from the list of available addresses (use the scroll bar if necessary) and confirm with OK.

**Note :** This function is only operational for TSX SAY 100 modules with software version 2.0 or later.



## 4 Bits and words associated with the AS-i application

### 4.1 Addressing slave I/O on the AS-i bus

The reading of inputs and the updating of outputs from slave devices connected to the AS-i bus is performed automatically, at the beginning and end of each scan of the task in which the TSX SAY 100 module is configured respectively.

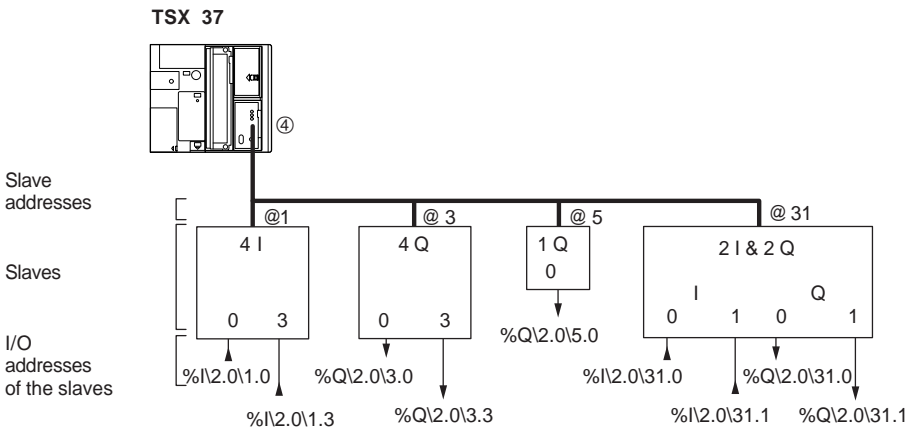
The user program can access these inputs and outputs using language objects with the following syntax :

<b>%</b>	<b>I or Q</b>	<b>\</b>	<b>4.0</b>	<b>\</b>	<b>n</b>	<b>.</b>	<b>i</b>
Symbol	Type of object I = input Q = output		Module/channel address for TSX SAY 100 x = rack number y = position number 0 = channel 0 of module		Slave number <b>0 to 31</b>		Bit rank <b>0 to 3</b>

**Examples :**

%I2.0\1.3 means : input 3 of slave 1, with the TSX SAY100 module located in slot 2 of rack 0.

%Q2.0\31.0 means : input 0 of slave 31, with the TSX SAY100 module located in slot 2 of rack 0.



**Reminder :** the physical address of an AS-i slave is programmed via the XZM C11 portable terminal.

---

## 4.2 Language objects associated with the AS-i module

---

The objects described below are not absolutely necessary for programming the AS-i function, but they provide additional information on the operation of the bus and the AS-i module. Additional functions are available for advanced programming of the function.

The AS-i standard defines three information lists for the slaves on the bus. These lists are available in the various PL7 objects as follows :

AS-i standard	Meaning	PL7 objects
List of <b>Projected Slaves</b> : LPS	List of slave addresses which must be taken into account when a configuration is received. In this case the AS-i bus is in AS-i protected mode.	AS-i protected mode corresponds to the PL7 configured mode, this list is memorized in configuration data : %KWxy.0.1 and %KWxy.0.2.
List of <b>Detected Slaves</b> : LDS	Memorization of the addresses of all the slaves present on the AS-i bus.	This list is memorized in the PL7 status data : %MWxy.0.4 and %MWxy.0.5. These exchanges are explicit.
List of <b>Activated Slaves</b> : LAS	List of addresses of projected and detected slaves. Their identification must also be consistent with that given in the reference configuration information.	This list is memorized in %IWxy.0.2 and %IWxy.0.3. These exchanges are implicit.



### 4.2-1 Configuration objects

This configuration data is initialized from the configuration screen. It is transmitted to the TSX SAY 10 module when a new configuration is received of on a warm or cold start.

Object	Function	Meaning
%KWxy.0	AS-i master	Byte 0 = 10 (AS-i identification in the communication function)
%KWxy.0.1 %KWxy.0.2	List of projected slaves : LPS	bit x = 1 if slave x is projected
%KWxy.0.3 to %KWxy.0.18	Configuration of I/O and identification code (ID) of configured slaves	Byte 0 = slave 0, bit 0-3 = I/O conf, bit 4-7 = ID code Byte 1 = slave 1, bit 0-3 = I/O conf, bit 4-7 = ID code
%KWxy.0.19	Fallback or insertion of slave	Byte 0 x0 = 0 if outputs fallback to 0 x0 = 1 if fallback with outputs maintained x1 = 0 no automatic addressing x1 = 1 automatic addressing possible
%KWxy.0.20 to %KWxy.0.51	Information on the catalog	catalog identifier slave 1 to slave 31  %KW4.0.20 corresponds to slave 0

### 4.2-2 AS-i channel status objects (implicit exchanges)

These objects are exchanged automatically on each PLC scan, and provide information on the current status of the AS-i segment.

Object	Function	Meaning
%lxy.0.ERR	Channel fault	= 1 if fault is on the AS-i segment
%lxy.0.0	Validity of inputs	= 1 if valid inputs = 0 if invalid inputs (offline mode, Data exchange off mode, or channel fault)
%lxy.0.0  %IWxy.0.1	List of faulty slaves	bit 0 = 1 if slave 0 is faulty or missing bit 1 = 1 if slave 1 is faulty or missing to bit 31 = 1 if slave 31 is faulty or missing
%IW xy.0.2  %IWxy.0.3	List of active slaves : LAS	bit 0 = 1 if slave 0 is activated bit 1 = 1 if slave 1 is activated to bit 31 = 1 if slave 31 is activated

A slave is declared activated if it belongs to the PL7 configuration, if it is detected as present, and if its configuration is correct (PL7 configuration = detected configuration).

### 4.2-3 AS-i channel status objects (explicit exchanges)

These objects are only exchanged on a PL7 instruction : READ\_STS %CHxy.0. The status provides information on all the slaves present on the AS-i communication channel.

Object	Function	Meaning
%MWxy.0.2	Standard status	Byte 0 bit 0 = not used bit 1 = 1 if one of the slaves is faulty bit 2 = 1 supply fault (power supply off or terminal block fault) bit 3 = 1 physical configuration different from PL7 configuration bit 4 = 1 internal software fault bit 5 = not used bit 6 = 1 communication fault with the processor bit 7 = 1 configuration fault in parameter setting or control
%MWxy.0.3	AS-i specific status	bit 0 = 1 if configuration is correct bit 1 = 1 if a slave with address 0 is present bit 2 = 1 automatic addressing selected during configuration bit 3 = 1 if automatic addressing is ready for operation bit 4 = 1 if the function is in configuration mode bit 5 = 1 if the function is in normal mode bit 6 = 1 if the AS-i supply is faulty bit 7 = 1 if offline phase is activated bit 8 = 1 data exchange inactive (Data exchange off mode)
%MWxy.0.4 %MWxy.0.5	List of detected slaves : LDS	bit 0 = 1 if a slave with address 0 is detected to bit 31 = 1 if a slave with address 31 is detected
%MWxy.0.6 %MWxy.0.7 to %MWxy.0.21	Configuration of I/O and identification code (ID) for all detected slaves	Byte 0 = slave with even address Byte 1 = slave with odd address bit 0-3 = I/O conf, bit 4-7 = ID code
%MWxy.0.22	Parameter data of the last slave for which parameters were defined	contains the response (value of parameters transmitted) of the last slave for which parameters were defined. This allows PL7 to check that the slave has received the information
%MWxy.0.23	Address of the last slave for which parameters were defined	contains the address of the last slave for which parameters were defined

**Note** : exchange words %MWxy.0.0 and report words %MWxy.0.1 are also used by the module, and are described in the part entitled "Common Features of Application-Specific Functions".

**4.2-4 Command objects of the AS-i communication channel**

A single command word can be used to manage the AS-i master switch to offline or Data exchange off state.

The PL7 instruction is : WRITE\_CMD %CHxy.0

Object	Function	Meaning
%MWxy.0.24	state command	bit 0 = 1 switch to offline state bit 1 = 1 exit offline state bit 2 = switch to Data exchange off state bit 3 = exit Data exchange off state

**4.2-5 Adjustment objects of the AS-i communication channel**

The AS-i slave parameter objects can be modified without stopping the AS-i function. The WRITE\_PARAM %CHxy.0 instruction is used to send the value of the PL7 adjustment data to the TSX SAY 100 module.

The READ\_PARAM %CHxy.0 instruction is used to read the current parameters in the TSX SAY 100 module, and therefore to initialize the value of %MW objects. If the parameters sent are refused, there is an inconsistency between the PL7 values and the values stored in the module.

The SAVE\_PARAM %CHxy.0 and RESTORE\_PARAM %CHxy.0 instructions are used to save the module parameters and to restore the PL7 parameters from the backup.

Object	Function	Meaning
%MWxy.0.25 to %MWxy.0.56	parameters of configured slaves	%MWxy.0.25 : parameters of slave 0 (not significant) %MWxy.0.26 : parameters of slave 1 ..... %MWxy.0.55 : parameters of slave 30 %MWxy.0.56 : parameters of slave 31

---

#### 4.2-6 TSX SAY 100 module language objects

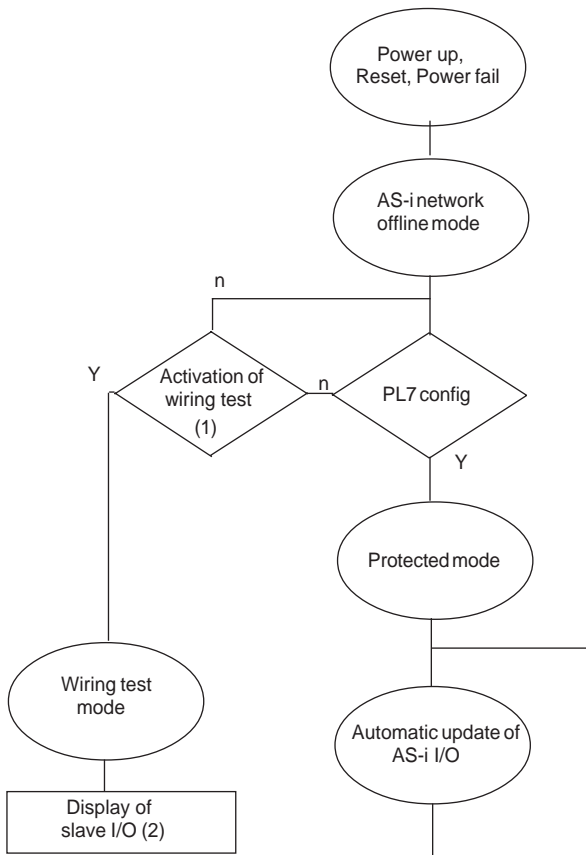
The TSX SAY 100 module has an associated status word and an error bit which summarize the state of the module.

Object	Function	Meaning
%Ix.MOD.ERR	Module fault	=1 if module fault
%MWxy.MOD.2	Standard module status	Byte 0 bit 0 = 1 if internal fault bit 1 = 1 if configuration fault bit 2 = 1 if supply fault bit 3 = not used bit 4 = not used bit 5 = not used bit 6 = 1 if module missing bit 7 = not used

**Note** : words %MWxy.MOD.0 and %MWxy.0.1 are also used by the module, and are described in the part entitled "Common Features of Application-Specific Functions".



### 5.1 General



Starting up the module

- power up the TSX/PMX/PCX 57
- action on the handle
- reset the base

Resetting the module

- self-tests
- detection of slaves present

Managing the AS-i network

- reading inputs
- writing outputs
- monitoring the network

- (1) Wiring test mode is activated using the top button on the front panel of the module.  
 (2) The module exits wiring test mode and switches to protected mode if it receives a configuration.

Protected mode is used for a configuration in operation. The other modes or services (wiring test mode, offline mode, Data exchange off mode) are used for help with diagnostics or start-up.

**Important :** information concerning the correspondence between the operating modes of the TSX/PMX/PCX 57 and those of the AS-i bus :

PLC	AS-i
Configured mode	Protected mode
Non-configured mode	Configured mode

---

## 5.2 AS-i protected mode

---

This mode is usually used for an application which is currently running. The module is configured by PL7 in the configuration screen where the slaves are declared.

In this mode, a slave is only activated if it has been declared during configuration and detected. The module continually checks that the list of detected slaves is the same as the list of projected slaves, and monitors the power supply.

---

## 5.3 AS-i wiring test mode

---

The AS-i wiring test is a module function which is only accessible if the PLC is in a "non-configured" state, or in one of the following cases :

- a rack equipped solely with a power supply and a TSX SAY 100 module,
- a PLC with a processor but no application,
- a PLC with a processor but no TSX SAY 100 module configured.

The wiring test cannot be operated by PL7 Junior / Pro, therefore :

- access is only possible from the front panel of the module (1),
- the objects which are accessible (in read only) are the I/O bits of the connected slaves and the LDS and LAS lists; the slave adjustment parameters cannot be accessed.

**Reminder** : in this mode, which can be used for a new installation, all the slaves must have different addresses in order to avoid clashes when a response is sent to the module (if there is a slave with address 0 on the bus, the module refuses to invalidate the configuration save).

(1) See the TSX DM 57 33E setup manual for how to test the bus wiring using the centralized display unit on the PLC and the display button on the module.

## 5.4 Advanced operating modes

The following modes are advanced operating modes which can be used for debugging or maintenance. They should only be used by specialists in AS-i communication.

### AS-i offline operating mode

This mode cannot be accessed from AS-i function screens in PL7 Junior / Pro software. On entering this mode, the module first resets all the slaves present to zero and stops exchanges on the bus.

Setting to offline mode can be achieved as follows :

- directly by the PL7 application program by setting bits of %MWxy.0.23
 

bit 1	bit 0	
0	0	normal operating mode
0	1	activation of offline mode
1	0	deactivation of offline mode
1	1	no effect

Bit 7 of the AS-i specific status word %MW xy.0.3 is set to 1, indicating that the AS-i offline phase is active.

- automatically on detection of an AS-i power supply fault :  
In this case, bits 6 and 7 (AS-i power supply fault and offline phase active) of the AS-i specific status word %MWxy.0.3, , are set to 1. When the fault disappears, both bits are reset to 0 and AS-i exchanges restart.

Setting the AS-i bus to offline mode :

AS-i power supply fault	PL7 offline phase	
0	0	normal operation
0	1	offline
1	0	offline
1	1	offline

In offline mode, the image of the I/O in the module is fixed in the state it was in when the mode was entered. On exiting the mode, if the list of present slaves (LPS) is the same as the list of detected slaves (LDS), the system restarts. If this is not the case, a fault is generated and it is necessary to go to diagnostics or configuration mode.

---

### AS-i "data exchange off" operating mode

This mode cannot be accessed from the AS-i function screens in PL7 Junior / Pro software.

In this mode, exchanges on the bus continue, but the data is no longer updated.

Setting to "data exchange off" mode can be achieved as follows :

- directly by the PL7 application program by setting bits of %MWxy.0.23

bit 3	bit 2	
0	1	activation of the data exchange off mode
1	0	deactivation of the data exchange off mode
1	1	no effect

Bit 8 of the AS-i specific status word %MW xy.0.3 is set to 1, indicating that the AS-i data exchange off mode is active.

## 6.1 Performance

### Managing the AS-i bus :

The AS-i module manages the AS-i bus independently. The performance graph below shows the exchange time between the slaves and the module. The module exchanges data between each slave configured on the bus and the module and vice versa. The internal memory of the module is updated on each exchange.

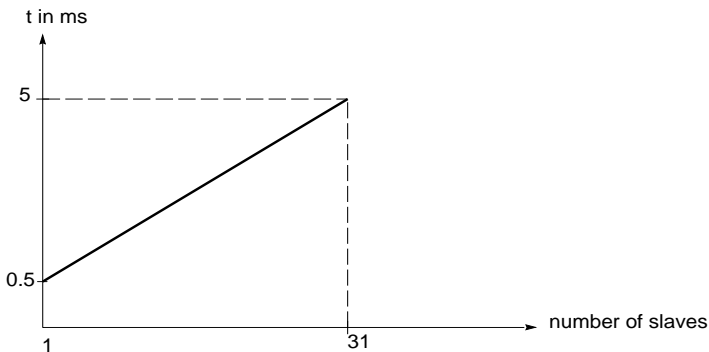
### Exchanges on the AS-i bus :

The maximum exchange time is 5ms for 31 activated slaves. This is the time required for all the slaves present on the bus to be exchanged and therefore for the internal memory of the module to be updated. The formula is as follows :

$$t = 156\mu\text{s} (n+2) \quad \text{if } n < 31$$

$$t = 156\mu\text{s} (n+1) \quad \text{if } n = 31$$

$n$  = number of activated slaves on the AS-i bus



The response time  $T$  is the time required between the activation of an input and the activation of an output on the same slave.

This time  $T$  is greater for a bus with 31 slaves in normal operation with no connection fault.

Examples for a PLC task of 10ms, 30ms, 60ms

PLC task	average T	max T
10ms	27 ms	37 ms
30ms	33 ms	55 ms
60ms	45 ms	80 ms



**Symboles****A**

Adding a profile	2/7
Addressing	4/1
Adjustment	3/5
Adjustment objects	4/5
Advanced operation	5/3
Application Browser	2/1
AS-i bus	1/2
AS-i channel status	4/3
AS-i slave	1/2
AS-i standard	4/2
AS-i wiring test mode	5/2
Associating a profile	2/6
Automatic slave addressing	2/9

**C**

Command objects	4/5
Configuration	2/1
Configuration objects	4/3
Configuration screen	2/3
Confirming the configuration	2/11
Connect	3/1
Connecting a device	2/4

**D**

"Data exchange off" mode	5/4
Debug screen	3/2
Debugging	3/1
Diagnostics	3/3
Diagnostics mode	3/3, 3/4

**E**

Explicit exchanges	4/4
External faults	3/3

**F**

Fallback to 0	2/10
Family	2/2
Faulty slave	2/9
Forcing	3/6

**I**

Implicit exchanges	4/3
Internal faults	3/3

**L**

Language objects	4/2
------------------	-----

**M**

Maintain	2/10
Master/slave	1/4
Modifying a profile	2/7
Module number	2/2

**O**

Offline mode	5/3
Operating mode	5/1

**P**

Performance	6/1
PLC menu	3/1
Profile	1/4
Protected mode	5/2

**R**

Reconfiguration	2/11
-----------------	------

**S**

Selecting a module	2/2
Slave configuration	2/8
Slave number	2/5
Slaves	3/4
Status of slaves	3/4

**T**

TSX SAY 100	1/3
TSX SAY 100 module	1/3

**V**

View	2/6
------	-----

---



<b>Section</b>	<b>Page</b>
<b>1 Presentation</b>	<b>1/1</b>
1.1 General	1/1
1.2 Wiring	1/1
1.3 Constituent elements	1/2
1.4 Requirements for using integrated MMI	1/3
<b>2 Description of the MMI functions</b>	<b>2/1</b>
2.1 General information on functions	2/1
2.1-1 Presentation	2/1
2.1-2 MMI function operating modes	2/2
2.2 SEND_MSG function	2/10
2.2-1 Function	2/10
2.2-2 Description	2/10
2.3 GET_MSG function	2/13
2.3-1 Function	2/13
2.3-2 Description	2/13
2.4 ASK_MSG function	2/16
2.4-1 Function	2/16
2.4-2 Description	2/16
2.5 SEND_ALARM function	2/20
2.5-1 Function	2/20
2.5-2 Description	2/20
2.6 DISPLAY_MSG function	2/23
2.6-1 Function	2/23
2.6-2 Description	2/23

<b>Section</b>	<b>Page</b>
2.7 DISPLAY_GRP function	2/24
2.7-1 Function	2/24
2.7-2 Description	2/24
2.8 DISPLAY_ALRM function	2/25
2.8-1 Function	2/25
2.8-2 Description	2/25
2.9 ASK_VALUE function	2/27
2.9-1 Function	2/27
2.9-2 Description	2/27
2.10 GET_VALUE function	2/29
2.10-1 Function	2/29
2.10-2 Description	2/29
2.11 CONTROL_LEDS function	2/30
2.11-1 Function	2/30
2.11-2 Description	2/30
2.12 ASSIGN_KEYS function	2/31
2.12-1 Function	2/31
2.12-2 Description	2/31
2.13 PANEL_CMD function	2/33
2.13-1 Function	2/33
2.13-2 Description	2/33
2.14 ADJUST function	2/35
2.14-1 Functions	2/35
2.14-2 Description of the parameters zone	2/35
2.14-3 Examples	2/37
2.15 Station documentation file	2/42

---

<b>Section</b>	<b>Page</b>
<b>3 Interface variables</b>	<b>3/1</b>
3.1 General	3/1
3.2 Operator panel address	3/2
3.3 Data to Send	3/4
3.4 Data to Receive	3/5
3.5 Report	3/5
<b>4 Precautions when using MMI and example of application</b>	<b>4/1</b>
4.1 Precautions when using MMI	4/1
4.2 Example	4/2

<b>Section</b>	<b>Page</b>
<b>5 Appendix</b>	<b>5/1</b>
5.1 Description of interface parameters	5/1
5.2 Description of coding «Data to Send» parameter of Integrated MMI functions	5/3
5.2-1 Displaying a PLC status message : SEND_MSG function	5/3
5.2-2 Status message entry controlled by the PLC : ASK_MSG and GET_MSG functions	5/6
5.2-3 Displaying a PLC alarm message : SEND_ALARM function	5/9
5.2-4 Displaying status message or alarm message or group of messages contained in the CCX17 memory : ASK_VALUE, DISPLAY_MSG, GET_VALUE, DISPLAY_ALARM, DISPLAY_GRP functions.	5/11
5.2-5 Displaying the LEDs on the indicator bank : CONTROL_LEDS function	5/12
5.2-6 Configuring the command keys : ASSIGN_KEYS function	5/13
5.2-7 Sending generic commands : PANEL_CMD function	5/14
<b>6 Index</b>	<b>6/1</b>

---

## 1.1 General

---

CCX 17 operator panels can be installed very easily using the dedicated functions called **Integrated MMI**. These functions are basic language elements.

Designed to standardize the use of an MMI terminal on a TSX Micro/TSX Premium PLC, these functions are used to trigger the display of messages or groups of messages, alarms or to enter values from the PLC program without using the means of communication between the operator panel and the PLC. The MMI function is thus totally integrated in the PLC application (**Consistency of data, Single save, Easy maintenance, Standardized operator panels, etc**).

The processing of these functions is **ASYNCHRONOUS** with the processing of the operative task which activated them.

---

## 1.2 Wiring

---

Please refer to the TSX Micro/Premium installation manual.

### 1.3 Constituent elements

These basic services provide :

1. on the one hand, the possibility of controlling the main functions of a CCX 17 operator panel **which contains no application** (the panel neither having been configured nor loaded by the external development software).
2. on the other hand, the possibility of controlling a CCX 17 operator panel **containing an application** created using **MMI17 WIN** or **PL7-M17 OS/2**.

They are in the form of a family of functions entitled **Integrated MMI**, which consists, for each command, of a description of the procedure and a predefined screen appropriate to the CCX 17 operator panel.



Library: Functions installed on this terminal

Function Information: <None> [down arrow]

Family	Lib.V.	↑	Name	Comment	↑
Integrated MMI	1.5		ASK_MSG	Blocking entry of a variable in CCX17	
Movement Command	1.0		ASK_VALUE	Blocking entry of a variable in a message>	
Orphee functions	1.2		ASSIGN_KEYS	Dynamic assignment of keys located at t>>	
Process control	1.8	↓	CONTROL_LEDS	Command to control LEDs located on th>>	↓

Close

They provide the following commands :

1. For a CCX17 which **contains no application**,
  - **SEND\_MSG**: display on CCX 17 of status messages contained in the PLC memory with or without variables
  - **GET\_MSG**: (asynchronous) free entry of values for PLC variables associated with the status messages
  - **ASK\_MSG**: (synchronous) blocking entry of a value for PLC variables associated with status messages
  - **SEND\_ALARM**: display of alarm messages contained in the PLC memory

- 
2. For a CCX17 which contains an application created using MMI17 WIN or PL7-M17 OS/2,
    - **DISPLAY\_MSG** : display a status message contained in the CCX17 memory
    - **DISPLAY\_GRP** : display a status message group contained in the CCX17 memory
    - **DISPLAY\_ALARM** : display an alarm message contained in the CCX17 memory
    - **GET\_VALUE** : (asynchronous) free entry of values for PLC variables associated with a status message contained in the CCX17 memory
    - **ASK\_VALUE** : (synchronous) blocking entry of values for PLC variables associated with a status message contained in the CCX17 memory
  
  3. For a CCX17 with or without an application,
    - **CONTROL\_LEDS** : control of the CCX 17 LEDs and relay
    - **ASSIGN\_KEYS** : configuration of the CCX 17 command keys
    - **PANEL\_CMD** : send a generic command
    - **ADJUST** : adjusts language objects

---

## 1.4 Requirements for using integrated MMI

---

**Integrated MMI** functions require program space of **1 Kword** (4.7 Kwords for ADJUST) and the reservation of variables for the data to be displayed (use %KW<sub>i</sub> constants). They operate with **version 2** CCX 17 operator panels. **Management of the CCX17 relay** requires version **2.1** or higher.

---

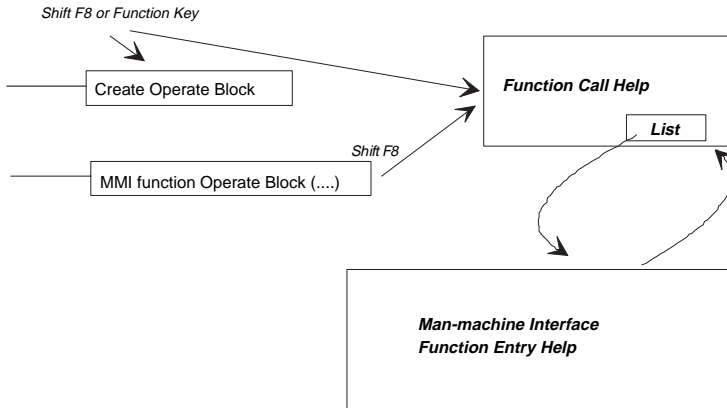
D



## 2.1 General information on functions

### 2.1-1 Presentation

Integrated MMI functions can be accessed in two different ways :



1. Create the MMI function block.  
The Shift F8 key (or the Menu function) creates the operate block and activates the Function call help.
2. Modify the MMI function block.  
After selecting the function (double-click left mouse button), the Shift F8 key (or double-click right mouse button) recalls the Function Call help with the block parameters.

**Note:**

When calling the specific help, the switch parameters are not necessary.


## 2.1-2 MMI function operating modes

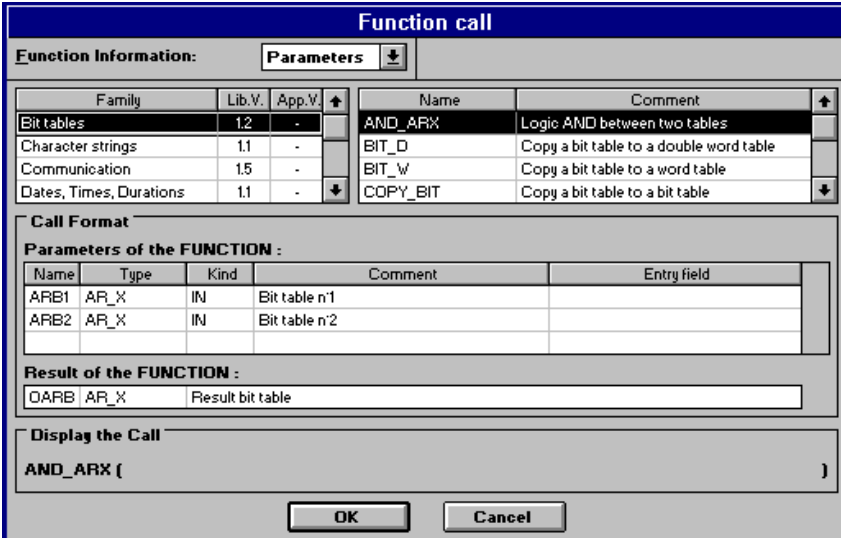
Assisted entry for functions is used to obtain the function code easily. The **Details** button is used to access the various parameters of the function appropriate to the CCX 17 operator panel.

### Using assisted entry of functions in LADDER

The principle described below is illustrated by the following example :

We intend to display the message « Oven Temperature : xxx.xx °C » including a data item from the process whose expected value is shown in word %MW100 of the application and will be displayed on the left in line 2.

1. Press the SHIFT - F8 keys simultaneously or select the  icon and place it in the rung. The Function call window is then displayed



**Function call**

Function Information: **Parameters**

Family	Lib.V.	App.V.	Name	Comment
Bit tables	1.2	-	AND_ARX	Logic AND between two tables
Character strings	1.1	-	BIT_D	Copy a bit table to a double word table
Communication	1.5	-	BIT_w	Copy a bit table to a word table
Dates, Times, Durations	1.1	-	COPY_BIT	Copy a bit table to a bit table

**Call Format**

Parameters of the FUNCTION :

Name	Type	Kind	Comment	Entry field
ARB1	AR_X	IN	Bit table n1	
ARB2	AR_X	IN	Bit table n2	

Result of the FUNCTION :

QARB AR\_X Result bit table

Display the Call

AND\_ARX ( )

OK Cancel

2. Choose the required family

Family	Lib.V.	App.V.
Event	1.0	-
Explicit exchanges	1.0	-
Integer tables	1.1	-
Integrated MMI	1.5	-

3. Select the name of the function

Name	Comment
GET_VALUE	Multiple entry of a variable in a message>>
PANEL_CMD	Send a command to CCX17
SEND_ALARM	Display an alarm message contained in t>>
SEND_MSG	Display a message contained in the PL>>

4. Press the **Details...** key

The following dialog box appears

5. First complete the fields associated with the message. The text of the message contains a variable zone where the expected value will be displayed. To specify the position of the variable zone, all that is required (by convention) is to enter an underscore at the appropriate place.

6. Initialize the data specific to the field type. By default, the value is updated periodically.

Field

Field Type:  None  Address  Date  Time

Symbol: Oven Address: %MW100

Comment: Water temperature  Update

7. Change the display format by clicking on the **Modify ...** button and choose the format type and number of digits before and after the decimal point, and confirm with **OK** or <Enter>.

Modify Display Format

Variable Type: WORD

Format Type:  Numerical  ASCII

Signed

Digits Before the Decimal Point: 3

Digits After the Decimal Point: 2

Display Format: 999.99

OK Cancel

8. Program the Parameters zone

Parameters

Terminal Address: %MW10 : 6 Report: %MW0 : 4

Data to Transmit: %KW10 : 41

9. Confirm the box with **OK** or <Enter> and the function block appears

OPERATE

SEND\_MSG(%MW10:6,%KW10:41,%MW0:4)

10. Press <Enter>

11. Confirm.

## Using the assisted entry of functions in IL

The principle described below is illustrated by the following example :

We intend to display the alarm message « Oven overheating : xxx.xx °C » including a data item from the process whose expected value is indicated in word %MW100 of the application.

1. Enter the instruction up to the function call :

```
! LD [Oven >= Overheating]
```

2. Press the SHIFT - F8 keys simultaneously or select the command **Utilities/Enter function call**

**Function call**

Function Information: Parameters

Family	Lib.V.	App.V.	Name	Comment
Bit tables	1.2	-	AND_ARX	Logic AND between two tables
Character strings	1.1	-	BIT_D	Copy a bit table to a double word table
Communication	1.5	-	BIT_W	Copy a bit table to a word table
Dates, Times, Durations	1.1	-	COPY_BIT	Copy a bit table to a bit table

**Call Format**

Parameters of the FUNCTION :

Name	Type	Kind	Comment	Entry field
ARB1	AR_X	IN	Bit table n1	
ARB2	AR_X	IN	Bit table n2	

Result of the FUNCTION :

OARB | AR\_X | Result bit table

Display the Call

AND\_ARX ( )

OK Cancel

3. Choose the required family

Family	Lib.V.	App.V.
Event	1.0	-
Explicit exchanges	1.0	-
Integer tables	1.1	-
Integrated MMI	1.5	-

For each family, the software shows the name, library version and application version.

4. Select the name of the function

Name	Comment	
GET_VALUE	Multiple entry of a variable in a message>>	↑
PANEL_CMD	Send a command to CCX17	
<b>SEND_ALARM</b>	<b>Display an alarm message contained in t&gt;&gt;</b>	
SEND_MSG	Display a message contained in the PL>>	↓

For each function, the software shows :

- the name
- the comment (if this is not shown in full, double-click on it)

5. Press the **Details...** key

The following dialog box appears

6. First complete the fields associated with the message. The text of the message contains a variable zone where the expected value will be displayed. To specify the position of the variable zone, all that is required (by convention) is to enter an underscore at the appropriate place.

7. Initialize the data specific to the type of field.

<b>Field Symbol :</b>	<input type="text" value="Oven"/>	<b>Address :</b>	<input type="text" value="%MW100"/>
<b>Comment :</b>	<input type="text" value="Water temperature"/>		

8. Change the display format by clicking on the **Modify ...** button and choose the type of format and the number of digits before and after the decimal point and confirm with **OK** or <Enter>.

**Modify Display Format**

Variable Type : WORD

**Format Type**

Numerical

ASCII

Signed

Digits Before the Decimal Point :

Digits After the Decimal Point :

Display Format : 999.99

**OK**

**Cancel**

9. Program the Parameters zone

<b>Parameters</b>			
<b>Terminal Address :</b>	<input type="text" value="%MW10"/>	<input type="text" value="6"/>	<b>Report :</b>
<b>Data to Transmit :</b>	<input type="text" value="%KW10"/>	<input type="text" value="30"/>	<input type="text" value="%MW0"/>
			<input type="text" value="4"/>

10. Confirm with **OK** or <Enter> and the function block appears

```
! LD [Oven >= Overheating]
  [SEND_ALARM (%MW10:6, %KW10:30, %MW0:4)]
```

**Do not forget to put the function between <[> and <]> characters**

11. Confirm.

**Note :**

If the syntax of the function is known, enter the name directly into the editor and press Shift F8.

## Using the assisted entry of functions in ST

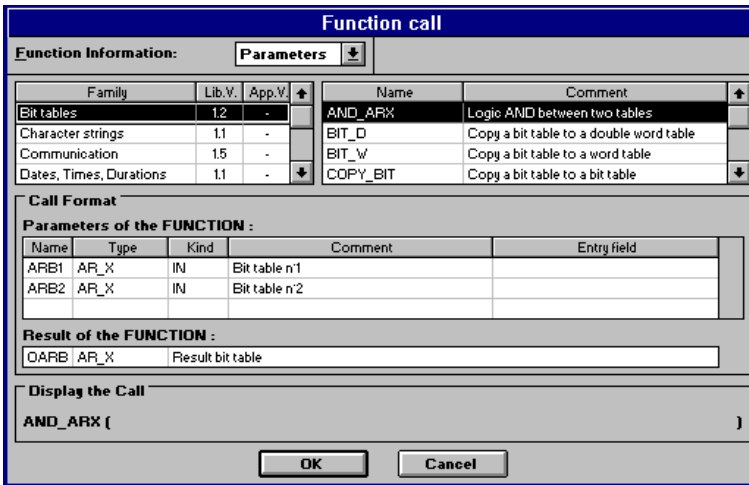
The principle described below is illustrated by the following example :

We intend, on user request, to display after group n°1 the number 2 group, previously defined within the CCX17 operator panel.

1. Enter the instruction up to the function call :

```
! (* On user request, display group 2 of CCX 17 *)
IF %M10 THEN
    %MW201 := 2;
```

2. Press the SHIFT - F8 keys simultaneously or select the **Utilities/Enter Function Call** command



3. Choose the required family

Family	Lib.V.	App.V.	
Event	1.0	-	
Explicit exchanges	1.0	-	
Integer tables	1.1	-	
Integrated MMI	1.5	-	

For each family, the software shows the name, library version and application version.



4. Select the name of the function

Name	Comment	
CONTROL_LEDS	Command to control LEDs located on the >	↑
DISPLAY_ALARM	Display an alarm contained in the CCX17	
<b>DISPLAY_GRP</b>	<b>Display a group of messages contained in &gt;</b>	
DISPLAY_MSG	Display a message contained in the CCX17	↓

For each function, the software shows :

- the name
- the comment (if this is not shown in full, double-click on it)

5. Press the **Details...** key

The following dialog box appears

6. Program the Parameters zone (If the “Data to Send” is %MWi type, the field “Message Group Number” cannot be accessed)

7. Confirm the box with **OK** or <Enter> and the function block appears

```
! (* On user request, display group 2 of CCX 17 *)
IF %M10 THEN
    %MW201 := 2;
    DISPLAY_GRP (ADR#0.0.4, %MW201, %MW0:4);
END_IF;
```

8. Confirm.

**Note** : If the syntax of the function is known, enter the name directly into the editor and press Shift F8/click right mouse.

---

## 2.2 SEND\_MSG function

---

### 2.2-1 Function

This is used to display on the screen of a CCX 17 operator panel, a status message possibly containing a dynamic variable.

D

---

### 2.2-2 Description

**SEND\_MSG**

**Parameters**  
Terminal Address :  Report :   
Data to Transmit :  :

**Message**  
Text :   Print  
Position  
Mode  
 Manual  
 Automatic  
Line :  Column :   
Column Alignment  
 Left  Centred  Right  
Attributes  
 Flashing  
 Reverse Video  
Size  
 Standard  
 Double  
Clear  
 None  Line  Screen

**Field**  
Field Type  
 None  Address  Date  Time  
Symbol :  Address :   Update  
Comment :   
Display Format

#### Parameters

These 3 fields are explained in section 3. They are compulsory. For the «Data to Send» field, use a %KW<sub>i</sub> type variable and leave the **Length** field blank. In any case, the system automatically calculates the strictly usable size.

<b>Message text</b>	Entry zone for text of the message displayed on the CCX 17 operator panel. It contains a maximum of <b>40</b> characters. The «_» sign (underscore) is reserved by the system to specify the display zone for the variable, if required, associated with the message. To define the position of the variable zone, all that is required is to enter (by convention) an underscore at the required position. The system then automatically calculates the number of «underscores» necessary for the display length of the variable. Characters whose ASCII code is less than 32 (20h) are not displayed. Conversely, those whose value is greater than 32 (20h) can be displayed either directly or via a combination of the «ALT» and «number» keys.
<b>Print</b>	Specifies that the message is printed when print is confirmed in the operator panel.
<b>Line</b>	Identifies the display line of the message (default value = 1).
<b>Column</b>	Identifies the display column of the message (default value = 1, min value 1, max value 40). In order to simplify the formatting of the message, PL7 Junior offers an <b>AUTOMATIC</b> mode for entering the column number. In this case, all that is required is to specify the type of <b>COLUMN ALIGNMENT</b> (Left, Center, Right), and the <b>COLUMN</b> field is calculated automatically. <b>AUTOMATIC</b> mode is local to the entry.
<b>Attributes</b>	Defines the display attributes of the message ( <b>normal, flashing, reverse video</b> ).
<b>Size</b>	Specifies the format of all the characters of the text or the variable to be displayed ( <b>Standard, Double</b> ).
<b>Delete</b>	This field is used to associate a single command to the message, which is executed before displaying the message. The choices are either <b>None</b> (no command associated with the message), or <b>Line</b> (deletes the line on which the message will be displayed) or <b>Screen</b> (clears the whole screen).

**Very important:**

If no variable is associated with the message, this command is not valid (use the PANEL\_CMD function).

<b>Field type</b>	Defines the type of field of the parameter displayed which is associated with the message. The possibilities are either <b>None</b> (no field is associated with the message displayed), or <b>Address</b> (the message is associated with a variable), or <b>Date</b> (the message is correlated with the current date) or <b>Time</b> (the message is associated with the current time). The date and time are synchronized periodically with those of the PLC.
<b>Symbol</b>	Specifies the symbol of the variable associated with the message. It must be defined in the station database. The address associated with this symbol is taken into account automatically when the screen is confirmed. <b>Comment</b> : On a TSX Agent on a FIPIO bus, the variable is read in the master PLC of the bus and not in the PLC sending the function.
<b>Address</b>	Specifies the address of the variable associated with the message. If the symbol exists in the station database, it is automatically taken into account. Permitted objects are either internal bits ( <b>%Mi</b> ), or internal words ( <b>%MWi</b> ) or internal double words ( <b>%MDi</b> ).
<b>Comment</b>	Displays the comment of the variable for consultation. It is defined in the station database.
<b>Update</b>	Specifies whether the variable contained in the message is to be updated periodically during display (Default value).
<b>Format</b>	Specifies the display format of the variable. The formats available are either <b>Numerical</b> or <b>Ascii</b> . The dialog box below, accessed via the «Modify» key, enables you to change the default display format, by specifying whether the variable is <b>signed</b> and defining the number of characters required before or after the <b>decimal point</b> . From the information shown, the software automatically calculates the display format

Modify Display Format

Variable Type : WORD

OK

Cancel

Format Type

Numerical

ASCII

Signed

Digits Before the Decimal Point : 3

Digits After the Decimal Point : 2

Display Format : 999.99

## 2.3 GET\_MSG function

### 2.3-1 Function

This is used to display on the screen of a CCX 17 operator panel a status message containing a variable which can be modified by the operator. Entry is performed in **multiple** mode. The operator can enter several successive values. In this case, the PLC program processes the value entered when the variable appears.

### 2.3-2 Description

The screenshot shows the 'GET\_MSG' configuration window. It is divided into several sections:

- Parameters:** Terminal Address: [ ] : 6, Report: [ ] : 4, Data to Transmit: [ ] : [ ]
- Message:** Text: [ ],  Print
- Position:** Line: [ 1 ], Column: [ 1 ]
- Mode:**  Manual,  Automatic
- Column Alignment:**  Left,  Centred,  Right
- Attributes:**  Flashing,  Reverse Video
- Size:**  Standard,  Double
- Field:** Symbol: [ ], Address: [ ],  Update, Comment: [ ], Display Format: [ ]
- Entry:** Value:  Not Checked,  Limited,  Increment

Buttons:

#### Parameters

These first 3 fields are explained in section 3. They are compulsory.

---

<b>Message text</b>	<p>Entry zone for text of the message displayed on the CCX 17 operator panel. It contains a maximum of <b>40</b> characters. The « _ » sign (underscore) is reserved by the system to specify the display zone for the variable, if required, associated with the message. To define the position of the variable, all that is required is to enter (by convention) an underscore at the required position. The system then automatically calculates the number of «underscores» necessary for the display length of the variable.</p> <p>Characters whose ASCII code is less than 32 (20h) are not displayed. Conversely, those whose value is greater than 32 (20h) can be displayed either directly or via a combination of the «ALT» and «number» keys.</p>
<b>Print</b>	Specifies that the message is printed when print is confirmed in the operator panel.
<b>Line</b>	Identifies the display line of the message (default value = 1).
<b>Column</b>	Identifies the display column (default value = 1, min value 1, max value 40). In order to simplify the formatting of the message, PL7 Junior offers an <b>AUTOMATIC</b> mode for entering the column number. In this case, all you have to do is specify the type of <b>COLUMN ALIGNMENT</b> (Left, Center, Right) and the COLUMN field is calculated automatically. The <b>AUTOMATIC</b> mode is local to the entry.
<b>Attributes</b>	Defines the display attributes of the message ( <b>normal</b> , <b>flashing</b> or <b>reverse video</b> ).
<b>Size</b>	Specifies the format of all the characters of the text or the variable to be displayed ( <b>Standard</b> or <b>Double</b> ).
<b>Symbol</b>	<p>Specifies the symbol of the variable associated with the message. It must be defined in the station database. The address associated with this symbol is taken into account automatically when the screen is confirmed.</p> <p><b>Comment:</b> On a TSX Agent on a FIPIO bus, the variable is read in the master PLC of the bus and not in the PLC sending the function.</p>

---

<b>Address</b>	Specifies the address of the variable associated with the message. If the symbol exists in the station database, it is automatically taken into account. Permitted objects are either internal bits ( <b>%Mi</b> ), or internal words ( <b>%MWi</b> ) or internal double words ( <b>%MDi</b> ).
<b>Comment</b>	Displays the comment of the variable for consultation. It is defined in the station database.
<b>Update</b>	Specifies whether the variable contained in the message is to be updated periodically during display (Default value).
<b>Format</b>	Specifies the display format of the variable. The only format available is <b>Numerical</b> . The dialog box below, accessed via the "Modify" key, enables you to change the default display format, by specifying whether the variable is <b>signed</b> and defining the number of characters required before or after the <b>decimal point</b> . From the information shown, the software automatically calculates the display format.

<b>Value</b>	Specifies whether, when entering the value associated with the variable, it is performed freely ( <b>Not checked</b> ) or must be monitored within min or max limits ( <b>Limited</b> ) or lastly, is performed by increment ( <b>Increment</b> ).
--------------	--

---

## 2.4 ASK\_MSG function

---

### 2.4-1 Function

This is used to display on the screen of a CCX 17 operator panel, a status message containing a variable which can be modified by the operator. The entry is performed in **synchronized** mode. It is valid for a single operator input with each message displayed. In this event, the value entered is returned to the ASK\_MSG function.

---

### 2.4-2 Description

The screenshot shows the ASK\_MSG dialog box with the following sections and controls:

- Parameters:**
  - Terminal Address : [ ] : 6
  - Data to Receive : [ ] : [ ]
  - Data to Transmit : [ ] : [ ]
  - Report : [ ] : 4
- Message:**
  - Text : [ ]
  - Print
- Position:**
  - Line : 1 (up/down arrows)
  - Column : 1 (up/down arrows)
  - Mode:  Manual,  Automatic
  - Column Alignment:  Left,  Centred,  Right
- Attributes:**
  - Flashing
  - Reverse Video
- Size:**
  - Standard
  - Double
- Field:**
  - Symbol : [ ]
  - Address : [ ]
  - Comment : [ ]
  - Display Format: [ ]
- Entry:**
  - Value:  Not Checked,  Limited,  Increment

Buttons:



<b>Parameters</b>	<p>These first 4 fields are explained in section 3. They are compulsory. The “<b>Data to Receive</b>” field defines the word which receives the value entered by the operator on the CCX 17 operator panel. It must be located in an internal word (%MW). The length of this field has a value of at least 2.</p> <p><b>Very important :</b> When using the MMI operator panel, the « <b>Data to Receive</b> » parameter contains the value entered. The <b>address/symbol</b> indicated in the following fields is not modified by the entry. It only applies when displaying on the CCX 17.</p>
<b>Message text</b>	<p>Entry zone for text of the message displayed on the CCX 17 operator panel. It contains a maximum of <b>40</b> characters. The “_” sign (underscore) is reserved by the system to specify the display zone for the variable, if required, associated with the message. To define the position of the variable, all that is required is to enter (by convention) an underscore at the required position. The system then automatically calculates the number of “underscores” necessary for the display length of the variable. Characters whose ASCII code is less than 32 (20h) are not displayed. Conversely, those whose value is greater than (20h) can be displayed either directly or via a combination of the “ALT” and “number” keys.</p>
<b>Print</b>	<p>Specifies that the message is printed when print is confirmed in the operator panel.</p>
<b>Line</b>	<p>Identifies the display line of the message (default value = 1).</p>
<b>Column</b>	<p>Identifies the display column of the message (default value = 1, min value 1, max value 40). In order to simplify the formatting of the message, PL7 Junior offers an <b>AUTOMATIC</b> mode for entering the column number. In this case, all that is required is to specify the type of <b>COLUMN ALIGNMENT</b> (Left, Center, Right), and the COLUMN field is calculated automatically. <b>AUTOMATIC</b> mode is local to the entry.</p>
<b>Attributes</b>	<p>Defines the display attributes of the message (<b>normal</b>, <b>flashing</b> or <b>reverse video</b>).</p>
<b>Size</b>	<p>Specifies the format of all the characters of the text or the variable to be displayed (<b>Standard</b> or <b>Double</b>).</p>

**Symbol** Specifies the symbol of the variable associated with the message. It must be defined in the station database. The address associated with the symbol is taken into account automatically when the screen is confirmed.

**Comment** : On a TSX Agent on a FIPIO bus, the variable is read in the master PLC of the bus and not in the PLC sending the function.

**Address** Specifies the address of the variable associated with the message. If the symbol exists in the station database, it is automatically taken into account. Permitted objects are either internal bits (**%Mi**), or internal words (**%MWi**) or internal double words (**%MDi**).

**Comment** Displays the comment of the variable for consultation. It is defined in the station database.

**Format** Specifies the display format of the variable. The only format available is **Numerical**. The dialog box below, accessed via the «Modify» key, enables you to change the default display format, by specifying whether the variable is **signed** and defining the number of characters required before or after the **decimal point**. From the information shown, the software automatically calculates the display format.

Modify Display Format

Variable Type : WORD

Format Type

Numerical

ASCII

Signed

Digits Before the Decimal Point : 3

Digits After the Decimal Point : 2

Display Format : 999.99

OK

Cancel

**Value** Specifies whether, when entering the value associated with the variable, it is performed freely (**Not checked**), or must be monitored within min or max limits (**Limited**) or lastly, is performed by increments (**Increment**).

Example of use

### Request to display a message with entry of a process value

- Programming

**ASK\_MSG**

**Parameters**

Terminal Address : ADR#0.04 : Data to Receive : %MW50 : 2

Data to Transmit : : Report : %MW0 : 4

**Message**

Text : Speed MOTOR = \_\_\_t/mn  Print

**Position**

Mode

Manual  Automatic

Line : 2 Column : 5

Column Alignment

Left  Centred  Right

**Attributes**

Flashing  Reverse Video

**Size**

Standard  Double

**Field**

Symbol : Motor\_2 Address : %MW50

Comment : Speed MOTOR 2

Display Format

999999

**Entry**

Value

Not Checked  Limited  Increment

Increment : 100

- Execution

When launching the ASK\_MSG function in the PLC, the activity bit changes to 1 and the message is displayed on the CCX 17 screen with the current value of « Motor\_2 » in reverse video (authorized entry).

When an operator entry is made on the CCX 17 operator panel, the value entered is written in word %MW50 of the parameter « Data to Receive » and the activity bit changes to 0.

### Warning

For ASK\_MSG, the time-out must be infinite (value 0), otherwise an operator entry arriving after the time-out has no effect.

---

## 2.5 SEND\_ALARM function

---

### 2.5-1 Function

This is used to activate an alarm message present in the PLC on the screen of a CCX 17 operator panel.

---

### 2.5-2 Description

The screenshot shows a dialog box titled "SEND\_ALARM". It is organized into three main sections:

- Parameters:** Includes "Terminal Address" (value 6), "Report" (value 4), and "Data to Transmit".
- Message:** Includes a "Text" input field, a "Print" checkbox, "Alarm Number" (value 900), a "Size" section with "Standard" (selected) and "Double" radio buttons, and an "Overprint" checkbox.
- Field:** Includes "Symbol" and "Address" input fields, a "Comment" input field, and a "Display Format" section with a "Modify..." button.

At the bottom of the dialog are "OK" and "Cancel" buttons.

#### Parameters

These 3 fields are explained in section 3. They are compulsory. For the "**Data to Send**" field, use a %KW<sup>i</sup> type variable and leave the field **Length** blank. In any case, the system automatically calculates the strictly usable size.

#### Message text

Entry zone for alarm message text displayed on the CCX 17 operator panel. It contains a maximum of **40** characters. The "\_" sign (underscore) is reserved by the system to specify the display zone for the variable, if required, associated with the message. To define the position of the variable zone, all that is required is to enter (by convention) an underscore at the required position. The system then calculates the number of "underscores" necessary for the length of display of the variable. Characters whose ASCII code is less than 32 (20h) are not displayed. Conversely, those whose value is greater than 32 (20H) can be displayed either directly or via a combination of the "ALT" and "number" keys.

---

<b>Overprint</b>	Specific to alarm messages. Its confirmation enables the alarm message to overprint as soon as it appears.
<b>Print</b>	Specifies that the message is printed when print is confirmed in the operator panel.
<b>Size</b>	Specifies the format of all the characters of the text or the variable to be displayed ( <b>Standard</b> or <b>Double</b> ).
<b>Alarm number</b>	Defines the identifier of the alarm message. It is between 900 and 999.

**Very important:**

Do not forget to deactivate the message when the alarm disappears in the PLC. (see PANEL\_CMD function)

<b>Symbol</b>	<p>Specifies the symbol of the variable associated with the message. It must be defined in the station database. The address associated with this symbol is taken into account automatically when the screen is confirmed.</p> <p><b>Comment:</b> On a TSX Agent on a FIPIO bus, the variable is read in the master PLC of the bus and not in the PLC sending the function.</p>
<b>Address</b>	<p>Specifies the address of the variable associated with the message. If the symbol exists in the station database, it is automatically taken into account. Permitted objects are either internal bits (<b>%Mi</b>), or internal words (<b>%MWi</b>) or internal double words (<b>%MDi</b>).</p>
<b>Comment</b>	<p>Displays the comment of the variable for consultation. It is defined in the station database.</p>

## Format

Specifies the display format of the variable. The formats available are either **Numerical** or **ASCII**. The dialog box below, accessed via the “Modify” key, enables you to change the default display format, specifying whether the variable is **signed** and defining the number of characters required before or after the **decimal point**. From the information shown, the software calculates the display format automatically.

**Modify Display Format**

Variable Type : WORD

Signed

**Format Type**

Numerical

ASCII

Digits Before the Decimal Point : 3

Digits After the Decimal Point : 2

OK

Cancel

Display Format : 999.99

Alarm messages are always displayed on the second line of the screen. They are time-stamped by the operator panel which synchronizes itself with the PLC's realtime clock.

---

## 2.6 DISPLAY\_MSG function

---

### 2.6-1 Function

This function is used to display a status message contained in the memory of the CCX17 operator panel.

---

### 2.6-2 Description

The screenshot shows a dialog box titled "DISPLAY\_MSG". It is divided into two main sections. The top section, labeled "Parameters", contains three input fields: "Terminal Address" with the value "6", "Data to Transmit" which is empty, and "Report" with the value "4". The bottom section, labeled "Value of Data to Transmit", contains a "Message number" field with a dropdown menu currently showing "1". At the bottom of the dialog are two buttons: "OK" and "Cancel".

#### Parameters

These 3 fields are explained in section 3. They are compulsory. For the field entitled "**Data to Send**", an immediate value cannot be entered directly. In this case, it should be left blank and the following field completed.

#### Message number

This is used to select a message contained in the CCX17 memory **which has already been configured**. The number is between 1 and 300.

---

## 2.7 DISPLAY\_GRP function

---

### 2.7-1 Function

This function is used to display a status message group contained in the memory of the CCX17 operator panel.

D

---

### 2.7-2 Description

The screenshot shows a dialog box titled "DISPLAY\_GRP". It is divided into two main sections. The top section, labeled "Parameters", contains three input fields: "Terminal Address" with the value "6", "Data to Transmit" (empty), and "Report" with the value "4". The bottom section, labeled "Value of Data to Transmit", contains a "Message group number" spin box with the value "1". At the bottom of the dialog are "OK" and "Cancel" buttons.

**Parameters** These 3 fields are explained in section 3. They are compulsory. For the field entitled "**Data to Send**", an immediate value cannot be entered directly. In this case, it should be left blank and the following field completed.

**Message group number** This is used to select a message group number. A group is an association of status messages. In this way, the CCX17 operator panel can display several messages at the same time. This number is between 1 and 100.



---

## 2.8 DISPLAY\_ALARM function

---

### 2.8-1 Function

This function is used to display an alarm message **contained in the memory of the CCX17 operator panel**.

---

### 2.8-2 Description

The screenshot shows a dialog box titled "DISPLAY\_ALARM". It is divided into two sections. The top section, labeled "Parameters", contains three input fields: "Terminal Address" with the value "6", "Data to Transmit" (empty), and "Report" with the value "4". The bottom section, labeled "Value of Data to Transmit", contains one input field: "Alarm message number" with the value "1". At the bottom of the dialog are "OK" and "Cancel" buttons.

#### Parameters

These 3 fields are explained in section 3. They are compulsory. For the field entitled "**Data to Send**", an immediate value cannot be entered directly. In this case, it should be left blank and the following field completed.

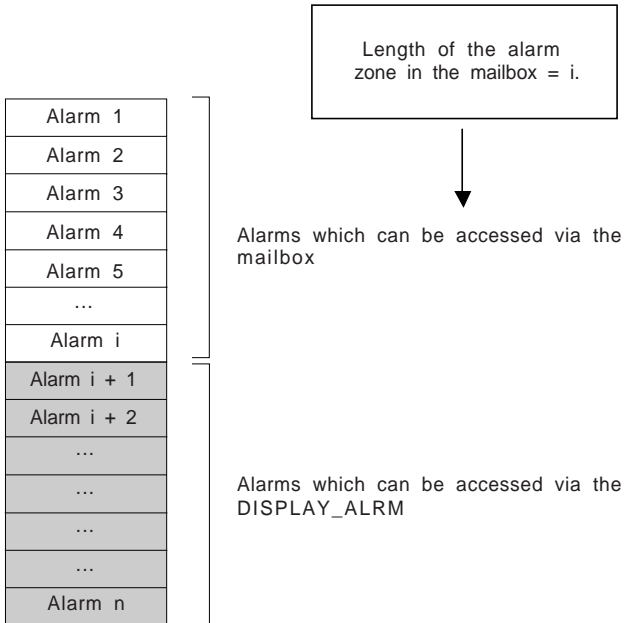
**Alarm message number** This is used to select an alarm message contained in the CCX17 memory which has already been **configured**. The number is between 1 and 300.

#### Very important:

Do not forget to deactivate the message when the alarm disappears (see PANEL\_CMD function)

Only alarms, whose number is **greater than the length of the mailbox**, can be accessed via the Integrated MMI function DISPLAY\_ALARM

D



## 2.9 ASK\_VALUE function

### 2.9-1 Function

This is used to display a status message **contained in the CCX17 memory** of the operator panel. This message contains a variable which can be modified by the operator. The entry is performed in **synchronized** mode. It is valid for a single operator input with each message display. In this case, the value entered is returned to the ASK\_VALUE function.

### 2.9-2 Description

The screenshot shows a dialog box titled "ASK\_VALUE". It has a blue title bar. Below the title bar, there are two main sections. The first section is titled "Parameters" and contains four fields: "Terminal Address" with a value of "6", "Data to Receive" (empty), "Data to Transmit" (empty), and "Report" with a value of "4". The second section is titled "Value of Data to Transmit" and contains a "Status message number" field with a value of "1". At the bottom of the dialog are "OK" and "Cancel" buttons.

#### Parameters

These first 4 fields are explained in section 3. They are compulsory. The **"Data to Receive"** field defines the word (single or double) which receives the value entered by the operator on the CCX 17 operator panel. It must be located in an internal word (%MW). The length of the field should have a value of **at least 2**.

An immediate value cannot be entered directly in the "Data to Send" field. In this case, it should be left blank and the following field completed.

#### Very important :

When using the operator panel, the **"Data to Receive"** parameter contains the value entered. If the status message variable differs from the **Data to Receive**, the status message variable will not be modified by the entry.

---

**Status message number** This is used to select a status message contained in the CCX17 memory which has already been **configured**. The number is between 1 and 300.

**Warning**

D

For ASK\_VALUE, the time-out must be infinite (value 0), otherwise an operator entry arriving after the time-out has no effect.

---

## 2.10 GET\_VALUE function

---

### 2.10-1 Function

This is used to display on the operator panel screen, a status message **contained in the CCX17 memory**. This message contains a variable which can be modified by the operator. The entry is performed in **multiple** mode. The operator can enter several successive values. In this case the PLC program processes the value entered when the variable appears.

---

### 2.10-2 Description

The screenshot shows a dialog box titled "GET\_VALUE". It is divided into two main sections. The top section, labeled "Parameters", contains two rows of input fields. The first row is "Terminal Address" followed by a text box containing "6" and a "Report" field followed by a text box containing "4". The second row is "Data to Transmit" followed by an empty text box. The bottom section, labeled "Value of Data to Transmit", contains a "Status message number" field with a dropdown menu showing "1". At the bottom of the dialog are two buttons: "OK" and "Cancel".

#### Parameters

These first 3 fields are explained in section 3. They are compulsory.

An immediate value cannot be entered directly in the "**Data to Send**" field. In this case, it should be left blank and the following field completed.

**Status message number** This is used to select a status message contained in the CCX17 memory which has already been **configured**. The number is between 1 and 300.

## 2.11 CONTROL\_LEDS function

### 2.11-1 Function

This is used to control the status of the relay (in version 2.1 and later) and of the LEDs on the small indicator bank of the CCX 17 operator panel. It can be used on a CCX17 **with** or **without** application.

### 2.11-2 Description

The screenshot shows a dialog box titled "CONTROL\_LEDS". It is divided into three main sections:

- Parameters:** Contains two input fields. The first is labeled "Terminal Address :" and contains the value "6". The second is labeled "Report :" and contains the value "4". Below these is a field for "Data to Transmit :" which is currently empty.
- Indicator Lamp Status:** This section contains three rows, each representing a different lamp. Each row has four radio button options: "Unchanged", "Off", "Flashing", and "On". In all three rows, the "Unchanged" option is selected.
- Relay Status:** This section contains three radio button options: "Unchanged", "Open", and "Closed". The "Unchanged" option is selected.

At the bottom of the dialog are two buttons: "OK" and "Cancel".

#### Parameters

These 3 fields are explained in section 3. They are compulsory.

For the field entitled “**Data to Send**”, use a **%KWi** type variable and leave the **Length** field blank. In any case, the system automatically calculates the strictly usable size.

#### Indicator lamp status

Defines the possible status allotted to each indicator lamp (**Unchanged, Off, Flashing, On**).

#### Relay status

Defines the relay status (**Unchanged, Open, Closed**) This function is only available with CCX17 version **2.1** or later.

## 2.12 ASSIGN\_KEYS function

### 2.12-1 Function

This is used to configure the command keys of the terminals. They are always associated with the internal bits of the master PLC of the communication. This procedure modifies the configuration of all the command keys of the operator panel. The maximum number of keys which can be configured is 12. Only the keys on the operator panel which is receiving the command are taken into account.

The function can be used on a CCX17 **with** or **without** application.

### 2.12-2 Description

**ASSIGN\_KEYS**

Parameters  
 Terminal address: [ ] : 6 Report: [ ] : 4  
 Data to transmit: [ ] : [ ]

Set by  
 PLC  CCX17

Command key 1  
 Inhibit Action  
 Symbol: [ ]  On Edge  
 Address: [ ]  On State

Command key 2  
 Inhibit Action  
 Symbol: [ ]  On Edge  
 Address: [ ]  On State

Command key 3  
 Inhibit Action  
 Symbol: [ ]  On Edge  
 Address: [ ]  On State

Command key 4  
 Inhibit Action  
 Symbol: [ ]  On Edge  
 Address: [ ]  On State

Display command keys

---

<b>Parameters</b>	These 3 fields are explained in section 3. They are compulsory. For the field entitled " <b>Data to Send</b> ", use a <b>%KWi</b> type variable and leave the field <b>Length</b> blank. In any case, the system automatically calculates the strictly usable size.
<b>Set by</b>	Specifies whether the PLC or CCX17 contains the key configuration data. <b>PLC</b> : the data taken into account is that entered in the " <i>Command key</i> " zones (keys can be configured independently). <b>CCX17</b> : the data taken into account is the application data contained in the operator panel (all keys are reconfigured).
<b>Command key</b>	Specifies the characteristics ( <b>inhibit, address/symbol, action</b> ) associated with each key. The dialog box presents, by default, the programming of the first 4 keys. To access the next keys (in groups of 4), all that is required is to press the « <b>Key .. to ..</b> » button. The dark outline around these buttons indicates that the keys in this group have been configured in this batch.
<b>Inhibit</b>	Disables the status of the key. Its confirmation inhibits the action and address/symbol fields.
<b>Action</b>	Defines the operating mode of the key. Selecting the <b>On Edge</b> mode means that pressing the key sets the associated bit to 1, and that releasing the key sets it to 0. Choosing the <b>On state</b> mode indicates that pressing the key changes the state of the bit. The default value is edge.
<b>Address</b>	Specifies the address of the internal bit <b>%Mi</b> associated with the key. If the symbol associated with this bit exists in the station database, it is automatically taken into account when the address is confirmed.
<b>Symbol</b>	Specifies the symbol associated with the bit. The address associated with this symbol is automatically taken into account.

---



## 2.13 PANEL\_CMD function

### 2.13-1 Function

This function is generic. It can be used on a CCX17 **with** or **without** application. It is used to send various simple commands of the following types to the operator panel :

- clear screen
- delete line
- print log of operator entries
- clear log of operator entries
- print log of alarm messages
- clear log of alarm messages
- cancel alarm number (see **SEND\_ALARM** and **DISPLAY\_ALARM**)

### 2.13-2 Description

**PANEL\_CMD**

**Parameters**

Terminal Address :  Report :

Data to Transmit :  :

**Commands**

**Clear**

Screen  Line Line Number :

**Entry Log**

Print  Clear

**Alarm Log**

Print  Clear

**Alarm Management**

Cancel an Alarm Alarm Number :

---

<b>Parameters</b>	<p>These 3 fields are explained in section 3. They are compulsory.</p> <p>For the field entitled "<b>Data to Send</b>", use a <b>%KWi</b> type variable and leave the field <b>Length</b> blank. In any case, the system automatically calculates the strictly usable size.</p>
<b>Send command</b>	<p>Identifies the type of command. In the case of deleting a line, its number must be specified. To cancel an alarm, the alarm number which corresponds to the identifier between <b>900</b> and <b>999</b> completed when <b>SEND_ALARM</b> is used, or between <b>1</b> and <b>300</b> when <b>DISPLAY_ALARM</b> is used, must be specified.</p>

---

## 2.14 ADJUST function

---

### 2.14-1 Functions

This function is used to adjust (read and write) language objects (one at a time) by controlling internal words in the PLC memory from a CCX 17 operator panel or a MAGELIS terminal.

Language objects which can be adjusted are :

- internal bits (%Mi)
- internal words (%MWi)
- internal double words (%MDi)
- local in-rack I/O (%I, %Q, %IW, %QW, %ID, %QD)
- remote I/O (%I, %Q, %IW, %QW, %ID, %QD)

### Warning

It is strongly recommended to :

- execute only one ADJUST instance per cycle
- execute the ADJUST instance only every n cycles
- configure the ADJUST function with consecutive words in order to optimize reading of the internal words on the CCX 17 operator panel and the MAGELIS terminal.

---

### 2.14-2 Description of the parameters zone

#### Activating the function (EN)

Setting an internal bit (%Mi) or an internal word bit (%MWi:Xj) assigned to the EN parameter to 1 triggers the execution of the ADJUST function.

#### Read / Write (R\_W)

The state of an internal bit (%Mi) or an internal word bit (%MWi:Xj) assigned to the R\_W parameter indicates whether the function should perform a read (bit = 0) or a write (bit = 1) operation.

#### Type of object (TYPE)

The value of the internal word (%MWi) assigned to the TYPE parameter indicates the type of object to be read or written :

0	: %Mi	5	: %IW
1	: %MWi	6	: %QW
2	: %MDi	7	: %ID
3	: %I	8	: %QD
4	: %Q		

---

### Object address (ADR)

The 8-word table (%MWi:8) assigned to the ADR parameter indicates the address of the object to be read or written :

- **Word 0** (rack) number of the rack containing the I/O module (I/O objects)
- **Word 1** (module) number of the position of the module in the rack (I/O objects), address of the processor in rack 0 (FIPIO link)
- **Word 2** (channel) number of the channel in the module (I/O objects), number of the channel of the integrated FIPIO link
- **Word 3** (rank) rank of the object to be read or written (I/O objects), number of the object (internal language objects)
- **Word 4** (connection point) number of the device connection point on the FIPIO bus (I/O objects)
- **Word 5** (number of the device module) number of the FIPIO device module (I/O objects) :  
0 for the base  
1 for the extension
- **Word 6** (channel in the module) number of the channel in the FIPIO module (I/O objects)  
rank of the slave bit on the AS-i bus (I/O objects)
- **Word 7** (slave address) number of the slave on the AS-i and NANET buses (I/O objects)

### Value to be written (VAL)

The double word (%MDi) assigned to the VAL parameter contains the value to be written in the object.

### Set to 1 or Increment (SINC)

The internal bit (%Mi) assigned to the SINC parameter is used, depending on the type of object to be written, to :

- set the value of the bit (%Mi, %Q) to 1
- increment the value of the word or double word (%MWi, %MDi, %QW, %QD) by 1

**Warning** : the R\_W parameter must be set to 1.

### Set to 0 or Decrement (RDEC)

The internal bit (%Mi) assigned to the RDEC parameter is used, depending on the type of object to be written, to :

- set the value of the bit (%Mi, %Q) to 0
- decrement the value of the word or double word (%MWi, %MDi, %QW, %QD) by 1

**Warning** : the R\_W parameter must be set to 1.

### Value of the object read (VRET)

The internal double word (%MDi) assigned to this parameter contains the value of the object which has just been read.

### 2.14-3 Examples

**Example 1 :** Read internal double word %MD12

ADJUST (%MW20:X0,%MW20:X1,%MW21,%MW22:8,%MD30,%MW20:X2,  
%MW20:X3,%MD32,%MW34:24)

Param.	Language object	Value to be entered	Comment
EN	: %MW20:X0	= 1	execution of the ADJUST function
R_W	: %MW20:X1	= 0	read operation
TYPE	: %MW21	= 2	type of object : %MD
ADR	: %MW22:8	= 0	/
		0	/
		0	/
	%MW25	= 12	object number (%MD12)
		0	/
		0	/
		0	/
		0	/
VAL	: %MD30	= 0	/
SINC	: %MW20:X2	= 0	/
RDEC	: %MW20:X3	= 0	/
VRET	: %MD32		value of object read
GEST	: %MW34:24		This parameter acts as a buffer for the transmission and reception of requests. Its size is 24 words.

**Example 2** : Write in-rack output %QW3.2 to value 15

ADJUST (%MW20:X0,%MW20:X1,%MW21,%MW22:8,%MD30,%MW20:X2,  
%MW20:X3,%MD32,%MW34:24)

Param.	Language object		Value to be entered	Comment
EN	: %MW20:X0	=	1	execution of the ADJUST function
R_W	: %MW20:X1	=	1	write operation
TYPE	: %MW21	=	6	type of object : %QW
ADR	: %MW22:8	=	0	rack number
			3	position of the module
			2	channel number
			0	/
			0	/
			0	/
			0	/
			0	/
VAL	: %MD30	=	15	value to be written
SINC	: %MW20:X2	=	0	/
RDEC	: %MW20:X3	=	0	/
VRET	: %MD32			/
GEST	: %MW34:24			This parameter acts as a buffer for the transmission and reception of requests. Its size is 24 words.

**Example 3** : Increment the output on FIPIO : %QW\1.2.12\0.1

ADJUST (%MW20:X0,%MW20:X1,%MW21,%MW22:8,%MD30,%MW20:X2,  
%MW20:X3,%MD32,%MW34:24)

Param.	Language object	Value to be entered	Comment
EN	: %MW20:X0	= 1	execution of the ADJUST function
R_W	: %MW20:X1	= 1	write operation
TYPE	: %MW21	= 6	type of object : %QW
ADR	: %MW22:8	= 0	/
		1	processor address
		2	number of the channel of the integrated FIPIO link
		0	/
		12	connection point number
		0	module number : base
		1	channel number
		0	/
VAL	: %MD30	= 0	/
SINC	: %MW20:X2	= 1	increment the value of the word by 1
RDEC	: %MW20:X3	= 0	/
VRET	: %MD32		/
GEST	: %MW34:24		This parameter acts as a buffer for the transmission and reception of requests. Its size is 24 words.

**Example 4** : Reset the output on the AS-i bus to zero : %Q\105.0\7.2

ADJUST (%MW20:X0,%MW20:X1,%MW21,%MW22:8,%MD30,%MW20:X2,  
%MW20:X3,%MD32,%MW34:24)

Param.	Language object		Value to be entered	Comment
EN	: %MW20:X0	=	1	execution of the ADJUST function
R_W	: %MW20:X1	=	1	write operation
TYPE	: %MW21	=	4	type of object : %Q
ADR	: %MW22:8	=	1	rack number
			5	position of the module
			0	channel number
			0	/
			0	/
			0	/
			2	bit rank (slave I/O)
VAL	: %MD30	=	7	slave number
			0	/
SINC	: %MW20:X2	=	0	/
RDEC	: %MW20:X3	=	1	reset output
VRET	: %MD32			/
GEST	: %MW34:24			This parameter acts as a buffer for the transmission and reception of requests. Its size is 24 words.



**Example 5** : Decrement word %QW4.0\2.1 (remote Nano PLC)


ADJUST (%MW20:X0,%MW20:X1,%MW21,%MW22:8,%MD30,%MW20:X2,  
%MW20:X3,%MD32,%MW34:24)

Param.	Language object		Value to be entered	Comment
EN	: %MW20:X0	=	1	execution of the ADJUST function
R_W	: %MW20:X1	=	1	write operation
TYPE	: %MW21	=	6	type of object : %QW
ADR	: %MW22:8	=	0	/
			4	position of the module
			0	channel number
			1	rank of the NANET object (word number)
			0	/
			0	/
			0	/
			2	slave number
VAL	: %MD30	=	0	/
SINC	: %MW20:X2	=	0	/
RDEC	: %MW20:X3	=	1	decrement the value of the word by 1
VRET	: %MD32			/
GEST	: %MW34:24			This parameter acts as a buffer for the transmission and reception of requests. Its size is 24 words.

---

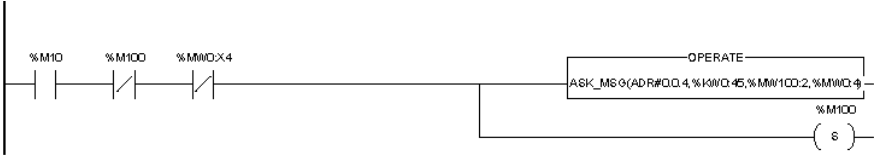
## 2.15 Station documentation file

---

The PL7 documentation file, which can be accessed by the **Application/Documentation** command or the  icon, is used to display the contents of each **Integrated MMI** function. It collects all the elements entered after the **“Details”** button is pressed.

D

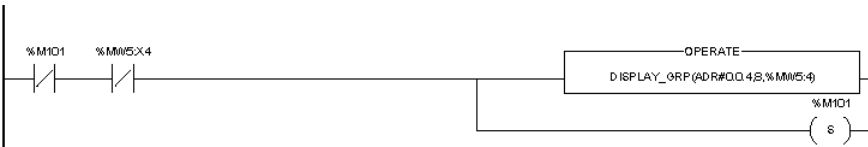
The application file appears as follows :



### Function parameters:

ASK\_MSG(ADR#0.0.4,%KW0:45,%MW100:2,%MW0:4)

Text	: Motorspeed=_____
Field	: None
Line	: 2
Column	: 6
Justification	: None
Print	: NO
Videoattributes	: Flashing
Font	: Standard



### Function parameters:

DISPLAY\_GRP(ADR#0.0.4,8,%MW5:4)

Message group number	: 8
----------------------	-----

### 3.1 General

The Integrated MMI functions form part of the procedures category. They do not return any values but have several parameters which must be entered.

The functions use 3 types of parameters :

- read only (**IN**), taken into account when execution of the function commences
- write only (**OUT**), set when execution of the function ends
- read/write (**IN/OUT**), the contents of which are taken into account when execution of the function commences and are then updated by the results of the function

A function is usually called as follows :

**Integrated MMI function (Operator panel address, Data to Send, Data to Receive, Report)**

## 3.2 Operator panel address

This parameter shows the destination address of the exchange. It can be located in internal words ( %MW ) or internal constants ( %KW ), or passed directly as an immediate value in the form of an operator **ADR#**.

Within the framework of MMI functions, the syntax authorized for ADR# is as follows :

### UNITELWAY addressing

- **ADR#**{<Network>.<Station>}<rack.module>.<channel>.<entity>
- **ADR#**{<Network>.<Station>}< rack.module>.<channel>.**SYS**
- **ADR#**< rack.module>.<channel>.<entity>
- **ADR#**< rack.module>.<channel> . **SYS**

The **SYS** keyword (value 254) corresponds to addressing the system channel (UNI-TE server) of a communication channel.

### FIPIO addressing

- **ADR#**{<Network>.<Station>}\< rack.module>.<channel>.<connection point >\ **SYS**
- **ADR#**\< rack.module>.<channel>.<connection point>\ **SYS**

Operation of the CCX 17 operator panel only authorizes one **intra-station** address. Consequently, for MMI functions, the {Network.Station} pair takes the value **{0.254}**.

ADR# is always classed as a table of 6 consecutive words. The coding of this operator for Integrated MMI functions is structured as follows :

Word number	High order	Low order
%MWi / %KW <sub>i</sub>	6 (UNITELWAY) 7 (FIPIO)	0
%MW <sub>i+1</sub> / %KW <sub>i+1</sub>	254 (station)	0 (network)
%MW <sub>i+2</sub> / %KW <sub>i+2</sub>	module	rack
%MW <sub>i+3</sub> / %KW <sub>i+3</sub>	entity / SYS/ connection point for FIPIO	channel
%MW <sub>i+4</sub> / %KW <sub>i+4</sub>	0	SYS (FIPIO)
%MW <sub>i+5</sub> / %KW <sub>i+5</sub>	0	0

### Examples of use

#### 1. Addressing a CCX 17 connected directly via the front panel of the TSX Micro (UniTelway), with the CCX 17 being at the UTW slave addresses 4-5

- MMI\_Function ( ADR#{0.254} 0.0.4 , etc. )
- %MW10:6 := ADR#0.0.4  
MMI\_Function ( %MW10:6 , etc. )  
Writing the {0.254} field is optional and corresponds to the default value assigned to this field.

- MMI\_Function ( %KW0:6 , etc. )

In this case, the %KWO:6 table must first be initialized under the Data editor, successively assigning the %KWi which make it up:

%KW0	<b>16#600</b>	<b>fixed value</b>
%KW1	<b>16#FE00</b>	<b>fixed value</b>
%KW2	<b>16#0000</b>	<module.rack>
%KW3	<b>16#0400</b>	<entity/SYS><channel>
%KW4	<b>0</b>	<b>fixed value</b>
%KW5	<b>0</b>	<b>fixed value</b>

## 2 Addressing a CCX 17 to FIPIO connection point 7

- MMI\_Function ( ADR#\0.1.7\SYS , .... )

- %MW10:6 := ADR#\0.1.7\SYS  
MMI\_Function ( %MW10:6 , .... )

- MMI\_Function ( %KW0:6 , .... )

In this case, the %KWO:6 table must first be initialized under the Data editor, successively assigning the %KWi which make it up :

%KW0	<b>16#700</b>	<b>FIPIO fixed value</b>
%KW1	<b>16#FE00</b>	<b>fixed value</b>
%KW2	<b>16#0000</b>	<module.rack>
%KW3	<b>16#0701</b>	<connection point><channel>
%KW4	<b>16#00FE</b>	<b>FIPIO fixed value</b>
%KW5	<b>0</b>	<b>fixed value</b>

## 3 Addressing a CCX 17 whose topological address is x.y.z with $x=x_1*100+x_2$ where $x_1$ represents the rack and $x_2$ the module.

- MMI\_Function ( ADR#\{0.254\} x.y.z , .... )

- %MW10:6 := ADR#\x.y.z  
MMI\_Function ( %MW10:6 , .... )

- MMI\_Function ( %KW0:6 , .... )

In this case, the %KWO:6 table must first be initialized under the Data editor, successively assigning the %KWi which make it up :

%KW0	<b>16#600</b>	<b>fixed value</b>
%KW1	<b>16#FE00</b>	<b>fixed value</b>
%KW2	<b>16#x2x1</b>	<module.rack>
%KW3	<b>16#zy</b>	<entity/SYS><channel>
%KW4	<b>0</b>	<b>fixed value</b>
%KW5	<b>0</b>	<b>fixed value</b>

Warning, coding performed in hexadecimal format.

---

### 3.3 Data to Send

---

This data must be located in internal words (%MW) or internal constants (%KW) for functions intended for a CCX17 **without** application. For those intended for a CCX17 **containing an application**, there is the added possibility of identifying it by **an immediate integer value**. They are specific to each type of MMI function. They are explained in the sections describing each function.

For information, their composition is as follows :

1. For a function dedicated for a **CCX 17 without application** :

- the first word contains a value marker 16#CC17. It has a dual role :
  - It enables the help screen to identify a correct message and to redisplay the values in the entry screen in order to modify or display the default values.
  - It enables the function, whilst executing, to check that the table received does contain a message for a CCX 17. In fact, it is possible to call an Integrated MMI function in a program without passing through the help/control screens. In the case of an unmarked message, the function can return a fault to the application immediately without sending suspect data to the terminal.
- the second word contains a command number
- the third word contains the length of the next string, and is expressed in bytes
- the next words contain the data to be sent to the terminal

2. For a function dedicated for a **CCX17 with application** :

- the first word contains a command number
- the second word contains the data to be sent to the operator panel

#### Rule for use

A priori, for reasons of efficiency, it is better to program the <Data to Send> parameter in %KW<sub>i</sub> form, not forgetting despite everything, to check that there is no zone overlap between each MMI function message.

In fact, by programming the parameter in this way, the software automatically initializes the data zone with the appropriate values.

Entering the %MW prohibits access to all the fields in the dialog box. It is therefore up to the user to transfer, manually or by program, the value of the constants to the words which he wishes to use. The format of the messages is described in section 5.2. (See also the PL7-MMI 17 and PL7-M17 (OS2) software documentation.)

### 3.4 Data to Receive

This must be located in an internal word (%MW). It is specific to the **ASK\_MSG** and **ASK\_VALUE** functions and is described in the section on these functions. (see Section 2.4.2 and 2.9.2)

### 3.5 Report

This is common to and required by all asynchronous communication functions. It is updated by the system, with the exception of the Time\_Out which must be indicated (100 ms unit) before the function is executed. It must be located in internal words (%MW). It is made up of :

- a parameter providing information on the function activity
- a parameter specifying the exchange number which identifies the current transaction (useful when using the CANCEL Communication function)
- a parameter containing the exchange report which is split into two return codes, one specifying the level of the communication and the other the level of the operation
- a time-out parameter enabling the response time to be checked
- a length parameter storing the number of bytes to be sent and/or the number of bytes received

These parameters require a table of 4 consecutive words (%MWi:4) with the following structure :

Word number	High order	Low order
%MWi	Exchange number	Bit 0 : activity bit
%MWi+1	Operation report	Exchange Report
%MWi+2	Time-out	
%MWi+3	Length	

#### Activity bit (%MWi:X0)

This indicates the execution state of a function. It is set to 1 when the function is launched and returns to 0 when the response is received, or at the end of the time-out, or when the operation is canceled (CANCEL function).

#### Exchange number

When a function is transmitted, the system automatically assigns it a number enabling the exchange to be identified. This number is used as a reference to stop the exchange in progress if necessary (use of CANCEL).

## Exchange report

This is common to all functions (except ADJUST). It gives information about the transaction at the communication level. It is significant when the value of the activity bit changes from 1 to 0. The various values of this report are shown in the table below :

Exchange report	Meaning (the codes are in hexadecimal)
0	Correct exchange
16 # 1	Exchange stopped on Time-out
16 # 2	Exchange stopped on user request (CANCEL function)
16 # 3	Incorrect address format (length is other than 6)
16 # 4	Incorrect Destination Address (addressing prohibited for CCX 17 eg : broadcast addressing)
16 # 5	Incorrect Report (length is other than 4)
16 # 6	Incorrect Specific Parameters (particularly with regard to the data to send )
16 # 7	No destination (connection problem)
16 # 9	Reception buffer too small
16 # A	Transmission buffer too small
16 # B	No System Resources (Communication saturation)
16 # 14	Negative response from CCX 17 (it rejects the command)
16 # FF	Message refused (the CCX 17 cannot process the message)

### • ADJUST function

Exchange report	Meaning (the codes are in hexadecimal)
0	Correct exchange
16 # 6	Incorrect Specific Parameters
16 # 14	Negative response from the PLC



## Operation report

This specifies the result of the operation on the remote application. It is significant only if the exchange report has the values :

- **0 (correct exchange)**, the operation report has the value 0.
- **6 (incorrect specific parameters) and 20 (negative response from CCX 17)**. In these two cases, a refusal code is stored in this field (see table below).
- **In other cases**, the operation report has the value 0.

Exchange report	Operation Report	Meaning (the codes are in hexadecimal)
0	0	Generic positive result
16#6	16#64	Broadcast address (ALL) prohibited
	16#65	Pair {Network.Station} other than {0.254}
	16#66	Data to send does not have the 16#CC17 marker
	16#67	Size of Data to send incorrect
	16#68	Invalid response from CCX 17
	16#69	«Data to Receive» not long enough
	16#FF	CCX 17 link inoperative
16#14	16#1	Command not recognized
	16#2	Command buffer overflow
	16#4	Command smaller than minimum required
	16#8	Command refused because application transfer in progress
	16#20	Incorrect data

### • ADJUST function

The operation report is only significant if the exchange report has the following values :

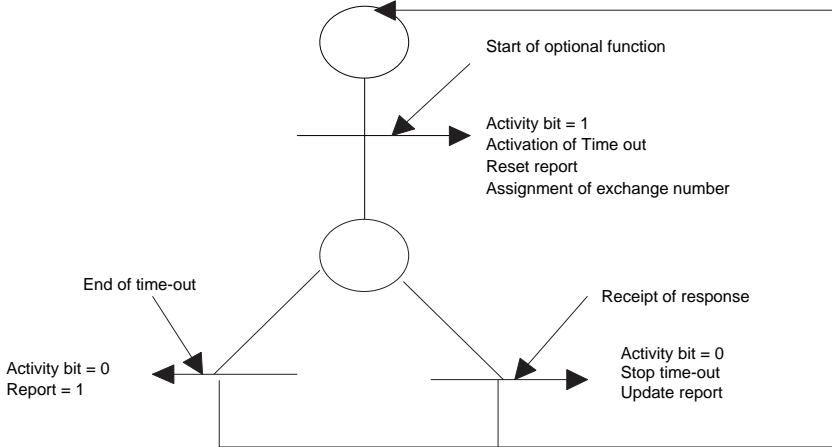
- 0 (correct exchange) : the operation report has the value 0.
- 6 (incorrect specific parameters). In this case, a refusal code is stored in this field (see table below).
- In other cases, the operation report has the value 0.

Exchange report	Operation report	Meaning (the codes are in hexadecimal)
0	0	Generic positive result
16#6	16#10	Number of management words less than 24
	16#11	Type of object to be read not found (greater than 8)
	16#12	Inconsistency between RDEC and SINC bits
	16#13	Value to be written incorrect
16#14	16#14	Object cannot be accessed
	16#15	System error

---

## Time-out

This determines the maximum wait time for the response. The unit of time is expressed in 100 ms. The value 0 means an infinite wait (value required for the ASK\_MSG and ASK\_VALUE functions). In this case, it is advisable to use the CANCEL function. If the time-out has elapsed, the exchange ends with an error report (value 1). In the same way, the arrival of a response after the end of the time-out is refused by the system.



## Length

This is expressed in words and is used to store the number of words received after the reception of a message for the **ASK\_MSG** and **ASK\_VALUE** functions (it is shown by the parameter Data to receive). In all other cases, its value is 0.

## 4 Precautions when using MMI and example of application

### 4.1 Precautions when using MMI

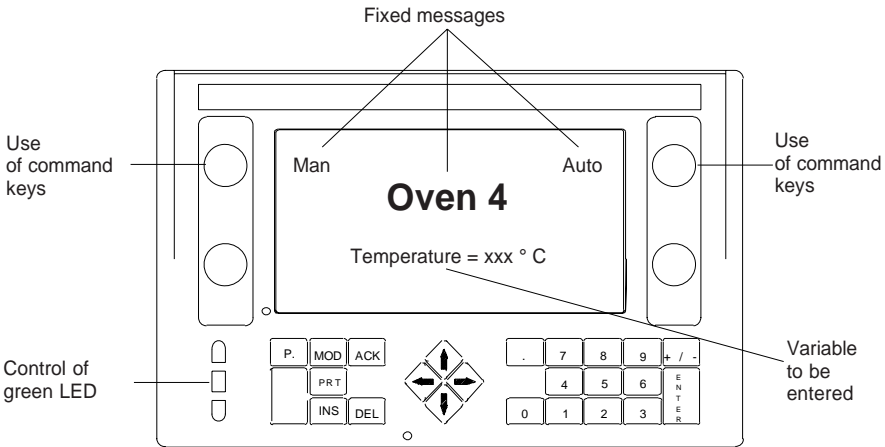
---

- Do not forget to initialize the Time-out parameter (%MWi+2) before launching the MMI function. In the case of the ASK\_MSG or ASK\_VALUE functions, 0 is compulsory.
- If the same word is used to save each function report, it is necessary to test the activity bit (%MWi:X0) at value 0 before launching another function.
- Synchronize the launching of the various Integrated MMI functions in order to avoid saturating the CCX 17 operator panel command buffer.
- Remember to size the %KW<sub>i</sub> internal constants appropriately to display the data to send.
- Do not hesitate to retain some margin in the allocation of the data to send (%KW<sub>i</sub>:n). In fact, if you need to modify the texts, there will be no problem of overlapping of the various references of %KW<sub>i</sub>:n.  
The maximum size for the Integrated MMI functions is 47 words.
- Warning, in online modification mode, it is not possible to create an Integrated MMI function if the application located in the PLC does not already possess a copy of this function.
- On power break or loss of communication, it is up to the application to return the CCX 17 to a consistent state (assignment of keys (see example) and messages on screen).
- Canceling modifications or deletion of a rung or a sequence (List) or statement (Structured Text) does not cancel the initialization of %KW<sub>i</sub> variables.

## 4.2 Example

The example used in the rest of this section depicts the installation of an MMI application linked to a T CCX 1720 LW operator panel connected on UNI-TELWAY.

### Description of the application



### Assigning PLC variables

- %M0 = MAN key (green CCX 17 LED flashes)
- %M2 = AUTO key (green CCX 17 LED lights)
- %MW10 = Temperature (value read and written)

The CCX 17 operator panel is connected directly to the front panel of the TSX Micro at UNI-TELWAY addresses 4-5. Its topological address is ADR#0.0.4.

### Details of the application

The application includes 3 status messages, an alarm message, management of 2 command keys and modification of the status of the green LED on the operator panel.

### Configuring the status messages

They are made up as follows :

- OVEN 4 (status message 1)
- Man (status message 2)
- Auto (status message 2) on the same line as the Man message
- Temperature = xxx °C (status message 3, read and write word %MW10 in decimal format), with value entry. Entry is incremental (increment of 50)

### Processing status messages

Status messages 1 and 2 are displayed by default with the option of clearing the screen. Status message 3 is displayed on user request (setting bit %M10 to 1)

### Configuring the alarm message

The application manages an alarm which is displayed on the screen when it appears.

- a message including the display of a variable: «Oven 4 overheat = xxx °C» (display of word %MW10 in decimal)

### Management of the alarm message

Setting bit %M12 to 1 (set if the temperature exceeds the value 500) triggers the display and printing of alarm message 1 (Oven 4 temp: xxx °C).

The “Cancel alarm number” command is activated after the alarm message and on user acknowledgment (canceling alarm and setting the %M20 bit to 1).

### Operation of the CCX17 requires this.

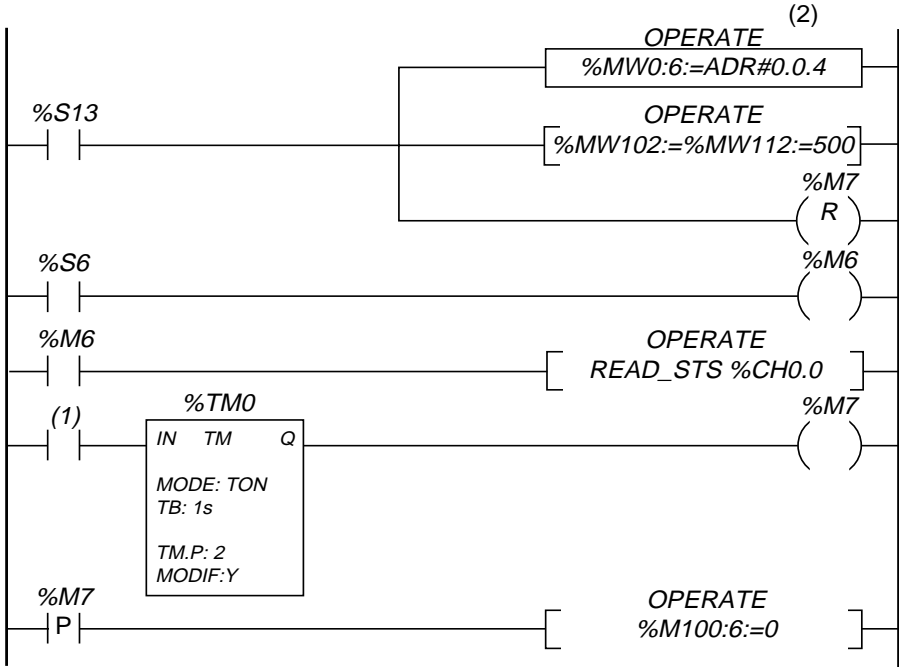
### Configuring the command keys

Command keys 1 and 2 are configured for edge type operation.

One controls internal bit %M0 (Man key), the other internal bit %M2 (Auto key).

**The program corresponding to this application is given below :**

(\*INIT operator panel adr and 2 time-outs assigned, one for STATUS messages and the other for ALARM message\*)



(1):%MW0.0.4:X11

This rung is used to initialize variables when the PLC is set to RUN. In addition, it is used to monitor the presence of the CCX 17 operator panel in order to reinitialize its screen upon disconnection/reconnection. This monitoring is carried out in two phases :

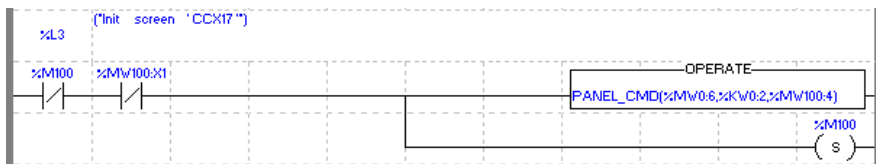
- Explicit exchange on the terminal port in order to update the information on the presence of the UNI-TELWAY slaves (every second, system bits %S6).
- Triggering screen initialization (%M100:6:=0) upon reappearance of the CCX 17 operator panel (rising edge on %MW0.0.4.X11 which represents the presence of slave N°4).

**Note**

(2) %MW0:6:=ADR#0.0.4

- <=> %MW0 = 0600H
- %MW1 = FE00H
- %MW2 = 0
- %MW3 = 0400H
- %MW4 = 0
- %MW5 = 0

This rung will clear the CCX17 operator panel screen completely using the PANEL\_CMD function.



D

**Note :**

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the PANEL\_CMD function has the value %MW100:X4.

---

This rung simultaneously **displays status message 1 «OVEN 4»** and **status message 2 «Man...Auto»** on the screen of the CCX 17 operator panel.

This display can be performed in two different ways :

1. either the CCX17 **does not contain** an application
2. or the CCX17 **contains** an application

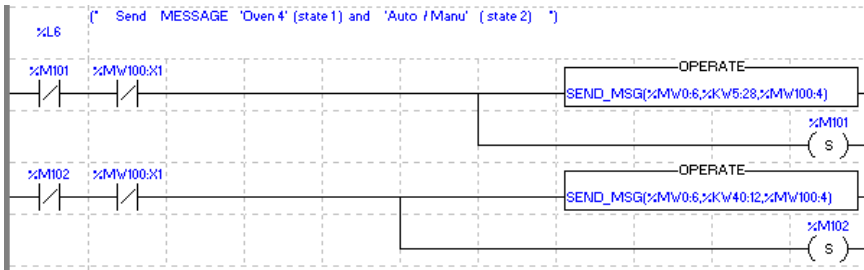
### **Case 1 : CCX17 without application**

The programming principle is as follows :

- describes the enable conditions (no Integrated MMI function being executed, SEND\_MSG function should only be launched once and the functions synchronized)
- launches display of the fixed message
- stores its execution

#### **Note :**

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the SEND\_MSG functions has the value %MW100:X4.





**SEND\_MSG**

Parameters  
Terminal:  :  Report:  :   
Data to Send:  :

Message  
Text:    Print

Position  
Mode:  Manual  Automatic  
Line:  Column:   
Column Alignment:  Left  Center  Right

Attributes  
 Blinking  Reverse Video

Font  
 Single  Double

Clear  
 None  Line  Screen

Field  
Type of Field:  None  Address  Date  Time  
Symbol:  Address:   Refresh  
Comment:   
Display Format:

**SEND\_MSG**

Parameters  
Terminal:  :  Report:  :   
Data to Send:  :

Message  
Text:   Print

Position  
Mode:  Manual  Automatic  
Line:  Column:   
Column Alignment:  Left  Center  Right

Attributes  
 Blinking  Reverse Video

Font  
 Single  Double

Clear  
 None  Line  Screen

Field  
Type of Field:  None  Address  Date  Time  
Symbol:  Address:   Refresh  
Comment:   
Display Format:

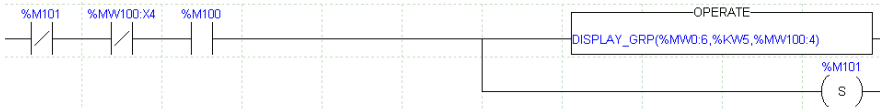
## Case 2 : CCX17 with application

The programming principle is as follows :

- describes the enable conditions (no Integrated MMI function being executed, DISPLAY\_GRP function should only be launched once and the functions synchronized)
- launches display of the fixed messages group
- stores its execution

### Note :

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the DISPLAY\_GRP functions has the value %MW100:X4.



**DISPLAY\_GRP**

**Parameters**

Terminal Address :  :     Report :     :

Data to Transmit :

**Value of Data to Transmit**

Message group number :

This rung displays status message 3 «Temperature = xxx °C» on the screen of the CCX 17 operator panel.

This display can be performed in two different ways :

1. either the CCX17 **does not contain** an application
2. or the CCX17 **contains** an application

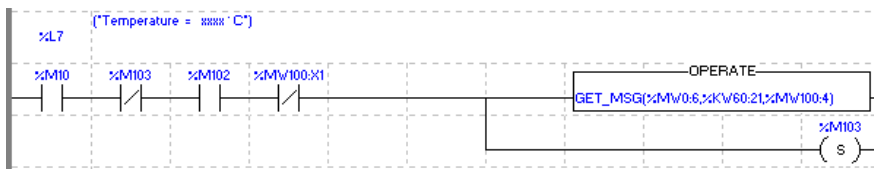
**Case 1 : CCX17 without application**

The programming principle is as follows :

- describes the enable conditions (no Integrated MMI function being executed, GET\_MSG function should only be launched once and the functions synchronized)
- launches display of the message on user request (%M10)
- stores its execution

**Note :**

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the GET\_MSG function has the value %MW100:X4.



**GET\_MSG**

**Parameters**

Terminal:  :       Report:  :

Data to Send:  :

---

**Message**

Text:        Print

**Position**

Line:       Column:

Mode:  Manual       Automatic

Column Alignment:  Left       Center       Right

**Attributes**

Blinking       Reverse Video

**Font**

Single       Double

---

**Field**

Symbol:       Address:        Refresh

Comment:

Display Format:

---

**Enter**

Value:  Not checked       Limited       Increment

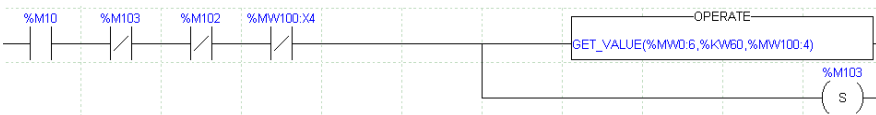
## Case 2 : CCX17 with application

the programming principle is as follows :

- describes the enable conditions (no Integrated MMI function being executed, GET\_VALUE function should only be launched once and the functions synchronized)
- launches display of the message on user request (%M10)
- stores its execution

### Note :

The unreadable contact variable corresponds to the activity bit suffixed with :XO and the report parameter of the GET\_VALUE has the value %MW100:X4.



### GET\_VALUE

**Parameters**

Terminal Address :  :  Report :  :

Data to Transmit :

**Value of Data to Transmit**

Status message number :

This rung displays alarm message «Oven overheat = xxx °C» on the screen of the CCX 17 operator panel.

This display can be performed in two different ways :

1. either the CCX17 **does not contain** an application
2. or the CCX17 **contains** an application

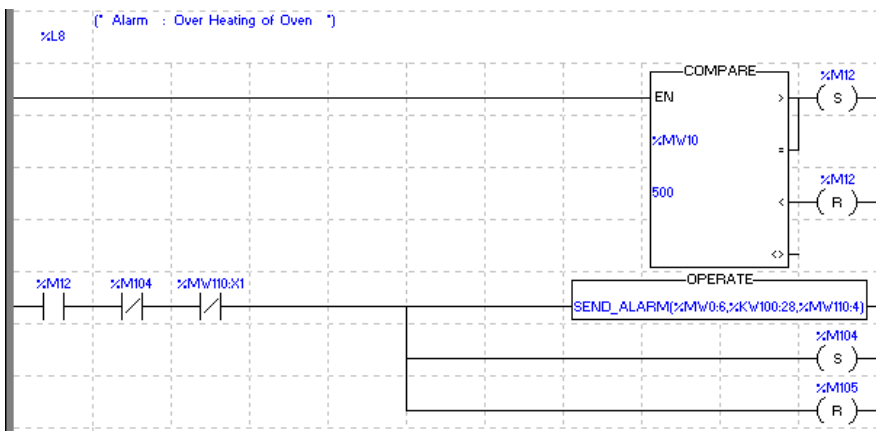
**Case 1 : CCX17 without application**

the programming principle is as follows :

- describes the enable conditions (if the critical stage is passed, no Integrated MMI function being executed and SEND\_ALARM function should only be launched once)
- launches display of the alarm message
- stores its execution

**Note :**

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the SEND\_ALARM function has the value %MW110:X4.



**SEND\_ALARM**

Parameters

Terminal:  :  Report:  :

Data to Send:  :

Message

Text:   Print

Alarm Number:  Font:  Single  Double  Display by Superimposition

Field

Symbol:  Variable:   Refresh

Comment:

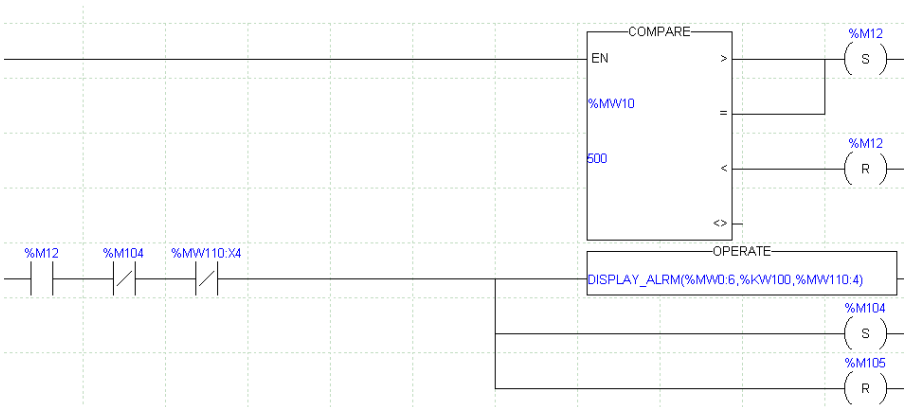
Display Format:



**Case 2 : CCX 17 with application**

The programming principle is as follows :

- describes the enable conditions (if the critical stage is passed, no Integrated MMI function being executed and the DISPLAY\_ALARM function should only be launched once)
- launches display of the alarm message
- stores its execution



**DISPLAY\_ALARM**

**Parameters**

Terminal Address :  :     Report :     :

Data to Transmit :

**Value of Data to Transmit**

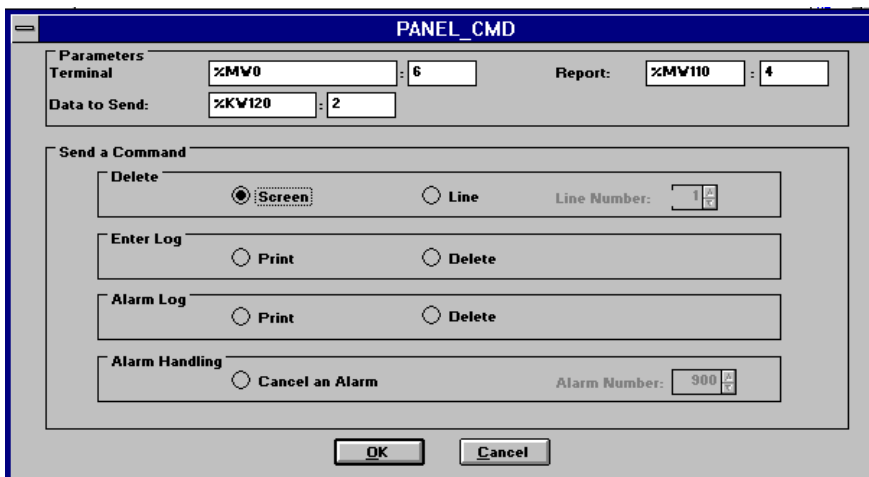
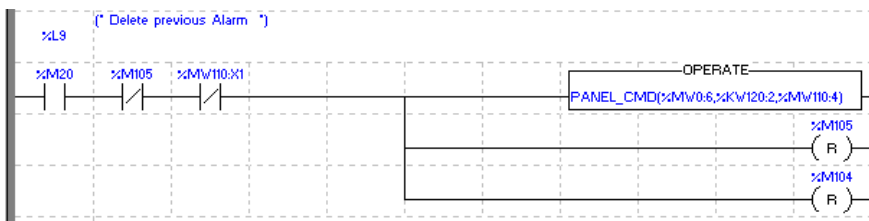
Alarm message number :

This rung **cancels the previous alarm if it has been triggered** :

- describes the enable conditions
- on fault acknowledgment (%M20) launches the PANEL\_CMD function with alarm number at **900** in the case of **CCX17 without application** and at **1** in the case of **CCX17 with application** but **not configured**, ie. no mailbox field
- stores its execution and synchronizes the SEND\_ALARM and PANEL\_CMD functions

**Note :**

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the PANEL\_CMD function has the value %MW110:X4.

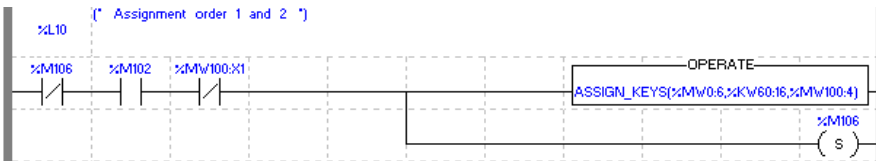


This rung **assigns command keys 1 and 2** of the CCX 17 operator panel in 3 phases :

- describes the enable conditions (no Integrated MMI function being executed, ASSIGN\_KEYS function should only be launched once and synchronized with the other functions)
- launches the execution
- stores its execution

#### Note :

The unreadable contact variable corresponds to the activity bit suffixed with :X0 and the report parameter of the ASSIGN\_KEYS function has the value %MW100:X4.



### ASSIGN\_KEYS

<b>Parameters</b>	
Terminal Address : %MW0 : 6	Report : %MW100 : 4
Data to Transmit : %KW200 : 16	

<b>Command Key 1</b> <input type="checkbox"/> Inhibit <input checked="" type="radio"/> PLC Symbol : <input type="text" value="Manu"/> Address : <input type="text" value="%M0"/> <input checked="" type="radio"/> On Edge <input type="radio"/> On State	<b>Command Key 2</b> <input type="checkbox"/> Inhibit <input type="text" value="Auto"/> Symbol : <input type="text"/> Address : <input type="text" value="%M1"/> <input checked="" type="radio"/> On Edge <input type="radio"/> On State
<b>Command Key 3</b> <input type="checkbox"/> Inhibit Symbol : <input type="text"/> Address : <input type="text"/> <input checked="" type="radio"/> On Edge <input type="radio"/> On State	<b>Command Key 4</b> <input type="checkbox"/> Inhibit Symbol : <input type="text"/> Address : <input type="text"/> <input checked="" type="radio"/> On Edge <input type="radio"/> On State

**Display Command Keys**  

Keys 1 to 4...
Keys 5 to 8...
Keys 9 to 12...

OK
Cancel

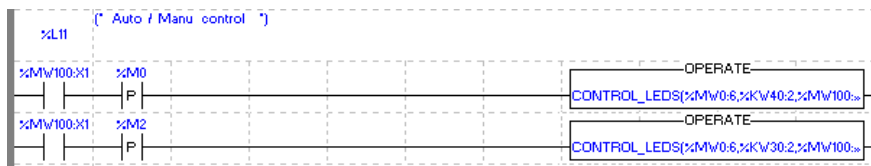
In the case of a CCX 17 with application, the assignment of command keys can be included in the CCX 17 application and this rung thus becomes useless.



This rung simultaneously controls the green LED in flashing mode on detection of the rising edge of the Man variable, and the green LED in On mode on detection of the rising edge of the Auto variable.

**Note:**

The report parameter of the CONTROL\_LEDS functions has the value %MW100:X4.






**CONTROL\_LEDS**

**Parameters**


Terminal Address :  :  Report :  :

Data to Transmit :  :

Indicator Lamp Status

	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Off	<input type="radio"/> Flashing	<input type="radio"/> On
	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Off	<input type="radio"/> Flashing	<input type="radio"/> On
	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Off	<input type="radio"/> Flashing	<input type="radio"/> On

Relay Status




	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Open	<input type="radio"/> Closed
---	--	----------------------------	------------------------------

D


**CONTROL\_LEDS**

**Parameters**  
Terminal Address :  :  Report :  :   
Data to Transmit :  :

**Indicator Lamp Status**

	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Off	<input type="radio"/> Flashing	<input type="radio"/> On
	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Off	<input type="radio"/> Flashing	<input type="radio"/> On
	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Off	<input type="radio"/> Flashing	<input type="radio"/> On

**Relay Status**

	<input checked="" type="radio"/> Unchanged	<input type="radio"/> Open	<input type="radio"/> Closed
---	--	----------------------------	------------------------------

## 5.1 Description of interface parameters

### 1 Operator panel address (in %MWi:6, %KWi:6 or ADR# format)

**ADR#{0.254}<rack.module>.<channel>.<entity>**

**ADR#{0.254}<rack.module>.<channel>.**SYS****

**ADR#<rack.module>.<channel>.<entity>**

**ADR#<rack.module>.<channel> **SYS****

**ADR#{0.254}\<rack.module>.<channel>.<connection point>**ISYS****

**ADR#<rack.module>.<channel>.<connection point>**ISYS****

ADR# is classed as a table of 6 consecutive words whose coding is :

Word number	High order	Low order
%MWi/%KWi	6(UNITELWAY)7(FIPIO)	0
%MWi+1/%KWi+1	(station)	(network)
%MWi+2/%KWi+2	module	rack
%MWi+3/%KWi+3	entity / SYS / FIPIO connection point	channel
%MWi+4/%KWi+4	0	SYS (FIPIO)
%MWi+5/%KWi+5	0	0

### 2 Data to Send and Data to Receive

These are located in the **%MWi** and **%KWi** for the first (also in an immediate whole **value** for functions dedicated to a CCX17 **with** application) and in the **%MWi** for the second.

### 3 Report

Requires a table of 4 consecutive words (%MWi:4)

Word number	High order	Low order
%MWi	Exchange number	Bit 0 : activity bit
%MWi+1	Operation report	Exchangereport
%MWi+2	Time-out in 100ms	
%MWi+3	Length	

Before launching the function, the Time\_out in **bold** should be initialized and the activity bit in italics (%MWi:X0) should be tested (value 0). For the ASK\_MSG and ASK\_VALUE functions, the value of the Time-out must be set to 0.

## Exchange report

Exchange report	Meaning
16 # 1,2	Exchange stopped on Time-out/CANCEL function
16 # 3,4,5,6	Incorrect Interface Parameters (6 -> Data to Send)
16 # 7	Destination missing (connection)
16 # 9,A	Reception/Transmission Buffer too small
16 # B	No System Resources (Communication saturated)
16 # 14	Negative response from CCX 17
16 # FF	Message refused (CCX 17 cannot process it due to its state)

## Operation report

Exchange Report	Operation Report	Meaning
16 # 6	16 # 64	Broadcast address (ALL) prohibited
	16 # 65	Pair {network.station} other than {0.254}
	16 # 66	Data to Send does not have 16#CC17 marker
	16 # 67/69	Data to Send incorrect size/Data to Receive too small
	16 # 68	Invalid response from CCX 17
	16 # FF	CCX 17 link inoperative
16 # 14	16 # 1	Command not recognized
	16 # 2	Command buffer overflow
	16 # 4	Command size smaller than minimum required
	16 # 8	Command refused because application transfer in progress
	16 # 20	Incorrect data

---

## 5.2 Description of coding «Data to Send» parameter of Integrated MMI functions

---

### 5.2-1 Displaying a PLC status message : SEND\_MSG function

It is possible to build status messages from the PLC application and to send them via the (%MWi) internal words in order to display them on the CCX17 operator panel screen. This is the role of the SEND\_MSG function.

For information purposes, the “**Data to Send**” parameter requires a maximum of **47** words. It is composed as follows :

- the **first word** contains a value marker **16#CC17**
- the **second word** contains the value 0
- the **third word** contains the length in bytes of the next words zone
- the **next words** must contain the message text to send, including the « underscores » representing the characters expected when the variable is displayed. This text has a maximum length of 40 characters.

If the text is made up of an odd number of characters, the last byte has a value of 0, if the text has an even length and fewer than 39 characters, the last word must contain the value 0.

- one **word** containing the line number where the message must be displayed
- one **word** containing the column number where the beginning of the message must be displayed
- a zone of **two words** (four characters) containing the characteristics of the message, and structured as follows:
  - the first character (in upper case) corresponds to the video attribute (**B** = flashing, **R** = reverse video, **A** = flashing and reverse video, **N** = no attribute)
  - the second character (in upper case) corresponds to the font size of the characters (**S** = single, **D** = double height and double length)
  - the second word (in upper case) corresponds to the print option (**Y followed by a space** = yes, **N followed by a space** = no)

If the user does not want to display the variable, the next word must be at 0 (in this case the rest of the parameters are ignored), if not the following parameters must be added:

- one **word** containing the position of the variable to display, counted in number of characters in relation to the beginning of the message
- one **word** containing the number of characters to display for the variable
- one **word** containing an additional command :
  - 0 : no command
  - 1 : clear screen
  - 2 : delete line before display
- one **word** containing the value 16#0030

- **two words** (in upper case) containing the type of entry field :
  - '**BIT** ' (concluded with a space) for a bit type
  - '**ANA** ' (concluded with a space) for a word type
  - '**LNG** ' (concluded with a space) for a double word type
  - '**DAY** ' (concluded with a space) for a date type
  - '**HOU** ' (concluded with a space) for an hour type
- one **word** containing the value 0
- one **word** (in upper case) containing the type of variable to display
  - '**B** ' (concluded with a space) for a bit type
  - '**W** ' (concluded with a space) for a word type
  - '**DW** ' for a double word type
 for a **date** or hour type, it contains the value **0**
- one **word** containing the **index of the address** of the variable to display for a **bit, word** or **double word** type, or containing the value **-1** for a **date** or **time** type
- **two words** (in upper case) containing the display format of the variable, and structured as follows :
  - byte 1 contains an item of data specifying whether the variable is signed (+ sign), or not (**space**)
  - bytes 2 and 3 take the following format

DISPLAY FORMAT	BYTE 2	BYTE 3
ASCII or NUMERICAL no decimals	ASCII code of the space, i.e. 20	ASCII code of the space, i.e. 20
NUMERICAL with less than 10 decimal places	ASCII code of the space, i.e. 20	ASCII code of the number of decimal places
NUMERICAL with at least 10 decimal places	ASCII code for the ten of the number of decimal places	ASCII code for the units of the number of decimal places

- byte 4 contains an item of data indicating the type of display :
  - N** for no format
  - D** for numerical
  - A** for Ascii
- one **word** (in upper case) specifying whether the variable should be refreshed or not :
  - '**Y** ' (concluded with a space) for yes
  - '**N** ' (concluded with a space) for no
- one **word** (in upper case) containing the value '**N**' (concluded with a space)
- **seven words** containing the value **0**

**Example of use :**

SEND\_MSG (ADR#0.0.4,%MW0:12, %MW100:4);

Note: In this table « spc » represents a space.

%MW0	<b>16#CC17</b>	Marker
%MW1	<b>0</b>	Fixed value
%MW2	<b>18</b>	Size in bytes of the next zone which contains 9 words
%MW3	<b>'Ov'</b>	Message text
%MW4	<b>'en'</b>	Message text (cont)
%MW5	<b>'spc4'</b>	Message text (end)
%MW6	<b>0</b>	Marks end of message
%MW7	<b>2</b>	Message position (line number)
%MW8	<b>15</b>	Message position (column number)
%MW9	<b>'ND'</b>	Message characteristics (No attribute and Double font)
%MW10	<b>'Nspc'</b>	Message characteristics (no print)
%MW11	<b>0</b>	The message does not contain a variable

---

## 5.2-2 Status message entry controlled by the PLC : ASK\_MSG and GET\_MSG functions

It is possible to build status messages controlled from the PLC application and to send them via the (%MWi) internal **words**. This is the role of the ASK\_MSG and GET\_MSG functions.

For information purposes, the “**Data to Send**” parameter requires a maximum of **47** words. It is composed as follows :

- the **first word** contains a value marker **16#CC17**
- the **second word** contains the type of command (command number = 33 for ASK\_MSG and 6 for GET\_MSG)
- the **third word** contains the length in bytes of the next **words** zone
- the **next words** must contain the message text to send, including the " underscores " representing the characters expected when the variable is displayed. This text has a maximum length of 40 characters.

If the text is made up of an odd number of characters, the last byte has a value of 0. If the text has an even length and fewer than 39 characters, the last word must contain the value 0.

- one **word** containing the line number where the message must be displayed
- one **word** containing the column number where the beginning of the message must be displayed
- a zone of **two words** (four characters) containing the characteristics of the message, and structured as follows :
  - the first character (in upper case) corresponds to the video attribute (**B** = flashing, **R** = reverse video, **A** = flashing and reverse video, **N** = no attribute)
  - the second character (in upper case) corresponds to the character font size (**S** = single, **D** = double height and double length)
  - the second **word** (in upper case) relates to the print option (**Y followed by a space** = yes, **N followed by a space** = no)
- one **word** containing the position of the variable to display, counted in number of characters in relation to the beginning of the message
- one **word** containing the number of characters to display for the variable
- one **word** containing an additional command :
  - 0 : no command (ASK\_MSG synchronized entry)
  - 24 : free entry permitted after display (GET\_MSG multiple entry)
- one **word** containing the value 16#0030
- **two words** (in upper case) containing the entry field type :
  - '**BIT** ' (concluded with a space) for a bit type
  - '**ANA** ' (concluded with a space) for a **word** type
  - '**LNG** ' (concluded with a space) for a double **word** type
- one **word** containing the value 0



- one **word** (in upper case) containing the variable type to display,
  - 'B ' (concluded with a space) for a bit type
  - 'W ' (concluded with a space) for a word type
  - 'DW ' for a double word type
- one **word** containing the index of the address of the variable to display
- **two words** (in upper case) containing the display format of the variable, and structured as follows :
  - byte 1 contains an item of data specifying whether the variable is signed (+ sign), or not (**space**)
  - bytes 2 and 3 take the following format

DISPLAY FORMAT	BYTE 2	BYTE 3
ASCII or NUMERICAL no decimals	ASCII code of the space, i.e. 20	ASCII code of the space, i.e. 20
NUMERICAL with less than 10 decimal places	ASCII code of the space, i.e. 20	ASCII code of the number of decimal places
NUMERICAL with at least 10 decimal places	ASCII code for the tensof the number of decimal places	ASCII code for the units of the number of decimal places

- byte 4 contains an item of data indicating the type of display :
  - N** for no format
  - D** for numerical
  - A** for Ascii
- one **word** (in upper case) specifying whether or not the variable should be refreshed :
  - 'Y ' (concluded with a space) for yes
  - 'N ' (concluded with a space) for no (obligatory value for ASK\_MSG)
- one **word** (in upper case) relating to the field attribute :
  - 'I ' (concluded with a space) for Increment
  - 'L ' (concluded with a space) for other cases
- one **word** indicating whether the variable is limited or not :
  - 0** for an unlimited variable
  - 1** use of minimum limit only
  - 2** use of maximum limit only
  - 3** use of minimum and maximum limits
- **two words** containing the value of the minimum limit
- **two words** containing the value of the maximum limit
- **two words** containing the value of the increment

#### Example of use :

Programming the GET\_MSG function from the %MW0:38 internal words

---

In this table « spc » represents a space.

%MW0	<b>16#CC17</b>	Marker
%MW1	<b>6</b>	Command number for GET_MSG
%MW2	<b>70</b>	Size in bytes of the next zone which contains 35 words
%MW3	<b>'Te'</b>	Message text
%MW4	<b>'mp'</b>	Message text (cont)
%MW5	<b>'er'</b>	Message text (cont)
%MW6	<b>'at'</b>	Message text (cont)
%MW7	<b>'ur'</b>	Message text (cont)
%MW8	<b>'e spc'</b>	Message text (cont)
%MW9	<b>'=spc'</b>	Message text (cont)
%MW10	<b>'_'</b>	Message text (cont)
%MW11	<b>'_ spc'</b>	Message text (cont)
%MW12	<b>'°C'</b>	Message text (end)
%MW13	<b>0</b>	Marks end of message
%MW14	<b>3</b>	Text position (line number)
%MW15	<b>11</b>	Text position (column number)
%MW16	<b>'NS'</b>	Message characteristics (no attribute, single size)
%MW17	<b>'Nspc'</b>	Message characteristics (no print)
%MW18	<b>15</b>	Variable position from beginning of message
%MW19	<b>3</b>	Number of characters to display
%MW20	<b>24</b>	Additional command (entry after display)
%MW21	<b>16#0030</b>	Reserved value
%MW22	<b>'AN'</b>	Entry field type (AN = beginning of ANA)
%MW23	<b>'A spc'</b>	Entry field type (cont)
%MW24	<b>0</b>	Reserved value
%MW25	<b>'W spc'</b>	Type of variable to display (W = word type variable)
%MW26	<b>10</b>	Index of the address of the variable to display (%MW10)
%MW27	<b>'spcspc'</b>	Display format continued (spc for an unsigned variable, spc for beginning of coding of the number of decimal places after the decimal point)
%MW28	<b>'spcD'</b>	Display format continued (spc for end of coding of the number of decimal places after the decimal point, D for decimal format)
%MW29	<b>'Y spc'</b>	The variable must be refreshed
%MW30	<b>'I spc'</b>	The entry is incremental type
%MW31	<b>0</b>	The variable is not limited
%MD32	<b>0</b>	Value of the minimum limit
%MD34	<b>0</b>	Value of the maximum limit
%MD36	<b>50</b>	Value of the increment

### 5.2-3 Displaying a PLC alarm message : SEND\_ALARM function

It is possible to activate alarm messages from the PLC application and to send them via the (%MWi) internal words to display them on the CCX17 operator panel screen. This is the role of the SEND\_ALARM function.

For information purposes, the “**Data to Send**” parameter requires a maximum of **37** words. It is composed as follows :

- the **first word** contains a value marker **16#CC17**
- the **second word** contains the value 0
- the **third word** contains the length in bytes of the next words zone
- the **fourth word** contains a fictitious number assigned to the alarm message (this word will subsequently serve to deactivate the alarm if required). The value of this word must be between 900 and 999.
- the **next words** must contain the message text to send, including the « underscores » representing the characters expected when the variable is displayed. This text has a maximum length of 40 characters.  
If the text is made up of an odd number of characters, the last byte has the value 0. If the text has an even length and has fewer than 39 characters, the last word must contain the value 0.
- a zone of **two words** containing the message characteristics, and structured as follows :
  - the first character (in upper case) corresponds to the character font size (**S** = single size, **D** double height and double length)
  - the second character (in upper case) corresponds to the print option (**Y** = yes, **N** = no)
  - the second word (in upper case) corresponds to the overprint option (**Y followed by a space** = yes, **N followed by a space** = no)

If the user does not wish to display the variable, the next word must be at 0, otherwise the following parameters must be added:

- one **word** containing the position of the variable to display, counted in number of characters in relation to the beginning of the message
- one **word** containing the number of characters to display for the variable
- one **word** containing the value **16#0030**
- two **words (in upper case)** containing the entry field type :
  - ‘**BIT** ‘ (concluded with a space) for a bit type
  - ‘**ANA** ‘ (concluded with a space) for a word type
  - ‘**LNG** ‘ (concluded with a space) for a double word type
- one **word** containing the value 0
- one **word** (in upper case) containing the type of variable to display
  - ‘**B** ‘ (concluded with a space) for a bit type
  - ‘**W** ‘ (concluded with a space) for a word type
  - ‘**DW** ‘ for a double word type

- one **word** containing the index of the address of the variable to display
- **two words** (in upper case) containing the display format of the variable, and structured as follows :
  - byte 1 contains an item of data specifying whether the variable is signed (+ sign), or not (space)
  - bytes 2 and 3 take the following format

DISPLAY FORMAT	BYTE 2	BYTE 3
ASCII or NUMERICAL no decimals	ASCII code of the space, i.e. 20	ASCII code of the space, i.e. 20
NUMERICAL with less than 10 decimal places	ASCII code of the space, i.e. 20	ASCII code of the number of decimal places
NUMERICAL with at least 10 decimal places	ASCII code for the tensof the number of decimal places	ASCII code for the units of the number of decimal places

- byte 4 contains an item of data indicating the type of display :

**N** for no format

**D** for numerical

**A** for Ascii

### Example of use :

SEND\_ALARM (ADR#0.0.4,%MW0:29, %MW100:4);

Note: In this table « spc » represents a space.

%MW0	<b>16#CC17</b>	Marker
%MW1	<b>0</b>	Fixed value
%MW2	<b>52</b>	Size in bytes of the next zone which contains 26 words
%MW3	<b>900</b>	Alarm message number
%MW4	<b>'Ov'</b>	Message text
%MW5	<b>'er'</b>	Message text (cont)
%MW6	<b>'he'</b>	Message text (cont)
%MW7	<b>'at'</b>	Message text (cont)
%MW8	<b>'ed'</b>	Message text (cont)
%MW9	<b>'spco'</b>	Message text (cont)
%MW10	<b>'ve'</b>	Message text (cont)
%MW11	<b>'nspc'</b>	Message text (cont)
%MW12	<b>'=spc'</b>	Message text (cont)
%MW13	<b>'_'</b>	Message text (cont)
%MW14	<b>'_spc'</b>	Message text (cont)
%MW15	<b>'°C'</b>	Message text (end)
%MW16	<b>0</b>	Marks end of message
%MW17	<b>'SY'</b>	Message characteristics (single size, print)
%MW18	<b>'Yspc'</b>	Message characteristics (overprint)

%MW19	<b>19</b>	Position of the variable from beginning of message
%MW20	<b>3</b>	Number of characters to display
%MW21	<b>16#0030</b>	Reserved value
%MW22	<b>'AN'</b>	Entry field type (AN = beginning of ANA)
%MW23	<b>'A<math>spc</math>'</b>	Entry field type (cont)
%MW24	<b>0</b>	Reserved value
%MW25	<b>'W<math>spc</math>'</b>	Type of variable to display (W = word type variable)
%MW26	<b>10</b>	Index of address of variable to display (%MW10)
%MW27	<b>'<math>spcspc</math>'</b>	Display format ( $spc$ for an unsigned variable, $spc$ for beginning of coding of the number of decimal places after the decimal point)
%MW28	<b>'<math>spcD</math>'</b>	Display format continued ( $spc$ for end of coding of the number of decimal places after the decimal point, D for decimal format)

---

#### 5.2-4 Displaying status message or alarm message or group of messages contained in the CCX17 memory : ASK\_VALUE, DISPLAY\_MSG, GET\_VALUE, DISPLAY\_ALRM, DISPLAY\_GRP functions.

It is possible for these functions to send via the (%MWi) internal words.

For information purposes, the “**Data to Send**” parameter requires 1 word containing either the status message number, the alarm message number or the group of messages number.

**Example of use :**

#### Programming the DISPLAY\_GRP function from the %MW0 internal word

DISPLAY\_GRP (ADR#0.0.4,%MW0, %MW10:4) with %MW0 := 3

---

### 5.2-5 Displaying the LEDs on the indicator bank : CONTROL\_LEDS function

It is possible to define the status of the relay (Version 2.1 and later) and LEDs on the indicator bank and to send them via (%MWi) internal words without passing through assisted entry (i.e. without using the %KWi internal constants). This is the role of the CONTROL\_LEDS function.

For information purposes, the “**Data to Send**” parameter requires 2 words (%MWi:2). It is composed as follows:

- the **first word** contains a **16#CC17** value marker
- the **second word** indicates the coding of each LED as well as the status of relay to be sent to the terminal.
  - bits 0 to 3 : status of green LED
  - bits 4 to 7 : status of yellow LED
  - bits 8 to 11 : status of red LED
  - bits 12 to 15 : status of relay

The status of each LED is coded on these 4 bits in the following way:

- 0000 : LED status unchanged
- 0001 : LED off
- 0010 : LED on
- 1111 : LED flashing

The status of relay is coded on bits 12 to 15 in the following way:

- 0000 : status of relay unchanged
- 0001 : status of relay open
- 0010 : status of relay closed

#### Example of use :

```
CONTROL_LEDS (ADR#0.0.4,%MW0:2, %MW10:4);
```

- %MW0 := 16#CC17
- %MW1 := 16#1112 (green LED on (2), yellow LED off (1), red LED off(1), status of relay open (1))

### 5.2-6 Configuring the command keys : ASSIGN\_KEYS function

It is possible to define the configuration of command keys (bit associated or not, operating mode of the key, assignment by the CCX17, etc) and to send them via the (%MWi) internal words. This is the role of the ASSIGN\_KEYS function.

For information purposes, the "Data to Send" parameter requires 16 words. It is composed as follows :

- the **first word** contains a value marker **16#CC17**

If the assignment is done by the PLC :

- the **second word** contains the list of keys to be configured. Each command key is coded on a bit (bit n°i at 0 command key not configured, bit n°i at 1 command key configured):
  - bit 0 : command key 1
  - bit 1 : command key 2
  - .....
  - bit 11 : command key 12
- the **next two words** indicate the operating mode of each command key. Each key is coded on two bits :
  - 00 : RESET
  - 01 : edge operating mode
  - 10 : toggle operating mode
  - 11 : no action
 Bit 0/1 = command key 1  
 Bit 2/3 = command key 2  
 .....  
 Bit 22/23 = command key 12
- the **next twelve words** contain either the value -1 when the command keys are not assigned, or the index of internal bits assigned to the command keys :
  - word 1 = index of bit assigned to key 1
  - .....
  - word 12 = index of bit assigned to key 12

**If the assignment is made by the CCX17 :**

- the **second word** contains the value **16#F000**
- the values of the **next fourteen words** are immaterial and will be ignored by the CCX17 operator panel

---

**Example of use :**

- ASSIGN\_KEYS (ADR#0.0.4,%MW0:16, %MW100:4);
- %MW0 := 16#CC17
- %MW1 := 16#000F (command keys 1 to 4 configured, others not)
- %MW2 := 16#FF09 and %MW3 := 16#00FF (key 1 in edge mode (01), key 2 in toggle mode (10), keys 3 and 4 inhibited (00), the others not programmed (11))
- %MW4 := 10 (key 1 assigned to bit %M10)
- %MW5 := 11 (key 2 assigned to bit %M11)
- %MW6 to %MW15 := -1 (key 3 to 12 not assigned)

---

**5.2-7 Sending generic commands : PANEL\_CMD function**

It is possible to generate different types of command (delete, print) and to send them via the (%MWi) internal words. This is the role of the PANEL\_CMD function. For information purposes, the “**Data to Send**” parameter requires a maximum of **3** words. It is composed as follows :

- the **first word** contains a **16#CC17** value marker
- the **second word** contains the command number :
  - 1 : clear screen
  - 2 : delete line
  - 9 : print message log
  - 10 : clear message log
  - 11 : print alarm log
  - 13 : clear alarm log
  - 29 : clear alarm (from **1 to 300**) intended for **CCX17**
  - 30 : clear alarm (from 900 to 999) intended for PLC
- the **third word** indicates the parameter settings of the command. To delete a line, it is equal to the line number and to cancel an alarm, it corresponds to the alarm number. For the other commands, it does not apply.

**Example of use :**

- PANEL\_CMD (ADR#0.0.4,%MW0:2, %MW10:4) with
  - %MW0 := 16#CC17 and %MW1:=1 (clear screen)



**A**

Activity bit 3/5  
 ADJUST 2/35  
 ASK\_MSG 1/2, 2/16, 3/5, 5/6  
 ASK\_VALUE 1/3, 2/27, 3/5, 5/11  
 ASSIGN\_KEYS 1/3, 2/31, 5/13  
 Assigning PLC variables 4/2  
 ASYNCHRONOUS 1/1  
 AUTOMATIC mode 2/11

**C**

Configuring the alarm message 4/3  
 Configuring the command keys 4/3  
 Configuring the status messages 4/3  
 CONTROL\_LEDS 1/3, 2/30, 5/12

**D**

Data to Receive 3/5, 5/1  
 Data to Send 3/4, 5/1, 5/3, 5/9, 5/11, 5/12  
 5/13, 5/14  
 Details of the application 4/2  
 DISPLAY\_ALARM 1/3, 2/25, 5/11  
 DISPLAY\_GRP 1/3, 2/24, 5/11  
 DISPLAY\_MSG 1/3, 2/23, 5/11  
 Documentation 2/42

**E**

Exchange number 3/5  
 Exchange report 3/6, 5/2

**F**

FIPIO addressing 3/2

**G**

GET\_MSG 2/13, 5/6  
 GET\_VALUE 1/3, 2/29, 5/11

**I**

IN 3/1  
 IN/OUT 3/1  
 Integrated MMI 1/1  
 intra-station address 3/2

**L**

Length 3/8

**M**

Management of the alarm message 4/3  
 multiple mode 2/13

**O**

Operation report 3/7, 5/2  
 Operator panel address 5/1  
 OUT 3/1

**P**

PANEL\_CMD 1/3, 2/33, 5/14  
 Precautions 4/1  
 Processing status messages 4/3

**R**

Report 3/5, 5/1

**S**

SEND\_ALARM 1/2, 2/20, 5/9  
 SEND\_MSG 1/2, 2/10, 5/3

**T**

Time-out 3/8

**U**

UNITELWAY addressing 3/2

---

D

**VOLUME 2**

**Counting**

**Application-  
Specific  
Functions**

---

**Axis Control**

---

**Stepper Motor Axis Control**

---

**VOLUME 1**

**Application-Specific Common Functions**

**Application-  
Specific  
Functions**

**Discrete**

**AS-i Bus Setup**

**Man-Machine Interface**

---

**VOLUME 3**

**Analog**

**Application-  
Specific  
Functions**

**PID Control**

**Process Control**

**Weighing**

---

**A**

**B**

**C**



<b>Section</b>	<b>Page</b>
<b>1 General</b>	<b>1/1</b>
1.1 Presentation	1/1
1.2 Downcounting function (TSX CTY 2A / 4A)	1/2
1.3 Upcounting function (TSX CTY 2A / 4A)	1/3
1.4 Up/down counting function (TSX CTY 2A / 4A)	1/4
1.5 Up/down counting and measurement with a TSX CTY 2C module	1/5
<b>2 Description of the upcounting and downcounting functions (TSX TY 2A/4A)2/1</b>	
2.1 Overview	2/1
2.2 Functions and timing diagrams	2/4
2.2-1 Invalid value	2/4
2.2-2 Enable	2/5
2.2-3 Preset or reset	2/7
2.2-4 Comparison	2/10
2.2-5 Counter outputs	2/11
2.2-6 Physical outputs Q0 and Q1	2/14
2.2-7 Event processing	2/18
2.3 Description of language objects associated with the upcounting and downcounting functions	2/20

<b>Section</b>	<b>Page</b>
<b>3 Description of up/down counting function (TSX CTY 2A / 4A)</b>	<b>3/1</b>
3.1 Overview	3/1
3.2 Functions and timing diagrams	3/3
3.2-1 Input interface	3/3
3.2-2 Invalid value	3/6
3.2-3 Enable	3/7
3.2-4 Preset	3/8
3.2-5 Capture (read input)	3/11
3.2-6 Comparison	3/12
3.2-7 Counter outputs	3/13
3.2-8 Physical outputs Q0 and Q1	3/16
3.2-9 Event processing	3/20
3.3 Description of the language objects associated with the function	3/21
<b>4 Description of the up/down counting and measurement function (CTY 2C) 4/1</b>	<b>4/1</b>
4.1 Overview	4/1
4.2 Functions and timing diagrams	4/6
4.2-1 Input interface	4/6
4.2-2 Invalid value	4/12
4.2-3 Enable	4/13
4.2-4 Speed monitoring	4/14
4.2-5 Preset	4/16
4.2-6 Capture (read input)	4/19
4.2-7 Comparison	4/21
4.2-8 Counter outputs	4/22
4.2-9 Outputs	4/24
4.2-10 Event processing	4/31
4.3 Description of the language objects associated with the function	4/32

<b>Section</b>	<b>Page</b>
<b>5 Setup (TSX CTY 2A / 4A / 2C)</b>	<b>5/1</b>
5.1 Introduction	5/1
5.1-1 Presymbolization	5/2
5.2 The configuration editor	5/3
5.2-1 Accessing the configuration editor	5/3
5.3 Configuring counter modules	5/4
5.3-1 Selecting the modules	5/4
5.3-2 Accessing the parameter settings for a counter module	5/5
5.4 Channel configuration mode (TSX CTY 2A / 4A / 2C)	5/7
5.4-1 Downcounting, upcounting and up/down counting functions	5/7
5.4-2 Up/down counting and measuring with a TSX CTY 2C module	5/10
5.5 Channel adjust mode (TSX CTY 2A / 4A / 2C)	5/16
5.6 Debug mode (TSX CTY 2A / 4A / 2C)	5/19
5.6-1 Debug screen in extended mode (TSX CTY 2A / 4A / 2C)	5/22
5.6-2 Diagnostic screens (TSX CTY 2A / 4A / 2C)	5/29
5.6-3 Faults and diagnostics (TSX CTY 2A / 4A / 2C)	5/30
<b>6 Events</b>	<b>6/1</b>
6.1 Presentation	6/1
6.2 Methodology for programming event processing	6/3
6.3 Event program execution	6/4

<b>Section</b>	<b>Page</b>
<b>7 Operating modes (TSX CTY 2A / 4A / 2C)</b>	<b>7/1</b>
7.1 Processing on power breaks and power returns	7/1
7.2 Processing on a warm restart	7/1
7.3 Processing on a cold start	7/2
7.4 Processing in STOP mode	7/2
7.5 Reconfiguration in online mode	7/2
<b>8 Summary of language objects</b>	<b>8/1</b>
8.1 Language objects associated with counting	8/1
8.1-1 Implicit exchange objects	8/4
8.1-2 Explicit exchange objects	8/8
8.2 Addressing objects	8/14
8.3 Explicit exchanges	8/16
8.3-1 Reading the status word	8/17
8.3-2 Reading adjustment parameters	8/17
8.3-3 Writing adjustment parameters	8/18
8.3-4 Saving adjustment parameters	8/18
8.3-5 Restoring adjustment parameters	8/18
8.3-6 Execution conditions	8/19



<b>Section</b>	<b>Page</b>
<b>9 Example</b>	<b>9/1</b>
9.1 External specifications of the example	9/1
9.2 Internal specifications of the application	9/2
9.2-1 Configuration	9/2
9.2-2 Processor	9/2
9.2-3 Counting	9/3
9.2-4 Discrete I/O	9/4
9.2-5 Internal bits and words	9/4
9.3 Program	9/5
9.3-1 Preprocessing	9/5
9.3-2 Sequential processing	9/6
9.3-3 Grafcet actions and transitions	9/7
9.4 Post-processing	9/10
9.5 Event processing	9/10
<b>10 Performance</b>	<b>10/1</b>
10.1 Time performance levels	10/1
10.2 Questions/Answers	10/3

<b>Section</b>	<b>Page</b>
<b>11 TSX CTY 2C special functions</b>	<b>11/1</b>
11.1 Presentation	11/1
11.2 Description	11/1
11.2-1 No special function	11/1
11.2-2 Special function number 1 : Time elapsed since last pulse	11/2
11.2-3 Special function number 2 : Internal capture and preset	11/3
11.2-4 Special function number 3 : Correct speed and moving part stationary	11/4
11.3 Compatibility of the special functions	11/7
11.4 Exclusivity of the special functions	11/7
<b>12 Index</b>	<b>12/1</b>

---

---

## 1.1 Presentation

---

TSX CTY 2A, TSX CTY 4A and TSX CTY 2C modules are 2-channel (CTY 2A / 2C) or 4-channel (CTY 4A) modules for TSX Premium, PMX Premium or PCX Premium modular PLCs. They are used to count pulses from a sensor at a maximum frequency of 40 kHz (CTY 2A / 4A) or 1 MHz (CTY 2C).

TSX CTY 2C modules also offer a **speed** function in points/second :

- over a measurement and sampling period which can be adjusted by the user, the speed is calculated and updated
- speed monitoring, using an **overspeed threshold** which can be adjusted by the user, provides a safety measure for the outputs if the overspeed threshold is exceeded (set to 0).

Counter modules can be installed in all the available slots in a PLC configuration, up to a maximum of :

- 8 application-specific channels in a TSX / PMX / PCX 57-10 configuration
- 24 application-specific channels in a TSX / PMX 57-20 or TSX 57-25 configuration
- 32 application-specific channels in a TSX 57-30 or TSX / PMX / PCX 57-35 configuration
- 48 application-specific channels in a TSX 57-40 or TSX / PMX 57-45 configuration

Counter modules can be used to perform the following functions for each channel :

- upcounting, downcounting and up/down counting with a TSX CTY 2A or TSX CTY 4A module
- Up/down counting and measuring with a TSX CTY 2C module

The sensor used on each channel may be :

- a 2 or 3-wire proximity sensor, PNP or NPN. If mechanical contact outputs are used, the immunity of the channel receiving the counting pulses should be increased to reduce bounce when closing the contact.
- an incremental encoder with 5 VDC differential output signals (encoder with RS 422/485 line driver)
- an incremental encoder with 10-30 VDC output signals (Totem Pole encoder),
- an SSI serial output absolute encoder, 485 standard interface (TSX CTY 2C only)
- a parallel output absolute encoder, using a TELEFAST adaptor ABE-7CPA11 (TSX CTY 2C only).

## 1.2 Downcounting function (TSX CTY 2A / 4A)

Subfunction	Counting pulses : 40 KHz Channels 0 and 1/ Channels 0,1, 2 and 3
Input interface	1 physical input <b>IA</b> : <b>5V, 24V, RS422/485</b>
Enable	<ul style="list-style-type: none"> <li>• hardware enable : <b>24V</b>,</li> <li>• software enable.</li> </ul>
Preset	<ul style="list-style-type: none"> <li>• physical preset : <b>24V</b>, 2 configuration modes,</li> <li>• software preset.</li> </ul>
Preset value	adjustable.
Comparison of the current value with :	0
Counter output	<p><b>counter output 0 :</b></p> <ul style="list-style-type: none"> <li>• set if value 0 is reached,</li> <li>• reset if preset done or direct preset.</li> </ul>
Events	<ul style="list-style-type: none"> <li>• preset,</li> <li>• enable,</li> <li>• crossing of value 0,</li> <li>• state of counter output 0,</li> <li>• overrun.</li> </ul>

### 1.3 Upcounting function (TSX CTY 2A / 4A)

Subfunction	Counting pulses : 40 KHz Channels 0 and 1/ Channels 0,1, 2 and 3
Input interface	1 physical input <b>IA</b> : 5V, 24V, RS422/485
Enable	<ul style="list-style-type: none"> <li>• hardware enable : <b>24V</b>,</li> <li>• software enable.</li> </ul>
Reset	<ul style="list-style-type: none"> <li>• physical reset : <b>24V</b>, 2 configuration modes</li> <li>• software reset.</li> </ul>
Reset value	reset value predefined at 0.
Comparison of the current value with :	high setpoint, threshold 0, threshold 1.
Counter outputs	<p><b>counter output 0 :</b></p> <ul style="list-style-type: none"> <li>• set if the setpoint value is reached,</li> <li>• reset if reset done or direct reset.</li> </ul> <p><b>counter output 1 :</b></p> <ul style="list-style-type: none"> <li>• 5 adjustable set conditions,</li> <li>• 5 adjustable reset conditions,</li> </ul>
Events	<ul style="list-style-type: none"> <li>• reset,</li> <li>• enable,</li> <li>• crossing of threshold 0,</li> <li>• crossing of threshold 1,</li> <li>• crossing of high setpoint,</li> <li>• direction of crossing,</li> <li>• state of counter output 0,</li> <li>• state of counter output 1,</li> <li>• overrun.</li> </ul>

## 1.4 Up/down counting function (TSX CTY 2A / 4A)

Subfunction	Counting pulses : 40 KHz Channels 0 and 1/ Channels 0,1, 2 and 3
Input interface	Physical inputs : <b>5V, 24V, RS422/485</b> : <ul style="list-style-type: none"> <li>• physical input <b>IA</b> +/- and 1 software input, counting direction,</li> <li>• physical input <b>IA</b> +/- and 1 physical input <b>IB</b>, counting direction,</li> <li>• physical input <b>IA</b> + and 1 physical input <b>IB</b> -,</li> <li>• physical inputs <b>IA</b> and <b>IB</b> incremental encoder with 1 physical zero marker input <b>IZ</b>.</li> </ul>
Enable	<ul style="list-style-type: none"> <li>• hardware enable : <b>24V</b>,</li> <li>• software enable.</li> </ul>
Preset	<ul style="list-style-type: none"> <li>• physical preset : <b>24V</b>, 7 configuration modes,</li> <li>• software preset.</li> </ul>
Preset value	adjustable.
Capture	<ul style="list-style-type: none"> <li>• hardware capture : <b>24V</b>, 2 configuration modes,</li> <li>• software capture.</li> </ul>
Comparison of the current value with	<ul style="list-style-type: none"> <li>• high setpoint,</li> <li>• low setpoint,</li> <li>• threshold 0,</li> <li>• threshold 1.</li> </ul>
Comparison of the captured value with	<ul style="list-style-type: none"> <li>• high setpoint,</li> <li>• low setpoint,</li> <li>• threshold 0,</li> <li>• threshold 1.</li> </ul>
Counter outputs	<b>counter outputs 0 and 1 :</b> <ul style="list-style-type: none"> <li>• 17 adjustable set conditions,</li> <li>• 17 adjustable reset conditions,</li> </ul>
Events	<ul style="list-style-type: none"> <li>• preset,</li> <li>• enable,</li> <li>• capture,</li> <li>• crossing of threshold 0,</li> <li>• crossing of threshold 1,</li> <li>• crossing of high setpoint,</li> <li>• crossing of low setpoint,</li> <li>• state of counter output 0,</li> <li>• state of counter output 1,</li> <li>• direction of crossing,</li> <li>• overrun.</li> </ul>

## 1.5 Up/down counting and measurement with a TSX CTY 2C module

Subfunction	Counting pulses : 1 MHz incremental encoder : 500 kHz x 1 or 250 kHz x 4 absolute encoder : transmission clock 1 MHz Channels 0 and 1
Input interface	physical inputs : <b>5V, 24V, RS422/485</b> : <ul style="list-style-type: none"> <li>• 1 physical input <b>IA +/-</b> and 1 software input, counting direction,</li> <li>• 1 physical input <b>IA +/-</b> and 1 physical input <b>IB</b>, counting direction,</li> <li>• 1 physical input <b>IA +</b> and 1 physical input <b>IB -</b>,</li> <li>• physical inputs <b>IA</b> and <b>IB</b> incremental encoder with a physical zero marker input <b>IZ</b>.</li> </ul> <b>5V</b> I/O (serial output absolute encoder or parallel output absolute encoder, via TELEFAST ABE-7CPA11): <ul style="list-style-type: none"> <li>• 1 physical input <b>SSI Data</b>,</li> <li>• 1 transmission clock output <b>SSICLK</b>.</li> </ul>
Enable / Output Q2	<ul style="list-style-type: none"> <li>• hardware enable or physical output Q2 : <b>24V</b> (configurable),</li> <li>• software enable.</li> </ul>
Preset	<ul style="list-style-type: none"> <li>• physical preset : <b>24V</b>, 7 configuration modes,</li> <li>• software preset (counting pulses or incremental encoder).</li> </ul>
<ul style="list-style-type: none"> <li>• Preset value</li> <li>• Offset value</li> </ul>	<ul style="list-style-type: none"> <li>• adjustable (counting pulses or incremental encoder),</li> <li>• adjustable (absolute encoder).</li> </ul>
Speed measurement period	adjustable.
Capture	<ul style="list-style-type: none"> <li>• hardware capture : <b>24V</b>, 3 configuration modes,</li> <li>• capture before physical preset,</li> <li>• software capture.</li> </ul>
Output Q3	<ul style="list-style-type: none"> <li>• programmable frequency physical output (adjustable),</li> <li>• adjustable programmable frequency value.</li> </ul>
Modulo value	configurable.
Value of thresholds 0 and 1	adjustable.
Overspeed threshold value	adjustable.
Comparison of the current value with :	<ul style="list-style-type: none"> <li>• threshold 0,</li> <li>• threshold 1.</li> </ul>

Subfunction (cont.)	Counting pulses : 1 MHz incremental encoder : 500 kHz x 1 or 250 kHz x 4 absolute encoder : transmission clock 1 MHz Channels 0 and 1
Comparison of the captured value with :	<ul style="list-style-type: none"> <li>• threshold 0,</li> <li>• threshold 1.</li> </ul>
Counter outputs	counter outputs 0 and 1 : for outputs Q0 and Q1 <ul style="list-style-type: none"> <li>• 13 adjustable SET and RESET conditions.</li> </ul>
Outputs Q0 to Q3	<ul style="list-style-type: none"> <li>• protection against short-circuits with manual/automatic reactivation,</li> <li>• configurable maintain or fallback,</li> <li>• manual or automatic mode.</li> </ul>
Events	<ul style="list-style-type: none"> <li>• preset,</li> <li>• enable,</li> <li>• capture with direction of edge during capture,</li> <li>• crossing of threshold 0 with direction of crossing,</li> <li>• crossing of threshold 1 with direction of crossing,</li> <li>• crossing of modulo in + direction,</li> <li>• crossing of modulo in - direction,</li> <li>• state of counter output 0,</li> <li>• state of counter output 1,</li> <li>• overrun.</li> </ul>



## 2 Description of the upcounting and downcounting functions (TSX CTY 2A / 4A)

### 2.1 Overview

---

These functions of TSX CTY 2A and TSX CTY 4A (\*) modules are used respectively to downcount or upcount pulses from a sensor (24 bits + sign), between the values :

- -16 777 216 and +16 777 215 (downcounting function),
- 0 and +16 777 215 (upcounting function).
- The upcounter or downcounter can be enabled by a physical input or by program. It can activate event processing.
- The current value is only accessible in read mode.
- The counter can be preset by a physical input or by program. It is configurable (2 modes) and can activate event processing.
- Event processing can also be activated when the current value exceeds the following values :
  - value 0, for downcounting,
  - high setpoint, threshold 0 or threshold 1, for upcounting.
- These functions also offer one reflex counter output (downcounting) or two reflex counter outputs (upcounting) which can be applied directly to one physical output associated with the channel and physically situated on the TSX CTY 2A / 4A module.

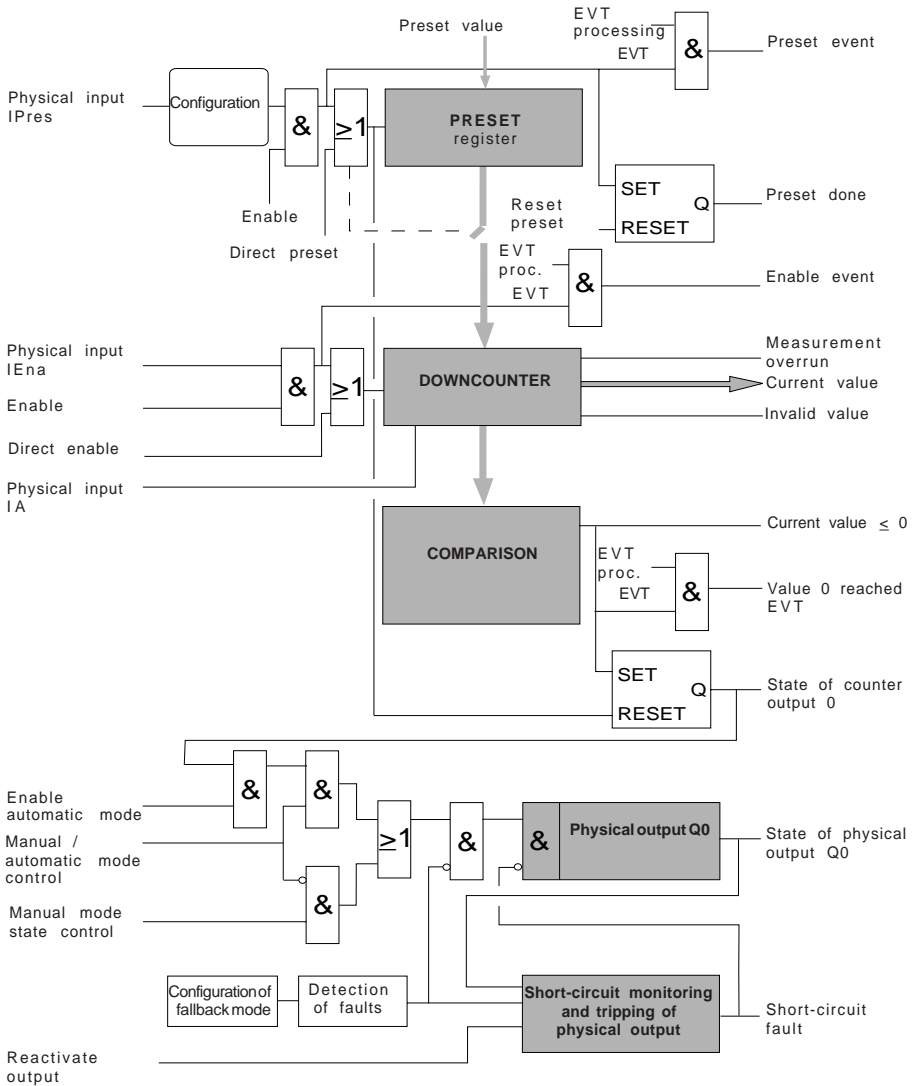
For counter output 0, the SET and RESET conditions are predefined.

- current value  $\leq 0$  (SET condition for downcounting),
- preset done or direct preset (RESET condition for downcounting),
- current value  $\geq$  high setpoint (SET condition for upcounting),
- reset done or direct reset (RESET condition for upcounting).

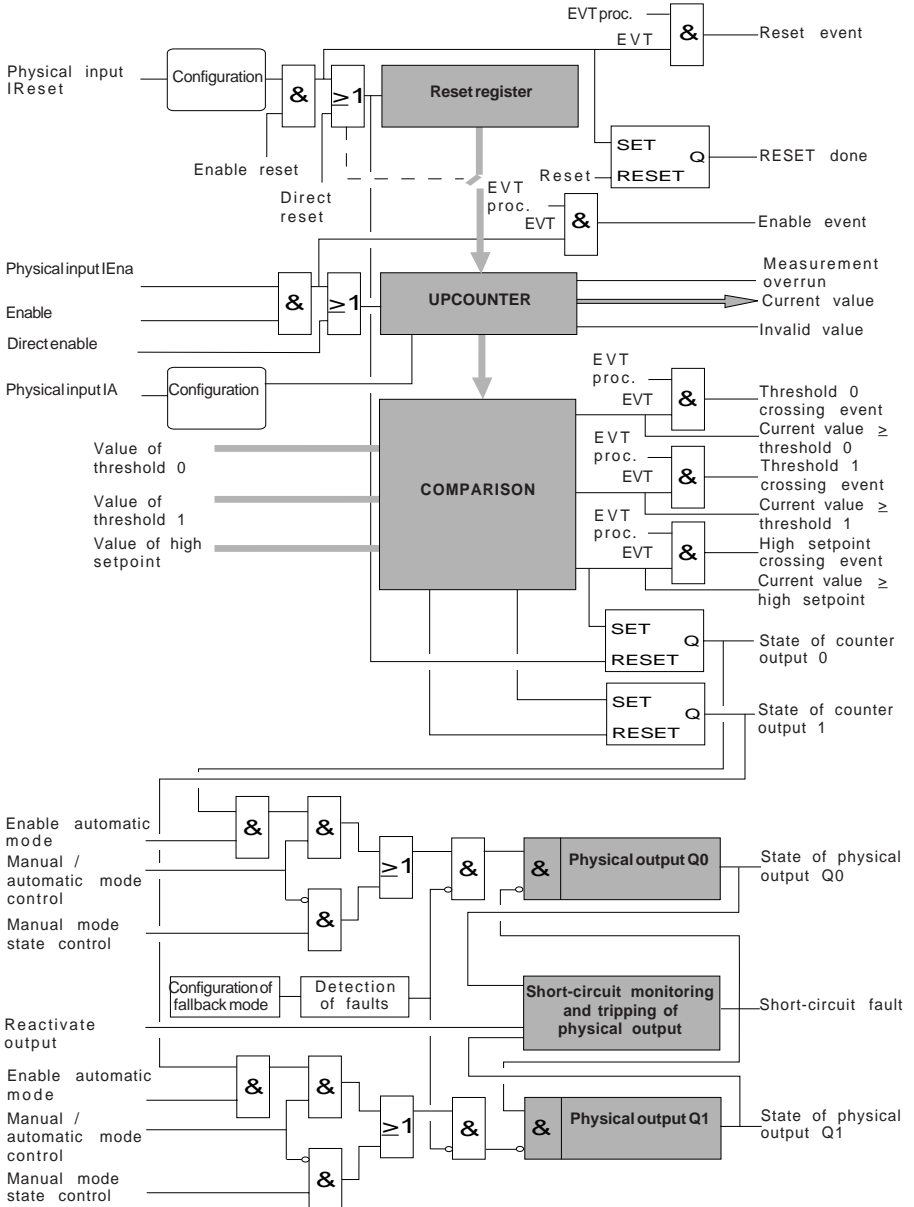
For counter output 1 (upcounting only) the parameters of the SET and RESET conditions can be defined (5 conditions).

(\*) The TSX CTY 2C module does not offer these functions.

## Downcounting function



**Upcounting function**



---

## 2.2 Functions and timing diagrams

---

### 2.2-1 Invalid value

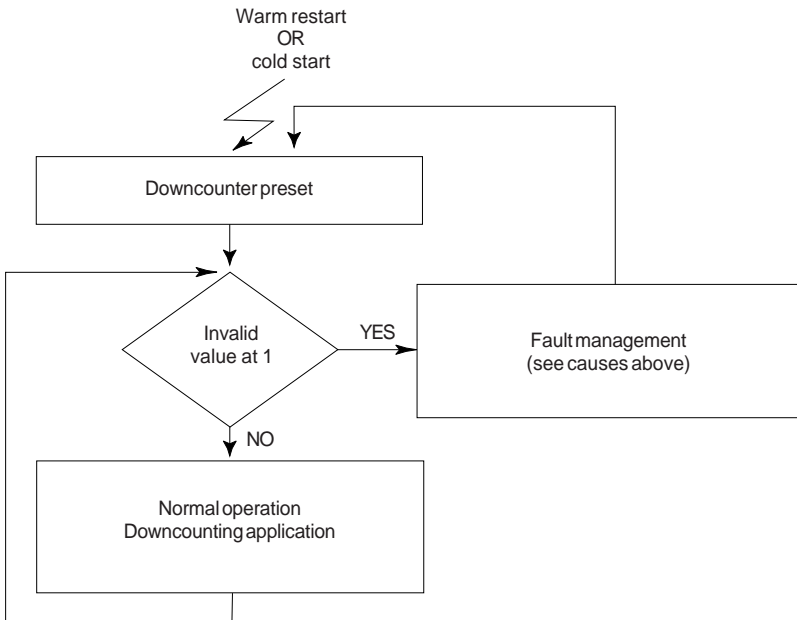
In addition to the diagnostic functions, the user has access to invalid value information. This is used to detect loss of pulses when upcounting or downcounting which may have been caused by :

- a cold start or warm restart of the application,
- an upcounter or downcounter input fault :
  - proximity sensor or encoder power supply fault,
  - encoder line break fault,
- a measurement overrun (upcounter or downcounter capacity).

In this case, word **%IWxy.i.2:X7** is at **state 1**, the content of the upcounter or downcounter cannot be used and the counter outputs (counter output 0 for downcounting or counter outputs 0 and 1 for upcounting) are set to state 0.

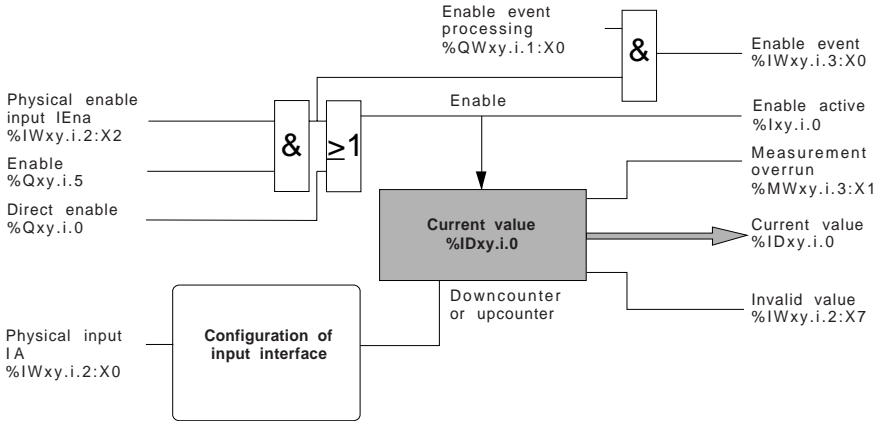
**%IWxy.i.2:X7** is at **state 0** when the downcounter is preset or the upcounter is reset, provided that none of the causes of the loss of pulses are present.

The methodology for managing **invalid values** via the application is as follows :

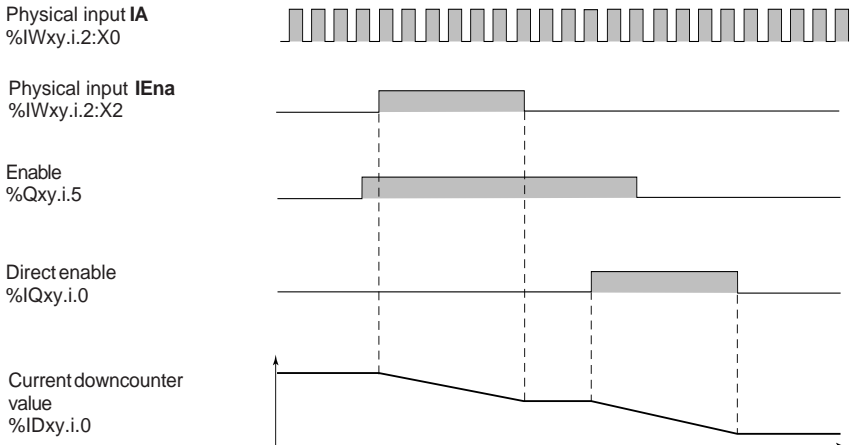


**2.2-2 Enable**

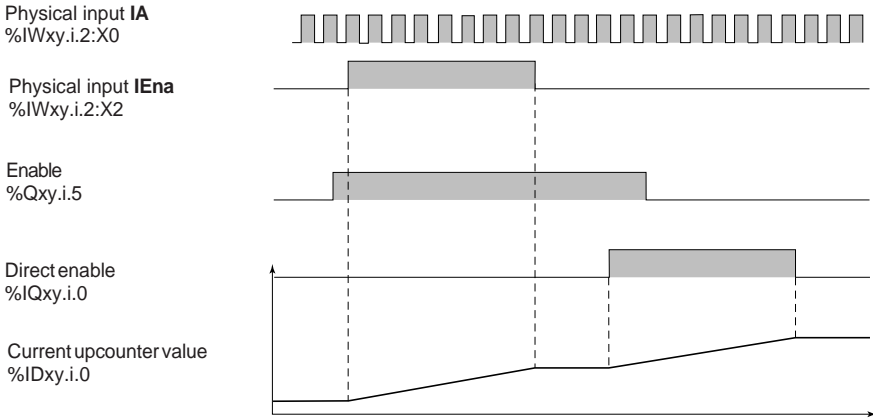
When the upcounter or downcounter is enabled, it can count depending on its physical up or down counter input. The type of contact is selected during configuration : solid state or mechanical contact.



The timing diagram below shows the enable function for the downcounter :



The timing diagram below shows the enable function for the upcounter :



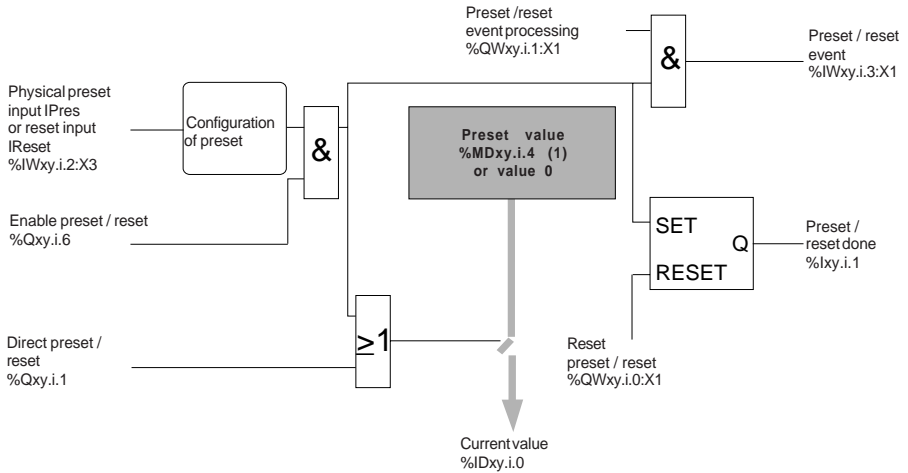
### 2.2-3 Preset or reset

The preset is used to initialize the downcounter to the preset value.

The reset is used to initialize the upcounter to value 0.

This is carried out on the rising or falling edge of input **IPres** (preset) or input **IReset** (reset), depending on the configuration.

The preset or the reset affects the invalid value object (see section 2.2.1).



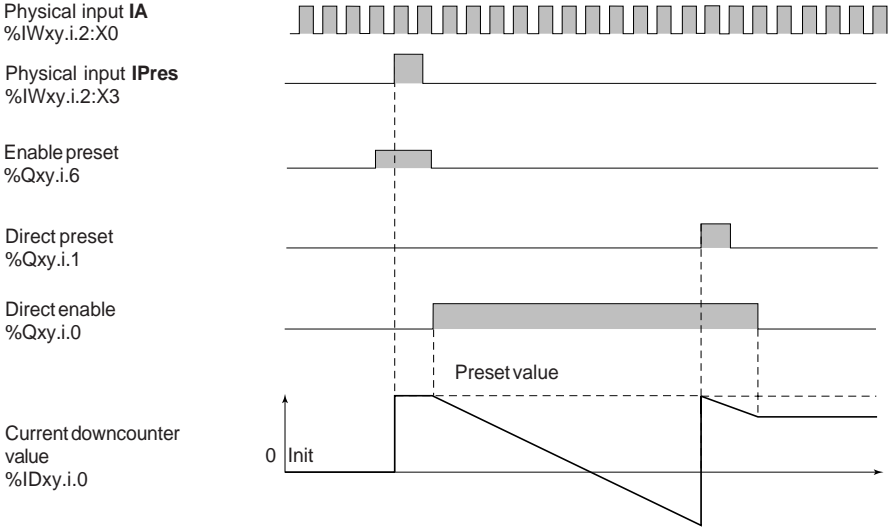
(1) The preset value object **%MDxy.i.4** is managed in accordance with the explicit exchange mechanism (section 8.3).

The configuration of **value 0 crossing** (downcounting) or **high setpoint crossing** (upcounting) enables two preset or reset modes to be defined for setting the current value to 0 or to the high setpoint value :

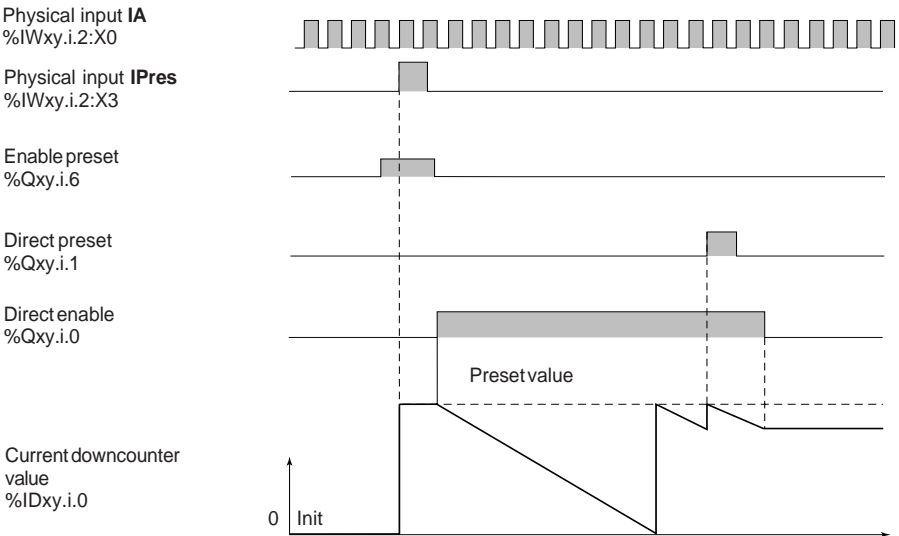
- no downcounter preset or upcounter reset,
- with downcounter preset or upcounter reset.

The following timing diagrams show the various cases for the crossing of value 0 for downcounting or the crossing of the high setpoint for upcounting :

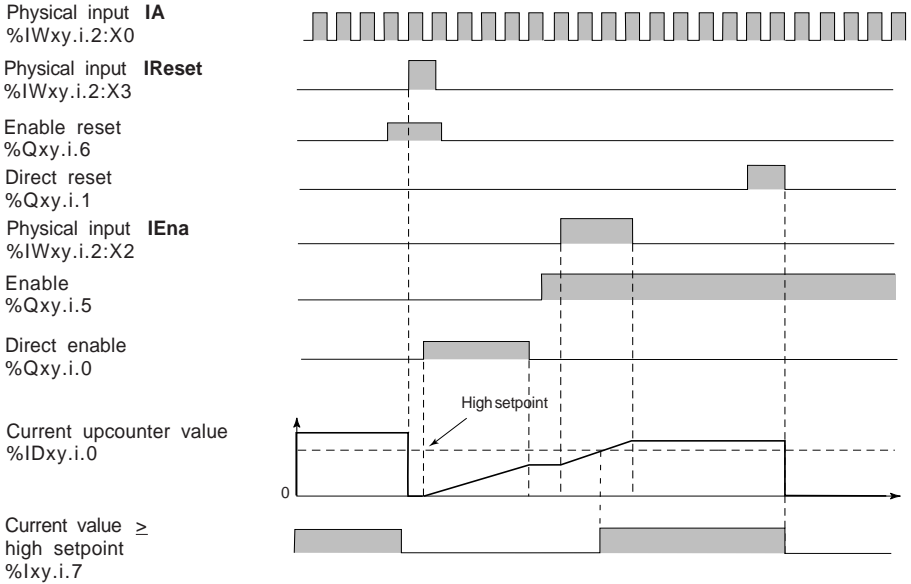
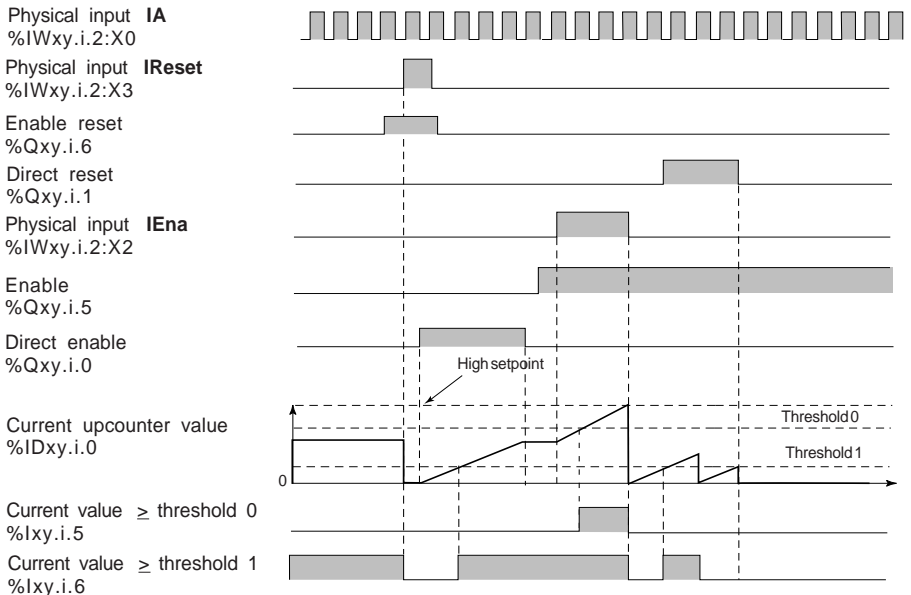
**Value 0 crossing with no downcounter preset :**



**Value 0 crossing with downcounter preset :**



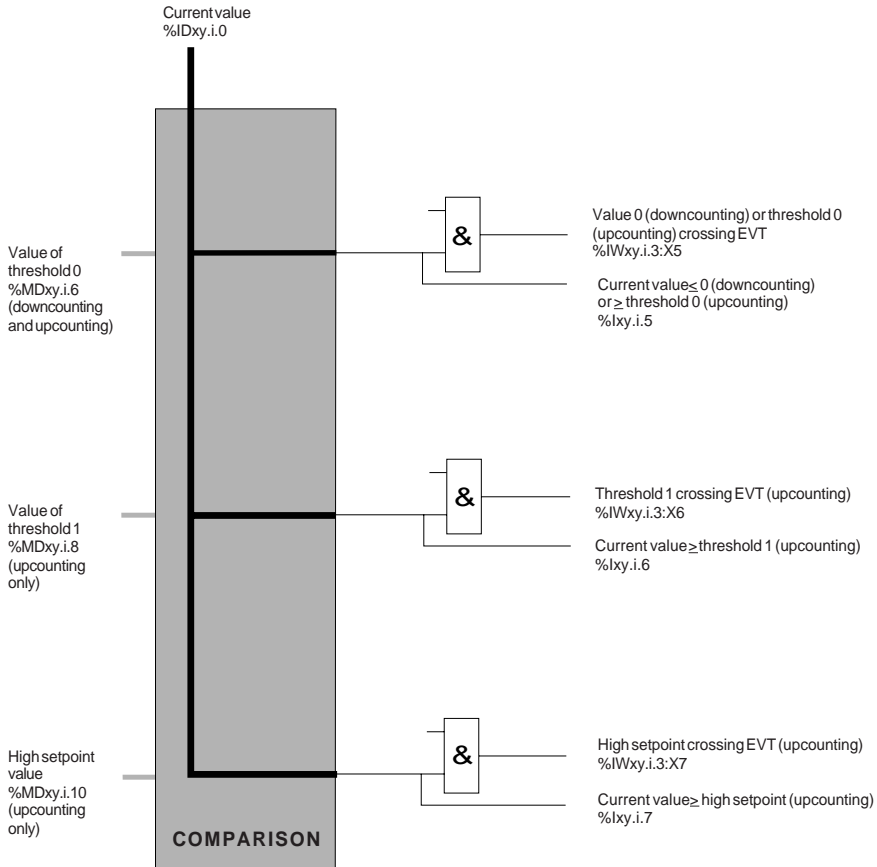


**High setpoint crossing with no upcounter reset:****High setpoint crossing with upcounter reset:**

## 2.2-4 Comparison

The comparison of the current value with value 0 (downcounting) or with thresholds and the high setpoint (upcounting) is given as language objects.

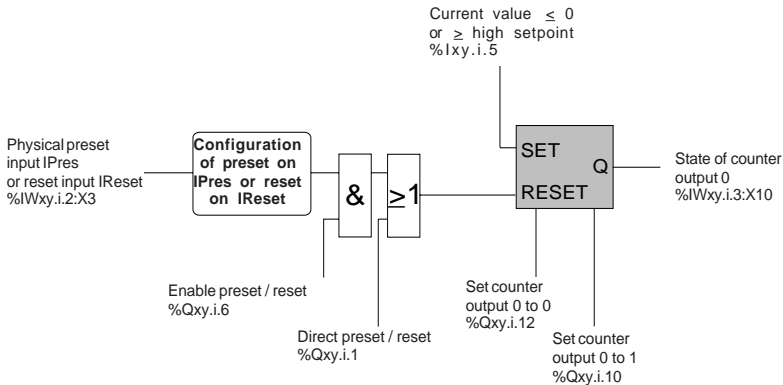
Events can be generated when the current value crosses value 0, threshold 0, threshold 1 or the high setpoint (see section 6, Events).



## 2.2-5 Counter outputs

For counter output 0, the SET and RESET conditions are predefined :

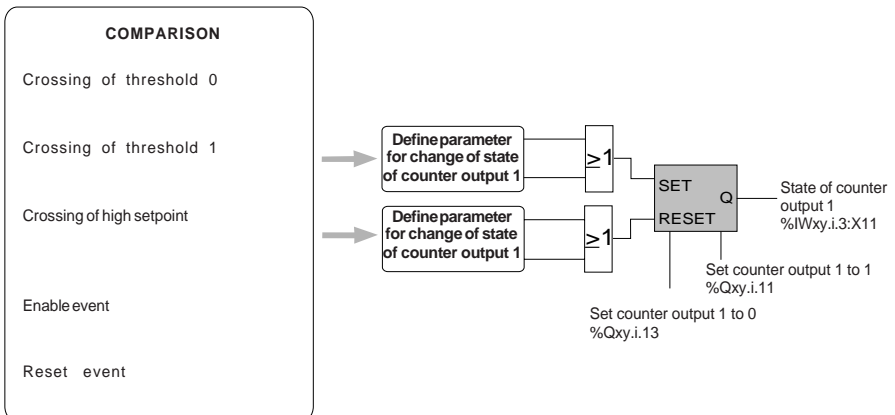
- SET condition :
  - current value  $\leq 0$  for downcounting,
  - current value  $\geq$  high setpoint for upcounting
- RESET condition :
  - direct preset or preset done, for downcounting,
  - direct reset or reset done, for upcounting.



For counter output 1, the parameters of the SET and RESET conditions can be defined in the adjustment screen (see section 5.4).

The SET and RESET input logic of counter output 1 permits 5 combinations of states relating to :

- the crossing of thresholds 0 and 1 or of the high setpoint by the current value of the upcounter,
- a counter enable or reset event.



**Caution**

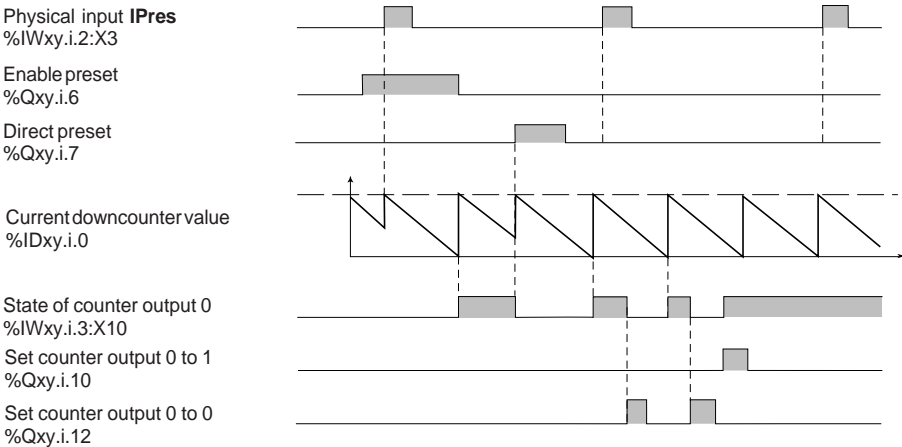
The invalid value language object **%IWxy.i.2:X7** at **state 1** shows that the current value of the downcounter or upcounter cannot be used. The outputs of counter outputs 0 and 1 are then set to 0 (see section 2.2.1).

The levels of priority concerning counter outputs 0 and 1 are as follows :

<b>Counter output 0</b> (downcounting and upcounting)	<b>Counter output 1</b> (upcounting only)
<p><b>Levels of priority</b></p> <p style="text-align: right;">High ↑</p> <p style="text-align: left;">↓ Low</p>	<p><b>Levels of priority</b></p> <p style="text-align: right;">High ↑</p> <p style="text-align: left;">↓ Low</p>
Set to 0	Set to 0
Set to 1	Set to 1
RESET	RESET
SET	SET
	Enable event
	Reset event
	Crossing of high setpoint
	Crossing of threshold 1
	Crossing of threshold 0

The following timing diagrams relate to the SET and RESET conditions of the counter outputs :

**Counter output 0** (predefined conditions)

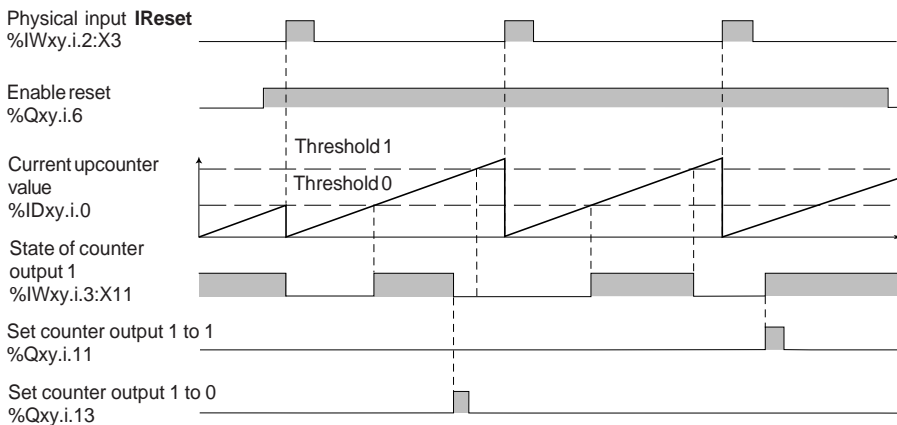


## Description of the upcounting and downcounting functions (TSX CTY 2A/4A) 2

**Counter output 1** (conditions can be defined in the adjustment screen) (upcounting only)

For example :

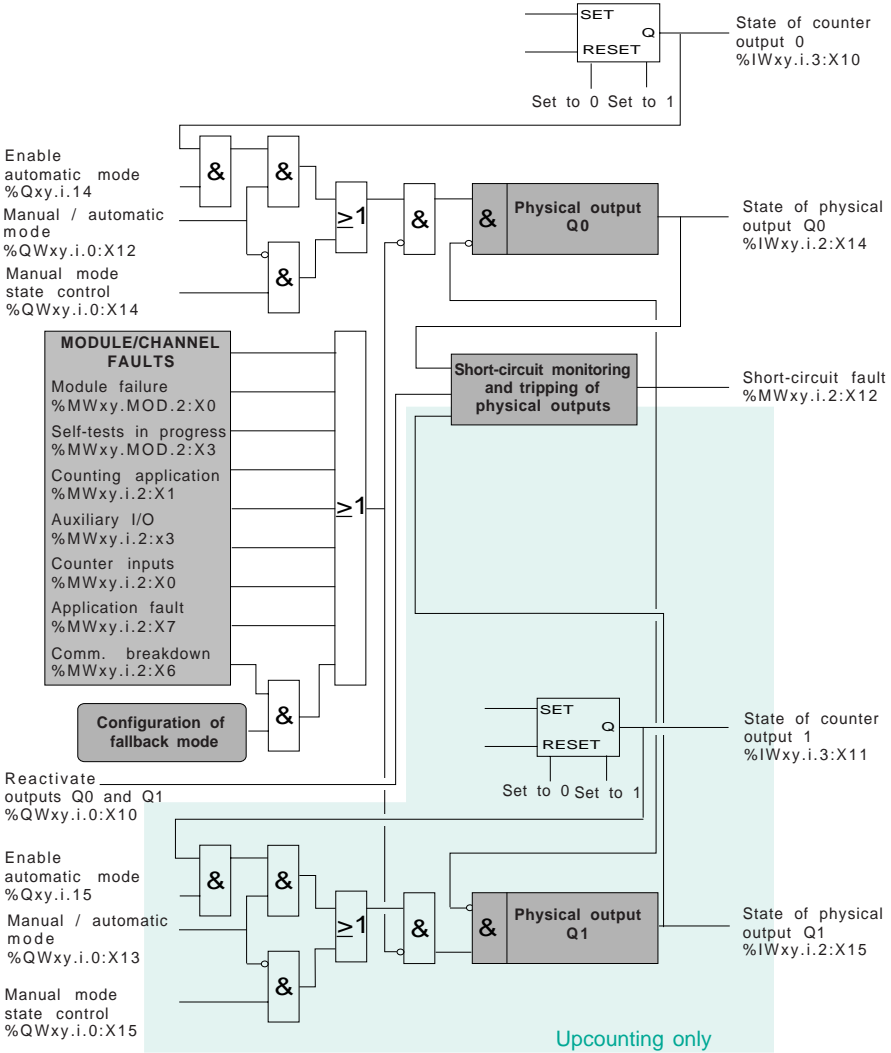
Change of state of counter outputs on	C1
Crossing of threshold 0	S
Crossing of threshold 1	R
Crossing of setpoint	
Enable event	
Reset event	R



### 2.2-6 Physical outputs Q0 and Q1

The state of counter outputs 0 and 1 can be applied directly to two **physical outputs** of the CTY2A / 4A module :

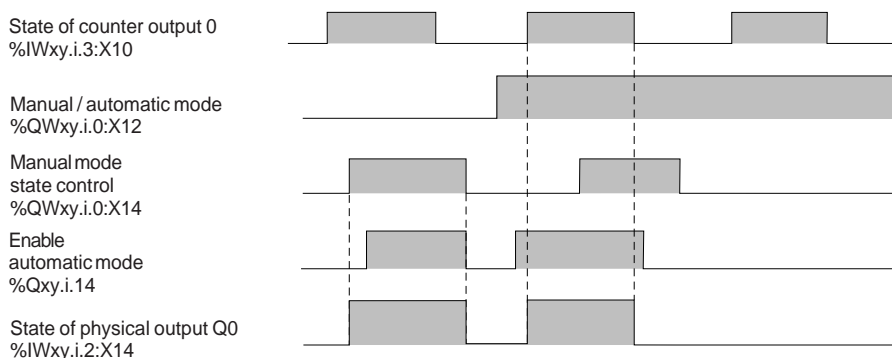
- output **Q0** for the state of counter output 0 (for downcounting and upcounting),
- output **Q1** for the state of counter output 1 (for upcounting only).



The state of the physical outputs Q0 and Q1 can be controlled in two modes :

- manual mode : the state of physical output Q0 or Q1 is defined by controlling the output in manual mode,
- automatic mode : the state of physical output Q0 or Q1 is the same as that of the associated counter output, provided that the output is enabled in automatic mode.

The following timing diagram and table show the operation of the outputs (for example, output Q0) :



Manual / automatic mode	Automatic mode output enable	Manual mode output control	Output state
Manual 0	X	0	0
	X	1	1
Auto 1	0	x	0
	1	x	State of counter output

The state of physical outputs Q0 and Q1 is forced to 0 when the output is tripped or for one of the following faults :

- module failure : %MWxy.MOD.2:X0,
- self-test running : %MWxy.MOD.2:X3,
- counter input fault : %MWxy.i.2:X0,
- counter application fault : %MWxy.i.2:X1,
- auxiliary I/O fault : %MWxy.i.2:X3,
- application fault : %MWxy.i.2:X7,
- communication breakdown : %MWxy.i.2:X6. When this fault occurs :
  - if the fallback mode is configured as **reset**, outputs Q0 and Q1 are forced to 0,
  - if the fallback mode is configured as **maintain**, outputs Q0 and Q1 are maintained in the state they were in before the fault occurred.

---

### **Protection against overloads and short-circuits :**

Physical outputs Q0 and Q1 (solid state outputs) have an internal electronic protection device which is used to detect an overload (typically a current higher than 625 mA) or a short-circuit at 0V (when the output is at state 1). The occurrence of such a fault causes the following :

- current limiting (625 mA),
- setting to 0 of the physical output (%IWxy.i.2:X14 and %IWxy.i.2:X15),
- setting to 1 of the short-circuit fault bit (%MWxy.i.2:X12). This bit is updated after a READ\_STATUS explicit exchange or in debug mode on the counter module channel,
- flashing of the **CH** indicator lamp associated with the counter channel,
- activation of the **I/O** indicator lamp of the counter module, as a steady red light.

The indication of the short-circuit fault disappears 1 second after the effective reactivation of physical output Q0 (the reactivation mode is defined during **configuration**).

### **Reactivation of the physical outputs**

When a fault has caused physical output Q0 or Q1 to trip, it must be reactivated.

As tripping has an adverse effect on the performance of the process controlled by the PLC, it is recommended to condition the reactivation of physical outputs Q0 and Q1 as a manual operation. Before reactivation, it is then possible for the operator to take all the necessary precautions with regard to the control system and personal safety (for example, requesting a transition to manual operation).

**If allowed by the process controlled by the PLC and at the user's own risk, it is possible to program an automatic reactivation.**

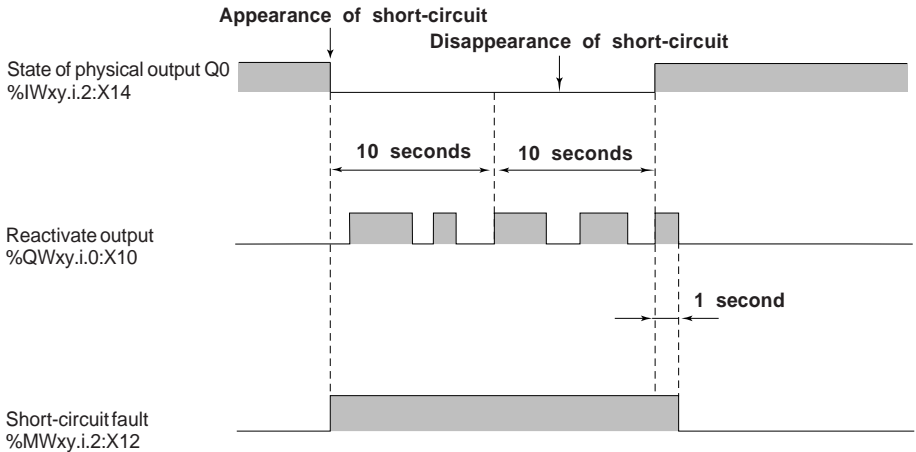
When one of the physical outputs Q0 and Q1 is short-circuited, both the outputs are set to 0 by the counter module. While the short-circuit persists, for safety reasons it is necessary to set physical outputs Q0 and Q1 to 0 by program (in automatic mode, set the automatic mode output enable objects of both physical outputs to 0. In manual mode, set the manual control objects of both physical outputs to 0).



### Manual reactivation of the physical outputs

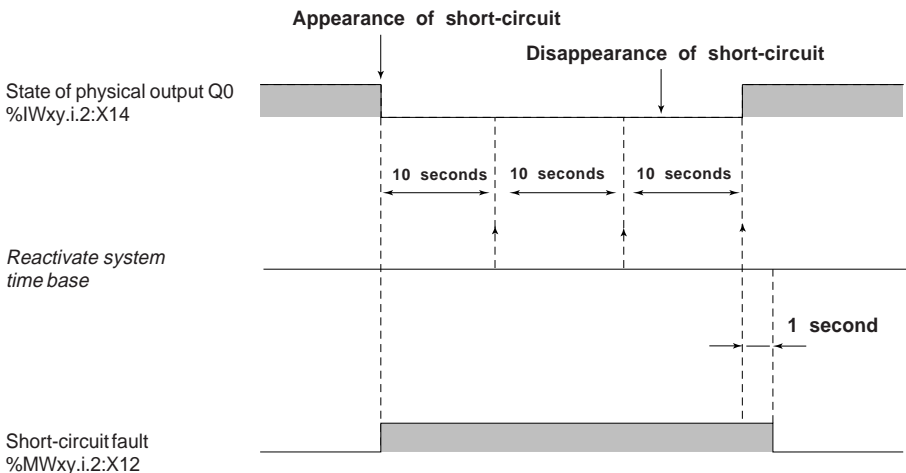
The short-circuit fault bit is set to 1 when the short-circuit appears. It is necessary to activate the output reactivation bit (%QWxy.i.0:X10) to reactivate the physical output on condition that the manual reactivation mode has been configured.

Reactivation will be effective at least 10 seconds after the short-circuit has been detected and when the short-circuit has disappeared.



### Automatic reactivation of the physical outputs

Reactivation is requested automatically by the CTY 2A/4A module every 10 seconds. The time base of 10 seconds is synchronous with the occurrence of the fault.



## 2.2-7 Event processing

The user can associate event processing (reflex action) with an upcounter or downcounter channel during configuration (see setup section).

If they are not masked, several events can activate event processing :

- value 0 crossing, for downcounting (see section 2.2-4),
- threshold or high setpoint crossing, for upcounting (see section 2.2-4),
- up or down counter enable (see section 2.2-2),
- preset or reset (see section 2.2-3).

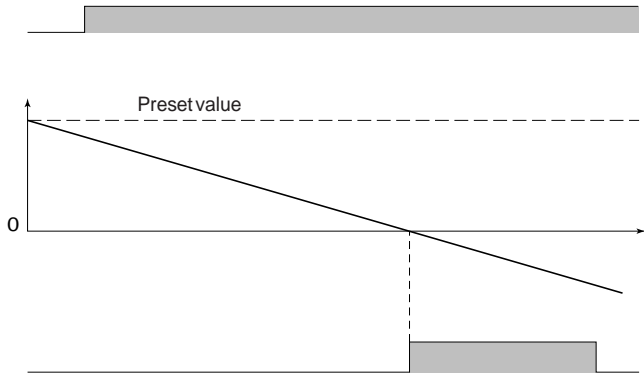
The following timing diagrams give an example of generating internal events in the upcounter or downcounter. During event processing, the user must identify the source of the event by testing the event object for state 1, and then launch the associated reflex action via the application program (see example of event processing in section 6).

### Unmasking of downcounting events

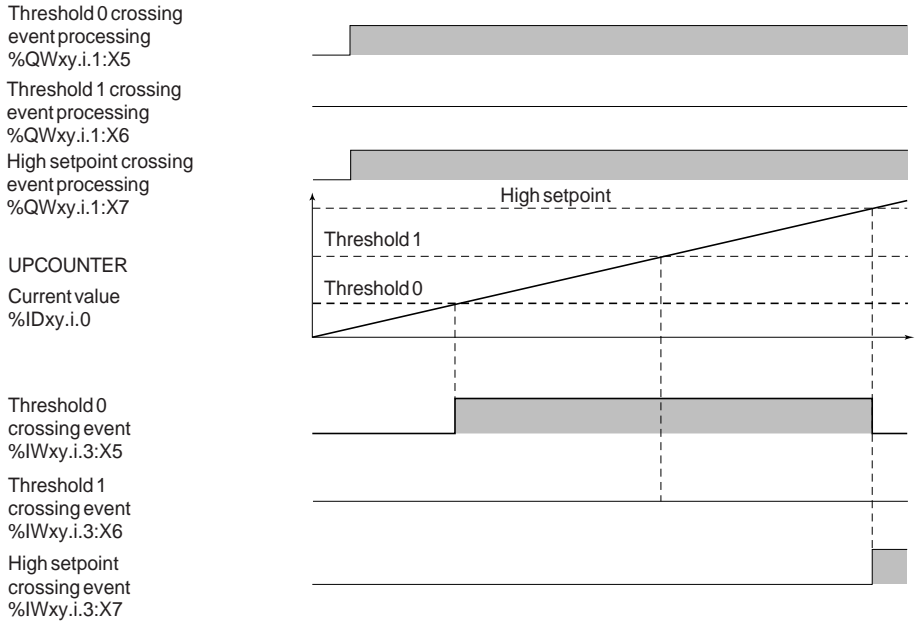
Value 0 crossing  
event processing  
%QWxy.i.1:X5

DOWNCOUNTER  
Current value  
%IDxy.i.0

EVENTS  
Value 0  
crossing event  
%IWxy.i.3:X5



## Unmasking of upcounting events



## 2.3 Description of language objects associated with the upcounting and downcounting functions

The language objects associated with the upcounting and downcounting functions are described in the following tables relating to the **enable**, the **current value**, the **preset** or **reset**, the **comparison**, the **counter outputs** and the **physical outputs**.

ENABLE	objects	description
Enable event	%IWxy.i.3:X0	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the enable done.
Enable active	%lxy.i.0	<b>state 1</b> : the upcounter (or downcounter) is enabled, <b>state 0</b> : the upcounter (or downcounter) is inhibited,
Physical enable input <b>IEna</b>	%IWxy.i.2:X2	represents the state of the physical enable input <b>IEna</b> .
Enable	%Qxy.i.5	<b>state 1</b> : enables the physical upcounter (or downcounter) input <b>IEna</b> , <b>state 0</b> : inhibits the physical upcounter (or downcounter) input <b>IEna</b> ,
Direct enable (by program)	%Qxy.i.0	<b>state 1</b> : enables the upcounter (or downcounter), enabled by the physical input.
Enable event processing	%QWxy.i.1:X0	<b>state 1</b> : the enable done event is not masked, <b>state 0</b> : the enable done event is masked (the event is neither processed nor stored).

Description of the upcounting and downcounting functions (TSX CTY 2A/4A) 2

CURRENT VALUE	objects	description
Current value	<b>%IDxy.i.0</b>	current value of the upcounter (or downcounter). This <b>word</b> can be read and tested. It is between : -16777216 and +16777215 (downcounting function), 0 and +16777215 (upcounting function).
Overrun event	<b>%IWxy.i.3:X15</b>	object to be tested for <b>state 1</b> in the event processing (identification of the event), in order to launch the action associated with an overflow of the stack of PLC events (serious error).
Invalid value	<b>%IWxy.i.2:X7</b>	<b>state 1</b> : the current value of the upcounter (or downcounter) cannot be used, <b>state 0</b> : the current value of the upcounter (or downcounter) can be used.
Measurement overrun	<b>%MWxy.i.3:X1</b>	<b>state 1</b> : the current value of the upcounter (or downcounter) is less than -16777216 (downcounting) or greater than +16777215 (upcounting), <b>state 0</b> : the current value of the upcounter (or downcounter) is between -16777216 and +16777215 (downcounting), or less than +16777215 (upcounting).
Physical upcounter (or downcounter) input <b>IA</b>	<b>%IWxy.i.2:X0</b>	represents the state of the physical upcounter (or downcounter) input <b>IA</b> .

<b>PRESET or RESET</b>	<b>objects</b>	<b>description</b>
Preset value	<b>%MDxy.i.4</b>	<b>word</b> which can be written, read and tested. It is between 0 and +16777215 for downcounting and is 0 for upcounting.
Preset event or reset event	<b>%IWxy.i.3:X1</b>	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the preset (downcounting) or reset (upcounting) done.
Preset done or reset done	<b>%lxy.i.1</b>	<b>state 1</b> : when the downcounter preset or the upcounter reset has been done. The preset or reset condition is defined in the configuration, <b>state 0</b> : on the rising or falling edge of the reset preset or reset.
Physical preset input <b>IPres</b> or reset input <b>IReset</b>	<b>%IWxy.i.2:X3</b>	represents the state of the physical preset input <b>IPres</b> (downcounting) or reset input <b>IReset</b> (upcounting).
Enable preset or enable reset	<b>%Qxy.i.6</b>	<b>state 1</b> : enables the physical preset input <b>IPres</b> (downcounting) or reset input <b>IReset</b> (upcounting), <b>state 0</b> : inhibits the physical preset input <b>IPres</b> (downcounting) or reset input <b>IReset</b> (upcounting).
Direct preset or direct reset (by program)	<b>%Qxy.i.1</b>	<b>on a rising edge</b> : sets the current value of the downcounter to the preset value or the current value of the upcounter to 0.
Reset preset or reset	<b>%QWxy.i.0:X1</b>	<b>on a rising or falling edge</b> : sets the preset done bit to 0 (downcounting) or reset done (upcounting).
Preset or reset event processing	<b>%QWxy.i.1:X1</b>	<b>state 1</b> : the preset done or reset done event is not masked, <b>state 0</b> : the preset done or reset done event is masked (the event is neither processed nor stored).

Description of the upcounting and downcounting functions (TSX CTY 2A/4A) 2

COMPARISON	objects	description
Value of threshold 0 (upcounting only)	%MDxy.i.6	<b>word</b> which can be written, read and tested. This word is between 0 and +16777215.
Value of threshold 1 (upcounting only)	%MDxy.i.8	<b>word</b> which can be written, read and tested. This word is between 0 and +16777215.
Value of high setpoint	%MDxy.i.10	<b>word</b> which can be written, read and tested. This word is between 0 and +16777215.
Current value $\leq$ 0 (downcounting)	%lxy.i.5	<b>state 1</b> : the current value of the upcounter (or downcounter) is less than or equal to 0 (downcounting) or greater than or equal to the value of threshold 0 (upcounting),
Current value $\geq$ threshold 0 (upcounting)		<b>state 0</b> : the current value of the upcounter (or downcounter) is greater than 0 (downcounting) or less than the value of threshold 0 (upcounting).
Current value $\geq$ threshold 1 (upcounting only)	%lxy.i.6	<b>state 1</b> : the current upcounter value is greater than or equal to the value of threshold 1, <b>state 0</b> : the current upcounter value is less than the value of threshold 1.
Current value $\geq$ high setpoint (upcounting only)	%lxy.i.7	<b>state 1</b> : the current upcounter value is greater than or equal to the value of the high setpoint, <b>state 0</b> : the current upcounter value is less than the value of the high setpoint.
Value 0 or threshold 0 crossing event	%IWxy.i.3:X5	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the crossing of value 0 (downcounting) or threshold 0 (upcounting).
Threshold 1 crossing event (upcounting only)	%IWxy.i.3:X6	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the crossing of threshold 1.

COMPARISON (cont.)	objects	description
High setpoint crossing event (upcounting only)	%IWxy.i.3:X7	object to be tested for <b>at state 1</b> in the event processing (identification of the event) in order to launch the action associated with crossing the high setpoint.
Value 0 (downcounting) or threshold 0 (upcounting) crossing event processing	%QWxy.i.1:X5	<b>state 1</b> : the crossing of value 0 (downcounting) or of threshold 0 (upcounting) event is not masked, <b>state 0</b> : the crossing of value 0 (downcounting) or of threshold 0 (upcounting) event is masked (the event is neither processed nor stored).
Threshold 1 crossing event processing (upcounting only)	%QWxy.i.1:X6	<b>state 1</b> : the threshold 1 crossing event is not masked, <b>state 0</b> : the threshold 1 crossing event is masked (the event is neither processed nor stored).
High setpoint event processing (upcounting only)	%QWxy.i.1:X7	<b>state 1</b> : the high setpoint crossing event is not masked, <b>state 0</b> : the high setpoint crossing event is masked (the event is neither processed nor stored).
COUNTER OUTPUTS	objects	description
State of counter output 0	%IWxy.i.3:X10	gives the current state of counter output 0
State of counter output 1 (upcounting only)	%IWxy.i.3:X10	gives the current state of counter output 1
Set counter output 0 to 1	%Qxy.i.10	<b>state 1</b> : sets counter output 0 to 1.
Set counter output 1 to 1 (upcounting only)	%Qxy.i.11	<b>state 1</b> : sets counter output 1 to 1.
Set counter output 0 to 0	%Qxy.i.12	<b>state 1</b> : sets counter output 0 to 0.
Set counter output 1 to 0 (upcounting only)	%Qxy.i.13	<b>state 1</b> : sets counter output 1 to 0.



Description of the upcounting and downcounting functions (TSX CTY 2A / 4A) 2

PHYSICAL OUTPUTS Q0 - Q1	objects	description
Enable automatic mode for output Q0	<b>%Qxy.i.14</b>	<b>state 1</b> : the state of physical output Q0 follows the state of counter output 0 in automatic mode (on condition that the output is not tripped and the CTY 2A / 4A module is not faulty), <b>state 0</b> : output Q0 is at state 0.
Enable automatic mode for output Q1 (upcounting only)	<b>%Qxy.i.15</b>	<b>state 1</b> : the state of physical output Q1 follows the state of counter output 1 in automatic mode (on condition that the output is not tripped and the CTY 2A / 4A module is not faulty), <b>state 0</b> : output Q1 is at state 0.
Manual/automatic mode control of output Q0	<b>%QWxy.i.0:X12</b>	<b>state 1</b> : output Q0 is controlled in automatic mode, <b>state 0</b> : output Q0 is controlled in manual mode.
Manual/automatic mode control of output Q1 (upcounting only)	<b>%QWxy.i.0:X13</b>	<b>state 1</b> : output Q1 is controlled in automatic mode, <b>state 0</b> : output Q1 is controlled in manual mode.
Manual mode state control of output Q0	<b>%QWxy.i.0:X14</b>	<b>state 1</b> : the state of physical output Q0 is at 1 (on condition that the output is not tripped and the CTY 2A / 4A module is not faulty), <b>state 0</b> : output Q0 is at state 0.
Manual mode state control of output Q1 (upcounting only)	<b>%QWxy.i.0:X15</b>	<b>state 1</b> : the state of physical output Q1 is at 1 (on condition that the output is not tripped and the CTY 2A / 4A module is not faulty), <b>state 0</b> : output Q1 is at state 0.

PHYSICAL OUTPUTS Q0 - Q1 (cont.)	objects	description
State of physical output Q0		<p><b>%IWxy.i.2:X14 manual mode</b> : the state of physical output Q0 is the same as the manual mode output state control (%QWxy.i.0:X15),</p> <p><b>automatic mode</b> : the state of physical output Q0 is the same as the state of counter output 0 (%IWxy.i.3:X10), when enabled, the output is at 1; otherwise, the output is at 0.</p> <p>The output is forced to state 0 when it is tripped or when the CTY 2A / 4A module is faulty.</p>
State of physical output Q1 (upcounting only)		<p><b>%IWxy.i.2:X15 manual mode</b> : the state of physical output Q1 is the same as the manual mode output state control (%QWxy.i.0:X15),</p> <p><b>automatic mode</b> : the state of physical output Q1 is the same as the state of counter output 1 (%IWxy.i.3:X10), when enabled, the output is at 1; otherwise, the output is at 0.</p> <p>The output is forced to state 0 when it is tripped or when the CTY 2A / 4A module is faulty.</p>
Short-circuit fault outputs Q0 and Q1 (upcounting only)	<b>%MWxy.i.2:X12</b>	<p><b>at state 1</b> : short-circuit on physical output Q0 or Q1,</p> <p><b>at state 0</b> : no short-circuits on physical outputs Q0 and Q1.</p>
Reactivate outputs Q0 and Q1 (upcounting only)	<b>%QWxy.i.0:X10</b>	<p><b>on a rising edge</b> : reactivation of the circuit-breaker for physical outputs Q0 and Q1.</p>

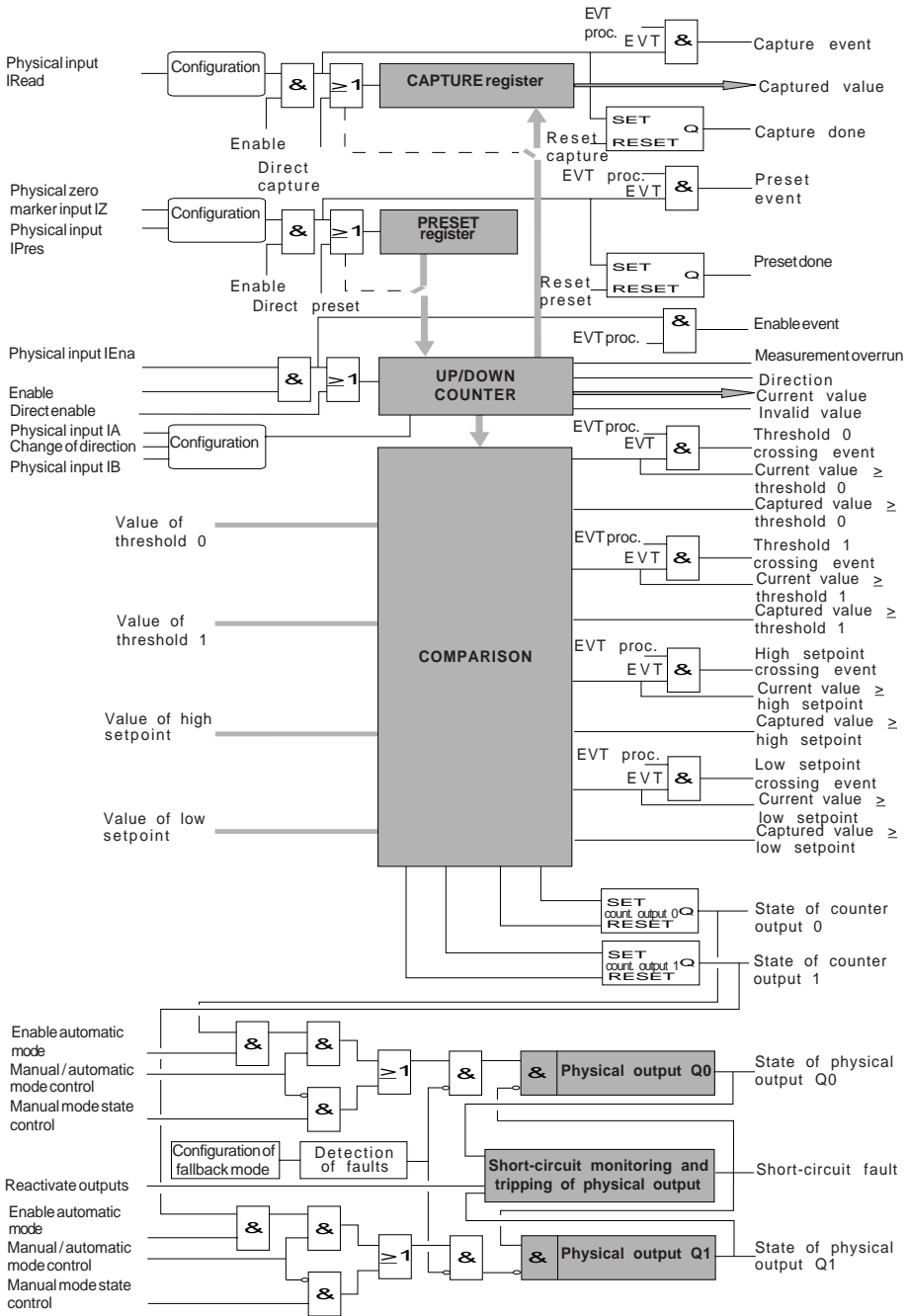
### 3 Description of up/down counting function (TSX CTY 2A / 4A)

#### 3.1 Overview

---

The function of TSX CTY 2A / 4A modules is used to up/down count pulses from a sensor (24 bits + sign), between the values -16 777 216 and + 16 777 215.

- the input interface offers four up/down counting configuration options :
  - **IA upcounts** and **IB downcounts**,
  - **IA up/down counts** and **IB** gives the **direction**,
  - **IA up/down counts** and **the application** gives the **direction**,
  - use of an **incremental encoder**.
- The up/down counter can be enabled by a physical input or by program. It can activate event processing.
- The current up/down counter value is only accessible in read mode.
- The up/down counter is preset by a physical input or by program. It is configurable (7 modes) and can activate event processing.
- The contents of the up/down counter are captured in the capture register by a physical input or by program. This is configurable (2 modes) and can activate event processing.
- Event processing can also be activated when the current value crosses the **high setpoint**, the **low setpoint**, **threshold 0** or **1**.
- This up/down counting function also offers two reflex counter outputs which can be applied directly to one physical output associated with the channel and physically situated on the TSX CTY 2A / 4A modules.  
For counter outputs 0 and 1, the parameters of the SET and RESET conditions (17 conditions) can be defined in the adjustment screen.



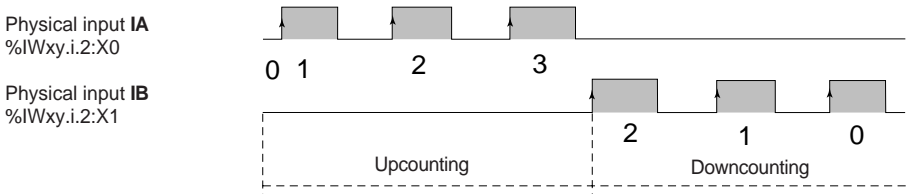
## 3.2 Functions and timing diagrams

### 3.2-1 Input interface

The input interface offers four up/down counting configuration options :

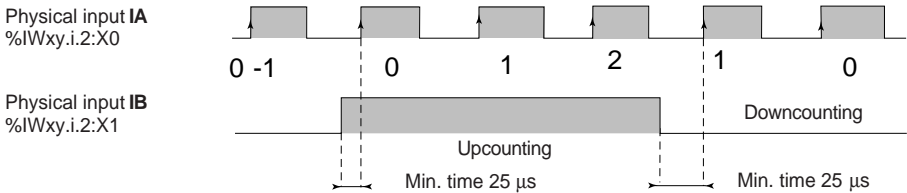
- **IA upcounts, IB downcounts :**

Pulses are taken into account by the up/down counter on the rising edges of 2 physical inputs IA and IB. Physical input **IA** is used to increment the up/down counter (upcounting) and physical input **IB** is used to decrement it (downcounting). If the pulses on inputs IA and IB are simultaneous, the up/down counter does not count.



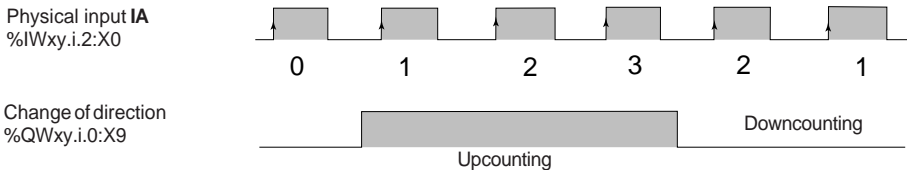
- **IA up/down counts, IB direction :**

Physical input **IB** defines the up/down counting direction which is carried out on the rising edges of pulses received on physical input **IA**.



- **IA up/down counts, application direction :**

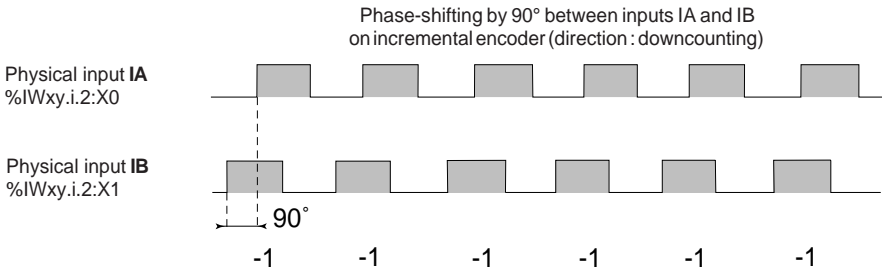
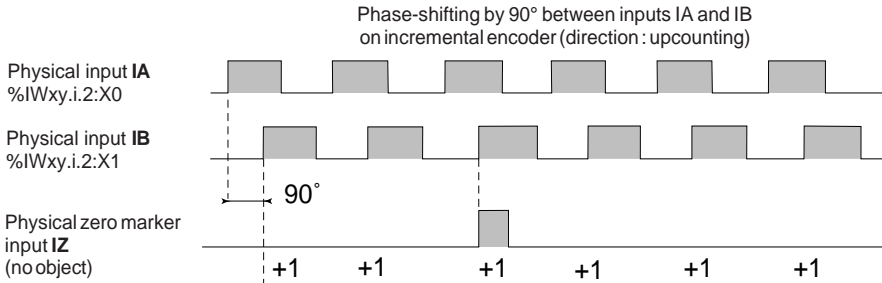
The **change direction** bit defines the up/down counting direction which is carried out on the rising edges of pulses received on physical input **IA**.



- **Incremental encoder**

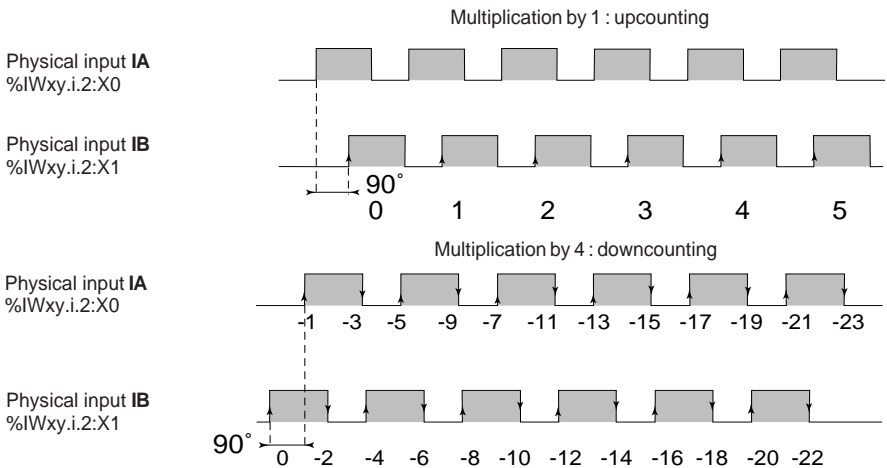
In this operating mode physical inputs **IA** and **IB** are connected to an incremental encoder which supplies two signals (pulses) IA and IB phase-shifted by  $90^\circ$ . Phase-shifting between inputs IA and IB determines the direction of rotation.

The incremental encoder also provides **zero marker** information on input **IZ**, which enables the up/down counter to be preset.



There are two options in the configuration screen :

- **line check** for an encoder with an RS 422 / 485 link. If this option is configured, the PLC signals a fault if a line break in the encoder cable is detected on one of the physical inputs IA, IB or IZ. Application processing corresponding to the fault can then be carried out.
- **multiplication by 4** which is used to improve the precision of the encoder. When the encoder is configured for multiplication by 1, up/down counting is performed on the rising edges of physical input IB; for multiplication by 4, up/down counting is performed on all rising and falling edges of physical inputs IA and IB.



#### Note

The type of contact (solid state or mechanical) is also selected during configuration of the input interface.

### 3.2-2 Invalid value

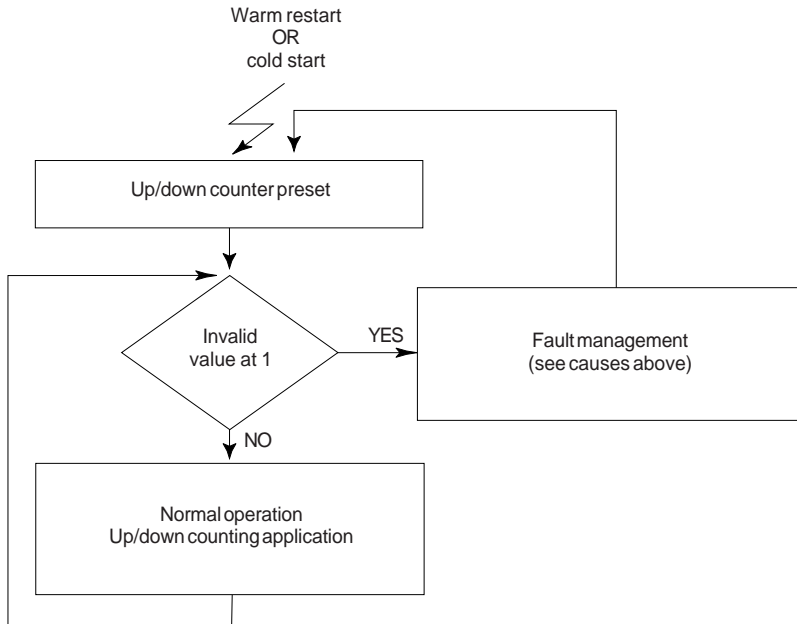
In addition to the diagnostic functions, the user has access to invalid value information. This is used to detect loss of pulses when up/down counting which may have been caused by :

- a cold start or warm restart of the application,
- an up/down counter input fault :
  - proximity sensor or encoder power supply fault,
  - encoder line break fault,
- a measurement overrun (up/down counter capacity).

In this case, word%IWxy.i.2:X7 is at **state 1**, the contents of the up/down counter cannot be used and the counter outputs are set to 0.

%IWxy.i.2:X7 is at **state 0** when the up/down counter is preset, provided that none of the causes of the loss of pulses are present.

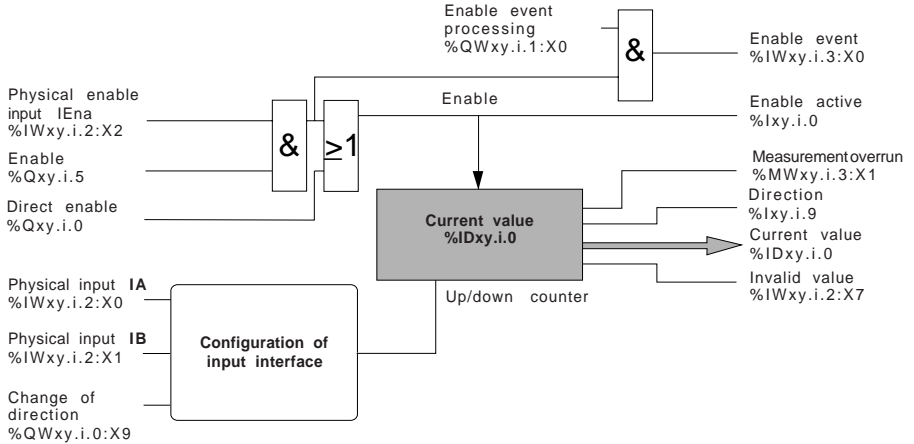
The methodology for managing **invalid values** via the application is as follows :



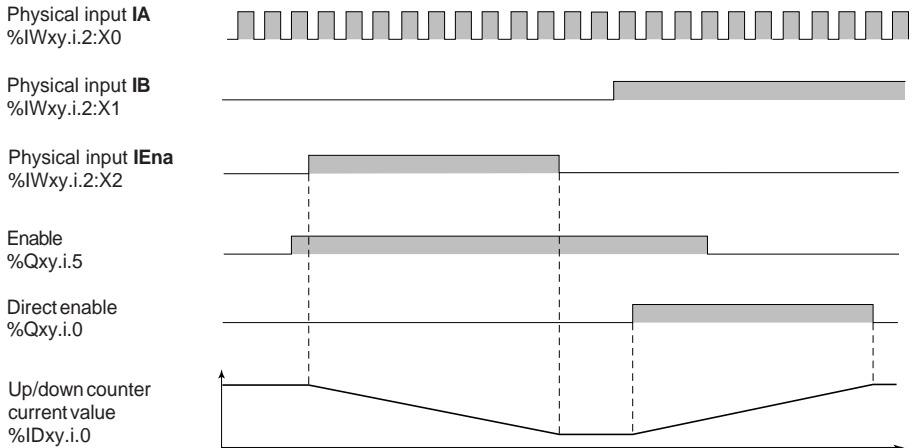


### 3.2-3 Enable

When the up/down counter is enabled, it can count in both directions depending on the physical up/down counter inputs.



The timing diagram below shows the enable function for the up/down counter :

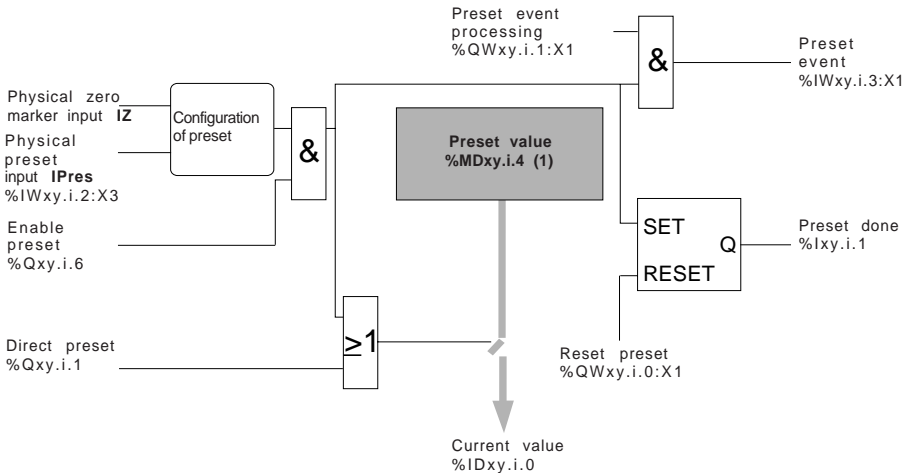


### 3.2-4 Preset

The preset is used to initialize the up/down counter to the preset value. The up/down counter can be preset in 7 modes which are combinations relating to the states, and/or the edges of physical inputs IPres and IZ :

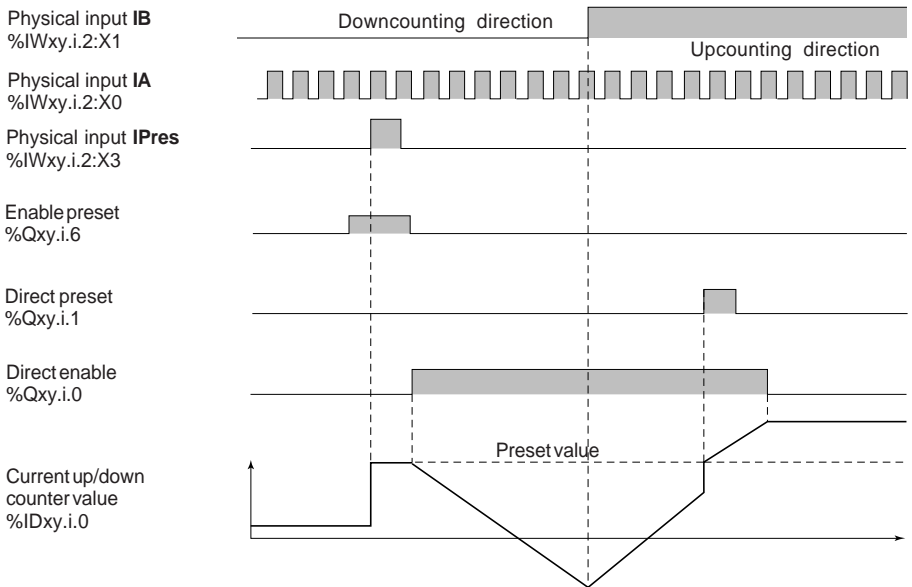
- rising edge of **IPres**,
- falling edge of **IPres**,
- rising edge of **IPres** + direction / falling edge of **IPres** - direction,
- rising edge of **IPres** - direction / falling edge of **IPres** + direction,
- state of **IPres**,
- short cam reference point (see diagram below),
- long cam reference point (see diagram below).

The preset affects the invalid value object (see section 3.2.1).

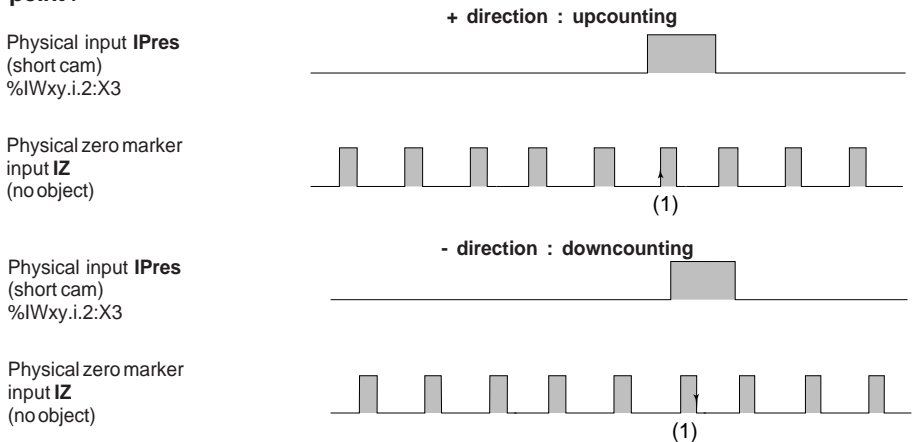


(1) The preset value object %MDx.i.4 is managed in accordance with the explicit exchange mechanism (section 8.3).

The following timing diagram shows the preset mode on the **rising edge of IPres**:



The following timing diagram shows the preset mode on the **short cam reference point** :



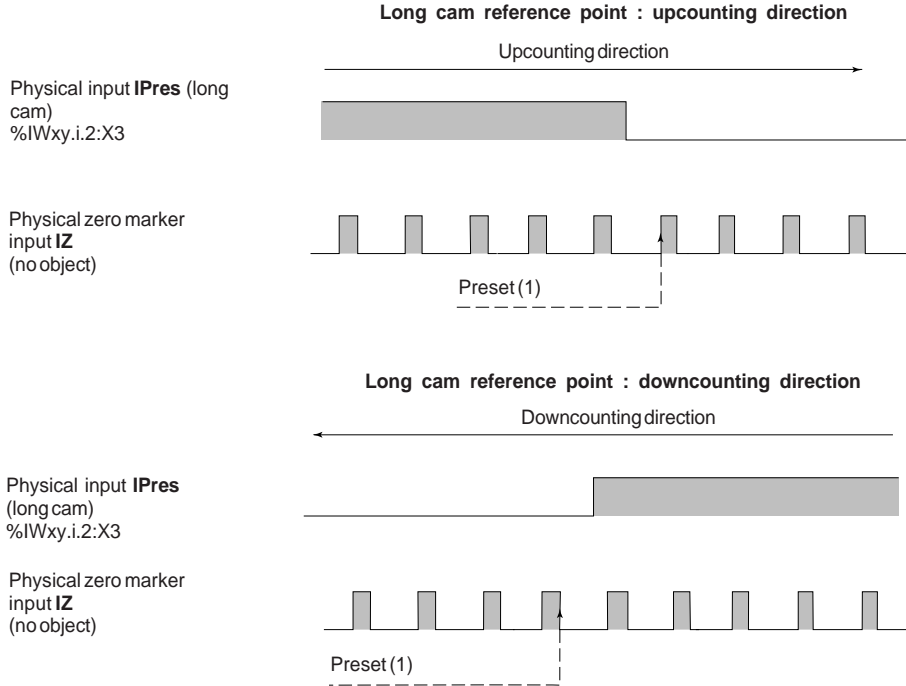
(1) The preset is taken into account :

- in the + direction (upcounting) : input **IPres** at state 1, rising edge of zero marker input **IZ** and software enable,
- in the - direction (downcounting) : input **IPres** at state 1, falling edge of zero marker input **IZ** and software enable,

## Note

In principle, if the short cam is less than one revolution of the incremental encoder, the zero marker will only appear once in the cam.  
If however there are several revolutions of the incremental encoder in the cam, the last active edge of the zero marker signal triggers a preset.

The following timing diagram shows the preset mode on the **long cam reference point** :

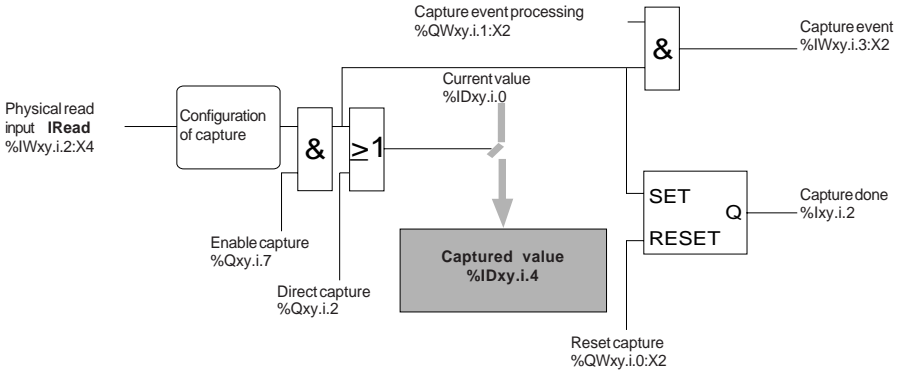


- (1) The preset is taken into account on the first rising edge of the zero marker input **IZ** which follows the changeover to state 0 of input **IPres**, for both the upcounting and the downcounting direction, and software enable.

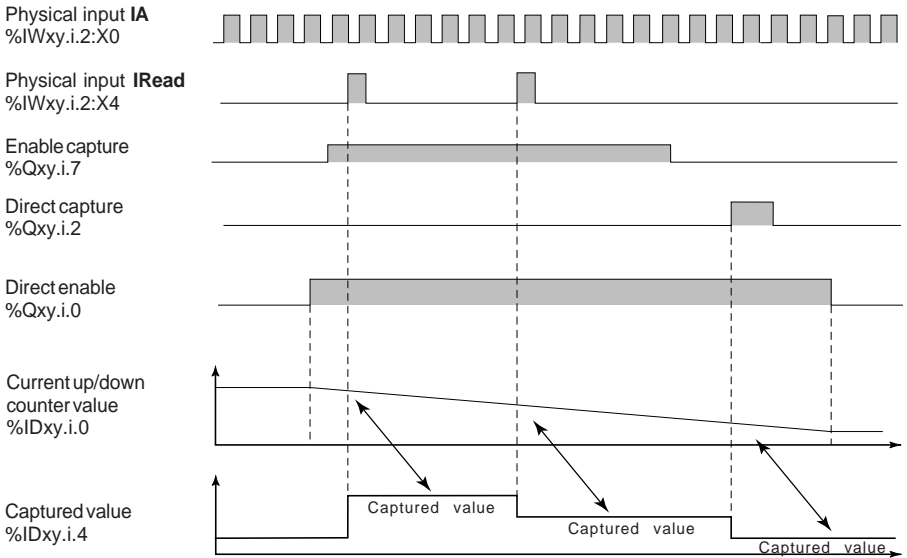
### 3.2-5 Capture (read input)

The capture function is used to copy the current up/down counter value to the capture register. The current value is captured :

- on a change of state of input **IRead** and software enable. In this case, the user must define whether the capture is carried out on a rising or falling edge,
- by program.



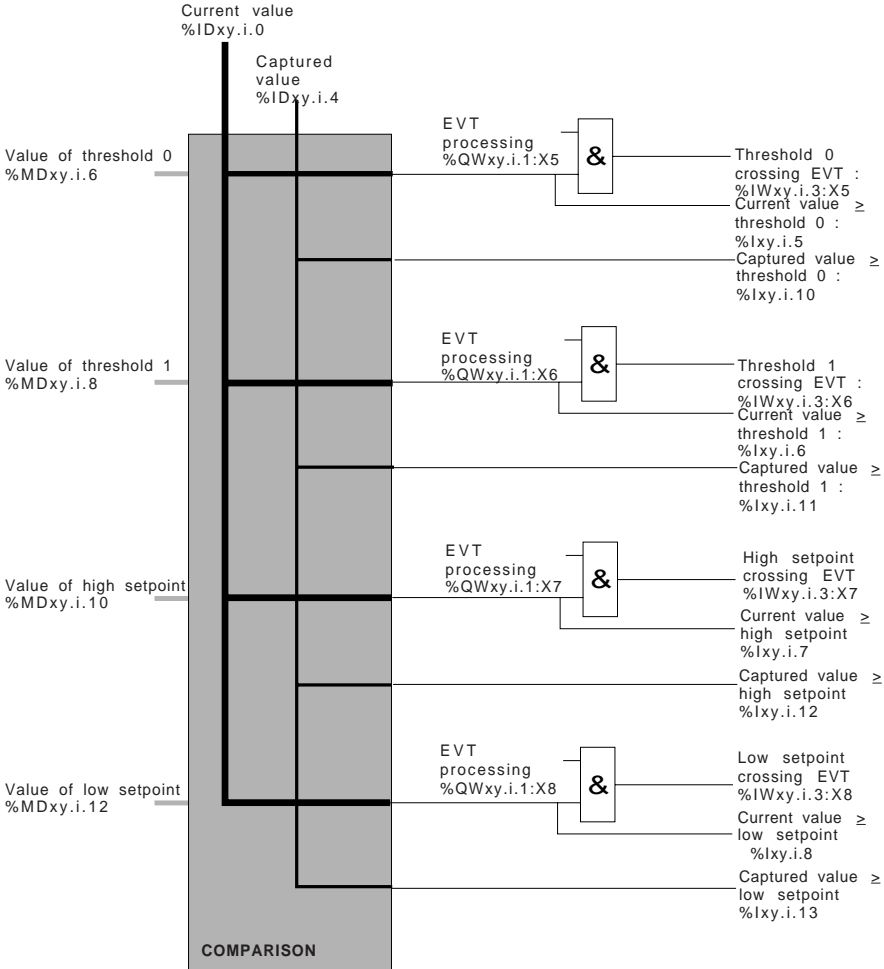
The following timing diagram shows the capture mode on the **rising edge of IRead** :



### 3.2-6 Comparison

The comparisons of the captured value and the current value with the thresholds and setpoints are given as language objects.

The crossing of thresholds and setpoints can generate events (see section 6).

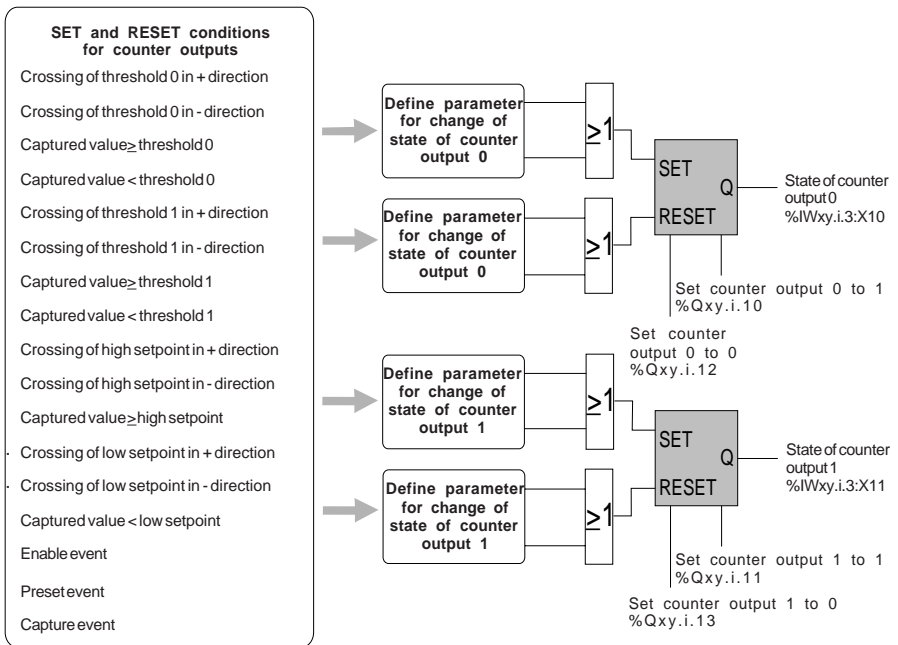


### 3.2-7 Counter outputs

For each counter output, the parameters of the SET and RESET conditions can be defined (see section 5.5).

The SET and RESET input logic allow 17 combinations of states relating to :

- the crossing of thresholds and setpoints by the current up/down counter value,
- the positions of the captured value in relation to the thresholds and setpoints,
- the up/down counter enable, preset and capture events.



#### Caution

**%IWxy.i.2:X7** (invalid value) at **state 1** shows that the content of the up/down counter cannot be used and counter outputs 0 and 1 are set to 0 (see section 3.2.2).

The guidelines for the levels of priority concerning counter outputs 0 and 1 are as follows :

**Counter outputs 0 and 1**

<p>Set to 0</p> <p>Set to 1</p> <p>RESET</p> <p>SET</p> <p>Enable event</p> <p>Position of the captured value in relation to the low setpoint</p> <p>Position of the captured value in relation to the high setpoint</p> <p>Position of the captured value in relation to threshold 1</p> <p>Position of the captured value in relation to threshold 0</p> <p>Crossing of low setpoint</p> <p>Crossing of high setpoint</p> <p>Crossing of threshold 1</p> <p>Crossing of threshold 0</p> <p>Preset event</p>	<p><b>Levels of priority</b></p> <p>High</p> <p>↑</p> <p>↓</p> <p>Low</p>
---	---

The adjustment screen below shows an example of the parameters for counter outputs 0 and 1 of an up/down counter channel :

**Preget value**

Initial value

**Threshold value**

Threshold0

Initial value

Threshold1

Initial value

**Setpoint Values**

High

Initial value

Low

Initial value

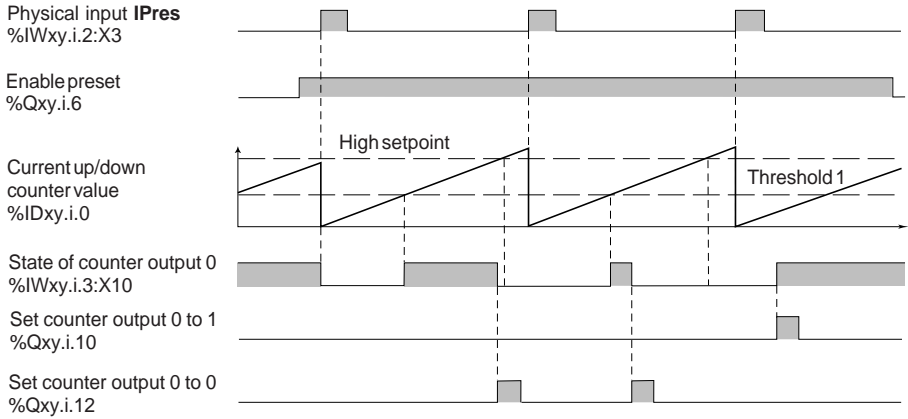
**Counter output state**

Change counter output state on:	C0	C0i	C1	C1i
Crossing of threshold0 in +direction	R	R		
Crossing of threshold0 in -direction				
Captured value >= Threshold0				
Captured value < Threshold0				
Crossing of threshold1 in +direction				
Crossing of threshold 1 in -direction				
Captured value >= Threshold1				
Captured value < Threshold1				
Crossing of high setpoint in +direction				
Crossing of High Setpoint in -direction				
Captured value >= High setpoint			S	S
Crossing of low setpoint in +direction				
Crossing of low setpoint in -direction				

Actions:

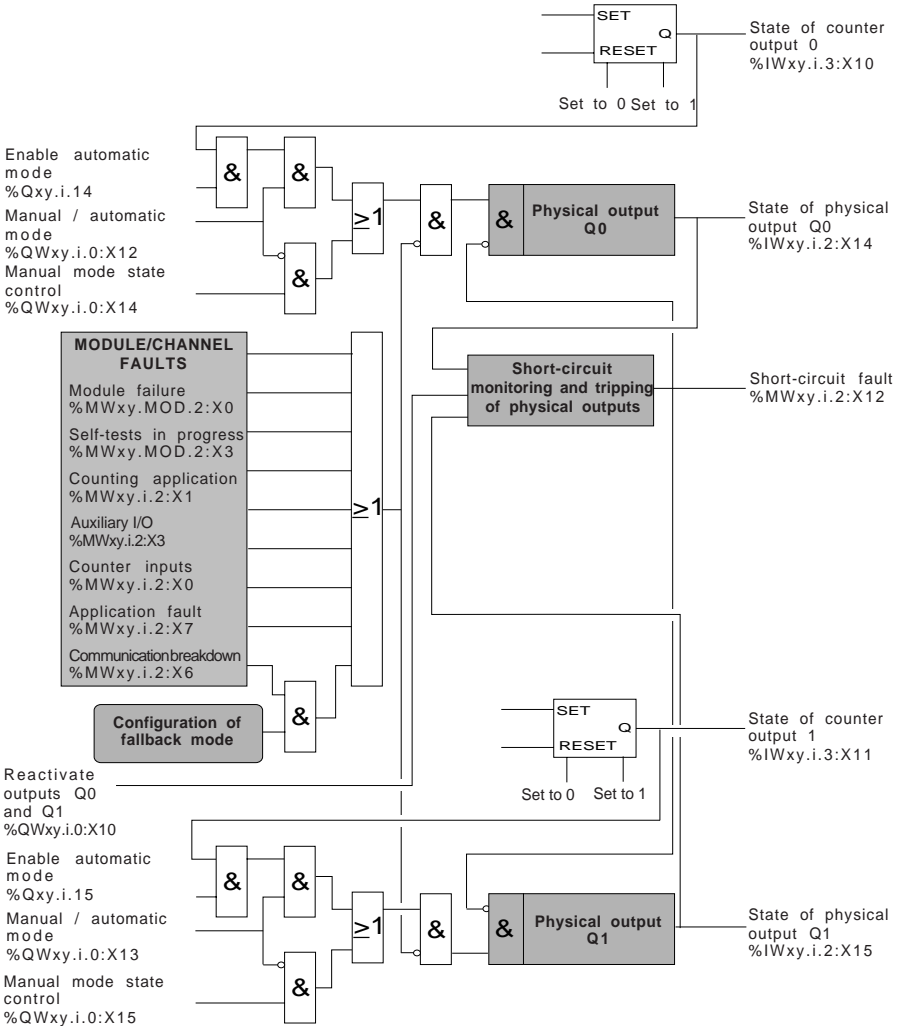


The timing diagram below shows the SET and RESET conditions defined in the previous adjustment screen for counter output 0 :



### 3.2-8 Physical outputs Q0 and Q1

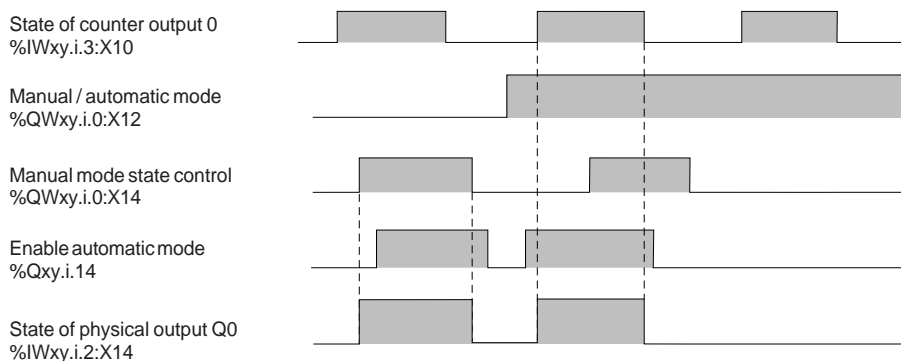
The state of counter outputs 0 and 1 can be applied directly to two **physical outputs** of the CTY 2A / 4A module (output **Q0** for the state of counter output 0 and output **Q1** for the state of counter output 1).



The state of the physical outputs Q0 and Q1 can be controlled in two modes :

- manual mode : the state of output Q0 or Q1 is defined by controlling the output in manual mode,
- automatic mode : the state of output Q0 or Q1 is the same as the state of the associated counter output, on condition that the output is enabled in automatic mode.

The following timing diagram and table show the operation of the outputs (for example, output Q0) :



Manual / automatic mode	Automatic mode enable	Manual mode state control	Output state
Manual 0	X	0	0
	X	1	1
Auto 1	0	X	0
	1	x	State of counter output

The state of physical outputs Q0 and Q1 is forced to 0 when the output is tripped or for one of the following faults :

- module failure : %MWxy.MOD.2:X0,
- self-test running : %MWxy.MOD.2:X3,
- counter input fault : %MWxy.i.2:X0,
- counter application fault : %MWxy.i.2:X1,
- auxiliary I/O fault : %MWxy.i.2:X3,
- application fault : %MWxy.i.2:X7,
- communication breakdown : %MWxy.i.2:X6. When this fault occurs :
  - if the fallback mode is configured as **reset**, outputs Q0 and Q1 are forced to 0,
  - if the fallback mode is configured as **maintain**, outputs Q0 and Q1 are maintained in the state they were in before the fault occurred.

---

### **Protection against overloads and short-circuits :**

Physical outputs Q0 and Q1 (solid state outputs) have an internal electronic protection device which is used to detect an overload (typically a current higher than 625 mA) or a short-circuit at 0V (when the output is at state 1). The occurrence of such a fault causes the following :

- current limiting (625 mA),
- setting physical outputs Q0 and Q1 to 0 (%IWxy.i.2:X14 and %IWxy.i.2:X15),
- setting to 1 of the short-circuit fault bit (%MWxy.i.2:X12). This object is updated after a READ\_STATUS explicit exchange or in debug mode on the counter channel,
- flashing of the **CH** indicator lamp associated with the counter channel,
- activation of the **I/O** indicator lamp of the counter module, as a steady red light.

The indication of the short-circuit fault disappears 1 second after the effective reactivation of physical output Q0 (the reactivation mode is defined during **configuration**).

### **Reactivation of the physical outputs**

When a fault has caused physical output Q0 or Q1 to trip, it must be reactivated.

As tripping has an adverse effect on the performance of the process controlled by the PLC, it is recommended to condition the reactivation of physical outputs Q0 and Q1 as a manual operation. Before reactivation, it is then possible for the operator to take all the necessary precautions with regard to the control system and personal safety (for example, requesting a transition to manual operation).

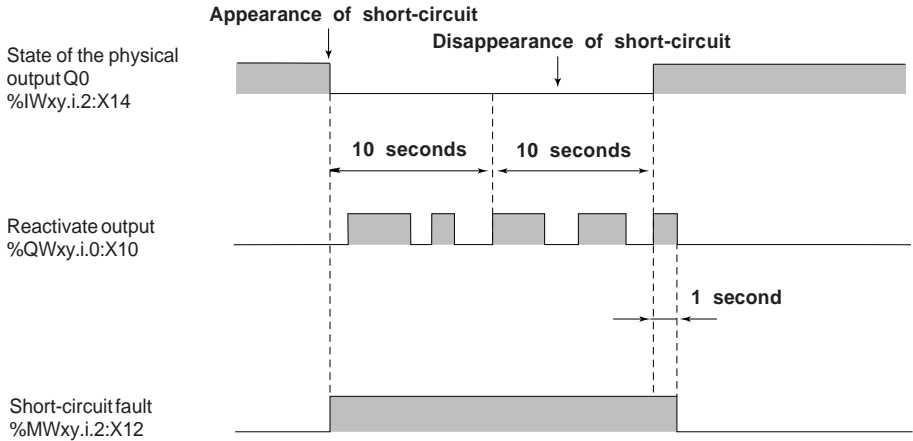
**If allowed by the process controlled by the PLC and at the user's own risk, it is possible to program an automatic reactivation.**

When one of the physical outputs Q0 and Q1 is short-circuited, both the outputs are set to 0 by the counter module. While the short-circuit persists, for safety reasons it is necessary to set physical outputs Q0 and Q1 to 0 by program (in automatic mode, set the automatic mode output enable objects of both physical outputs to 0. In manual mode, set the manual control objects of both physical outputs to 0).

**Manual reactivation of the physical outputs :**

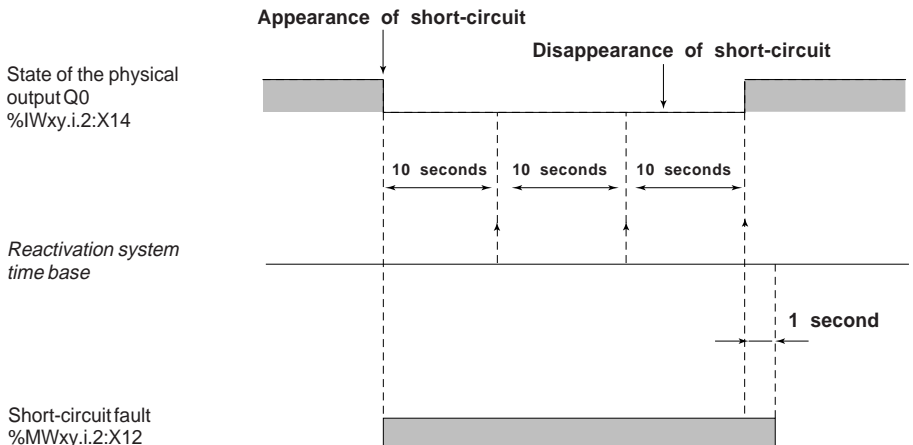
The short-circuit fault bit is set to 1 when the short-circuit appears. It is necessary to activate the output reactivation bit (%QWxy.i.0:X10) to reactivate the physical output on condition that the manual reactivation mode has been configured.

Reactivation will be effective at least 10 seconds after the short-circuit has been detected and when the short-circuit has disappeared.



**Automatic reactivation of the physical outputs :**

Reactivation is requested automatically by the CTY 2A/4A module every 10 seconds. The time base of 10 seconds is synchronous with the occurrence of the fault.



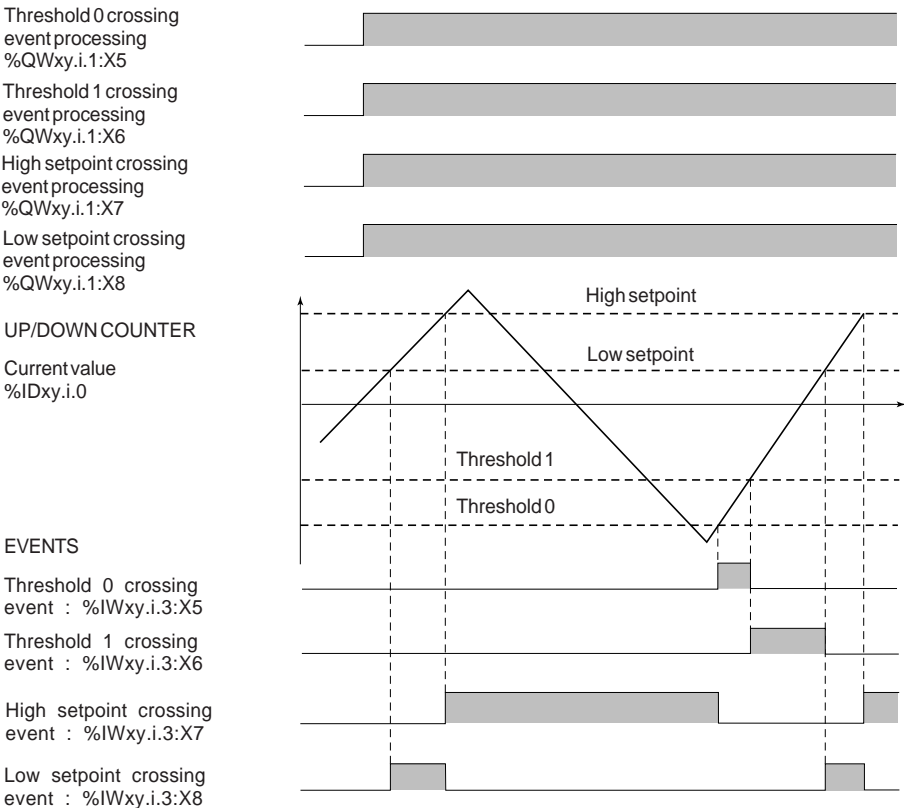
### 3.2-9 Event processing

The user can associate event processing (reflex action) with an up/down counter channel during configuration (see setup section).

If they are unmasked, several events can activate event processing :

- crossing of thresholds or high or low setpoints (see section 3.2.6),
- enable up/down counter (see section 3.2.3),
- preset (see section 3.2.4),
- capture (see section 3.2.5).

The following timing diagrams give an example of generating internal events in the up/down counter. During event processing, the user must identify the source of the event by testing the event object for state 1, and then launch the associated reflex action via the application program (see example of event processing in section 6).



### 3.3 Description of the language objects associated with the function

The language objects associated with up/down counting function are described in the following tables relating to the **enable**, the **current value**, the **preset**, the **capture**, the **comparison**, the **counter outputs** and the **physical outputs**.

ENABLE	objects	description
Enable event	%IWxy.i.3:X0	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the enable done.
Enable active	%lxy.i.0	<b>state 1</b> : the up/down counter is enabled, <b>state 0</b> : the up/down counter is inhibited.
Physical enable input <b>IEna</b>	%IWxy.i.2:X2	represents the state of the physical enable input <b>IEna</b> .
Enable	%Qxy.i.5	<b>state 1</b> : enables the physical up/down counter input <b>IEna</b> , <b>state 0</b> : inhibits the physical up/down counter input <b>IEna</b> ,
Direct enable (by program)	%Qxy.i.0	<b>state 1</b> : enables the up/down counter, enabled by physical input.
Enable event processing	%QWxy.i.1:X0	<b>state 1</b> : the enable done event is not masked, <b>state 0</b> : the enable done event is masked (the event is neither processed nor stored).

CURRENT VALUE	objects	description
Current value	<b>%IDxy.i.0</b>	current value of the up/down counter. This <b>word</b> can be read and tested. It is between : -16 777 216 and +16 777 215.
Overrun event	<b>%IWxy.i.3:X15</b>	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with an overflow of the stack of PLC events (serious error).
Invalid value	<b>%IWxy.i.2:X7</b>	<b>state 1</b> : the current up/down counter value cannot be used, <b>state 0</b> : the current up/down counter value can be used.
Measurement overrun	<b>%MWxy.i.3:X1</b>	<b>state 1</b> : the current up/down counter value is less than -16777216 or greater than +16777215, <b>state 0</b> : the current up/down counter value is between -16777216 and +16777215.
Direction (read)	<b>%lxy.i.9</b>	<b>state 1</b> : the up/down counter upcounts, <b>state 0</b> : the up/down counter downcounts.
Physical counter input <b>IA</b>	<b>%IWxy.i.2:X0</b>	represents the state of the physical up/down counter input <b>IA</b> .
Physical counter input <b>IB</b>	<b>%IWxy.i.2:X1</b>	represents the state of the physical up/down counter input <b>IB</b> .
Change of direction (write)	<b>%QWxy.i.0:X9</b>	<b>state 1</b> : the up/down counting direction is positive. The up/down counter upcounts, <b>state 0</b> : the up/down counting direction is negative. The up/down counter downcounts.



PRESET	objects	description
Preset value	%MDxy.i.4	<b>word</b> which can be written, read and tested. It is between : -16 777 216 and +16 777 215.
Preset event	%IWxy.i.3:X1	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the preset done.
Preset done	%lxy.i.1	<b>state 1</b> : when the preset has been done. The preset condition is defined in the configuration, <b>state 0</b> : on the rising or falling edge of the preset reset.
Physical preset input <b>IPres</b>	%IWxy.i.2:X3	represents the state of the physical preset input <b>IPres</b> .
Enable preset	%Qxy.i.6	<b>state 1</b> : enables the physical preset input <b>IPres</b> . <b>state 0</b> : inhibits the physical preset input <b>IPres</b> .
Direct preset (by program)	%Qxy.i.1	<b>on a rising edge</b> : sets the current value of the up/down counter to the preset value.
Reset preset	%QWxy.i.0:X1	<b>on a rising or falling edge</b> : sets the preset done bit to 0.
Preset event processing	%QWxy.i.1:X1	<b>state 1</b> : the preset done event is not masked, <b>state 0</b> : the preset done event is masked (the event is neither processed nor stored).
Physical zero marker input <b>IZ</b>	<b>no object</b>	represents the state of the physical zero marker input <b>IZ</b> :

CAPTURE	objects	description
Captured value	%IDxy.i.4	contents of the capture register. This <b>word</b> can be read and tested. It is between -16 777 216 and +16 777 215.
Capture event	%IWxy.i.3:X2	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the capture done.
Capture done	%lxy.i.2	<b>state 1</b> : when the capture has been done. The capture condition is defined in the configuration, <b>state 0</b> : on the rising or falling edge of the capture reset.
Physical read input <b>IRead</b>	%IWxy.i.2:X4	represents the state of the physical read input <b>IRead</b> .
Enable capture	%Qxy.i.7	<b>state 1</b> : enables the physical read input <b>IRead</b> , <b>state 0</b> : inhibits the physical read input <b>IRead</b> ,
Direct capture (by program)	%Qxy.i.2	<b>on a rising edge</b> : copies the current value of the up/down counter to the capture register.
Reset capture	%QWxy.i.0:X2	<b>on a rising or falling edge</b> : sets the capture done bit to 0.
Capture event processing	%QWxy.i.1:X2	<b>state 1</b> : the capture done event is not masked, <b>state 0</b> : the capture done event is masked (the event is neither processed nor stored).

COMPARISON	objects	description
Value of threshold 0	<b>%MDxy.i.6</b>	<b>word</b> which can be written, read and tested. This word is between -16 777 216 and +16 777 215.
Value of threshold 1	<b>%MDxy.i.8</b>	<b>word</b> which can be written, read and tested. This word is between -16 777 216 and +16 777 215.
Value of high setpoint	<b>%MDxy.i.10</b>	<b>word</b> which can be written, read and tested. This word is between -16 777 216 and +16 777 215.
Value of low setpoint	<b>%MDxy.i.12</b>	<b>word</b> which can be written, read and tested. This word is between -16 777 216 and +16 777 215.
Current value $\geq$ threshold 0	<b>%lxy.i.5</b>	<b>state 1</b> : the current up/down counter value is greater than or equal to the the value of threshold 0, <b>state 0</b> : the current up/down counter value is less than the value of threshold 0.
Current value $\geq$ threshold 1	<b>%lxy.i.6</b>	<b>state 1</b> : the current up/down counter value is greater than or equal to the value of threshold 1, <b>state 0</b> : the current up/down counter value is less than the value of threshold 1.
Current value $\geq$ high setpoint	<b>%lxy.i.7</b>	<b>state 1</b> : the current up/down counter value is greater than or equal to the value of the high setpoint, <b>state 0</b> : the current up/down counter value is less than the value of the high setpoint.
Current value $\geq$ low setpoint	<b>%lxy.i.8</b>	<b>state 1</b> : the current up/down counter value is greater than or equal to the value of the low setpoint, <b>state 0</b> : the current up/down counter value is less than the value of the low setpoint.

COMPARISON (cont.)	objects	description
Captured value $\geq$ threshold 0	%lxy.i.10	<b>state 1</b> : the up/down counter value captured is greater than or equal to the value of threshold 0, <b>state 0</b> : the up/down counter value captured is less than the value of threshold 0.
Captured value $\geq$ threshold 1	%lxy.i.11	<b>state 1</b> : the up/down counter value captured is greater than or equal to the value of threshold 1, <b>state 0</b> : the up/down counter value captured is less than the value of threshold 1.
Captured value $\geq$ high setpoint	%lxy.i.12	<b>state 1</b> : the up/down counter value captured is greater than or equal to the value of the high setpoint, <b>state 0</b> : the up/down counter value captured is less than the value of the high setpoint.
Captured value $\geq$ low setpoint	%lxy.i.13	<b>state 1</b> : the up/down counter value captured is greater than or equal to the value of the high setpoint, <b>state 0</b> : the up/down counter value captured value is less than the value of the low setpoint.
Threshold 0 crossing event	%lWxy.i.3:X5	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the crossing of threshold 0.
Threshold 1 crossing event	%lWxy.i.3:X6	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the crossing of threshold 1.
Highsetpoint crossing event	%lWxy.i.3:X7	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with crossing of the high setpoint.

COMPARISON (cont.)	objects	description
Low setpoint crossing event	<b>%IWxy.i.3:X8</b>	object to be tested <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with crossing of the low setpoint.
Threshold 0 crossing event processing	<b>%QWxy.i.1:X5</b>	<b>state 1</b> : the threshold 0 crossing event is not masked, <b>state 0</b> : the threshold 0 crossing event is masked (the event is neither processed nor stored).
Threshold 1 crossing event processing	<b>%QWxy.i.1:X6</b>	<b>state 1</b> : the threshold 1 crossing event is not masked, <b>state 0</b> : the threshold 1 crossing event is masked (the event is neither processed nor stored).
High setpoint crossing event processing	<b>%QWxy.i.1:X7</b>	<b>state 1</b> : the high setpoint crossing event is not masked, <b>state 0</b> : the high setpoint crossing event is masked (the event is neither processed nor stored).
Low setpoint crossing event processing	<b>%QWxy.i.1:X8</b>	<b>state 1</b> : the low setpoint crossing event is not masked, <b>state 0</b> : the low setpoint crossing event is masked (the event is neither processed nor stored).
Direction event processing	<b>%QWxy.i.1:X9</b>	<b>state 1</b> : upcounting direction during crossing of a threshold or setpoint, <b>state 0</b> : downcounting direction during crossing of a threshold or setpoint.

COUNTER OUTPUTS	objects	description
State of counter output 0	%IWxy.i.3:X10	gives the current state of counter output 0
State of counter output 1	%IWxy.i.3:X10	gives the current state of counter output 1
Set counter output 0 to 1	%Qxy.i.10	<b>state 1</b> : sets counter output 0 to 1,
Set counter output 1 to 1	%Qxy.i.11	<b>state 1</b> : sets counter output 1 to 1,
Set counter output 0 to 0	%Qxy.i.12	<b>state 1</b> : sets counter output 0 to 0,
Set counter output 1 to 0	%Qxy.i.13	<b>state 1</b> : sets counter output 1 to 0,

PHYSICAL OUTPUTS Q0 - Q1	objects	description
Enable automatic mode for output Q0	%Qxy.i.14	<b>state 1</b> : the state of the physical output Q0 follows the state of counter output 0 in automatic mode (on condition that the output is not tripped and the CTY 2A / 4A module is not faulty) <b>state 0</b> : output Q0 is at state 0.
Enable automatic mode for output Q1	%Qxy.i.15	<b>state 1</b> : the state of the physical output Q1 follows the state of counter output 1 in automatic mode (on condition that the output is not tripped and the CTY 2A / 4A module is not faulty) <b>state 0</b> : output Q1 is at state 0.
Manual/automatic mode control of output Q0	%QWxy.i.0:X12	<b>state 1</b> : output Q0 is controlled in automatic mode, <b>state 0</b> : output Q0 is controlled in manual mode.
Manual/automatic mode control of output Q1	%QWxy.i.0:X13	<b>state 1</b> : output Q1 is controlled in automatic mode, <b>state 0</b> : output Q1 is controlled in manual mode.

PHYSICAL OUTPUTS Q0 - Q1 (cont.)	objects	description
Manual mode state control of output Q0	<b>%QWxy.i.0:X14</b>	<b>state 1</b> : the state of the physical output Q0 is at 1 (on condition that the output is not tripped and the CTY 2A/4A module is not faulty) <b>state 0</b> : output Q0 is at state 0.
Manual mode state control of output Q1	<b>%QWxy.i.0:X15</b>	<b>state 1</b> : the state of the physical output Q1 is at 1 (on condition that the output is not tripped and the CTY 2A/4A module is not faulty) <b>state 0</b> : output Q1 is at state 0.
State of physical output Q0	<b>%IWxy.i.2:X14</b>	<b>manual mode</b> : the state of physical output Q0 is the same as the manual mode output state control (%QWxy.i.0:X15), <b>automatic mode</b> : the state of physical output Q0 is the same as the state of counter output 0 (%IWxy.i.3:X10), when enabled, the output is at 1; otherwise, the output is at 0. The output is forced to 0 when it is tripped or when the CTY 2A/4A module is faulty.
State of physical output Q1	<b>%IWxy.i.2:X15</b>	<b>manual mode</b> : the state of physical output Q1 is the same as the manual mode output state control (%QWxy.i.0:X14), <b>automatic mode</b> : the state of physical output Q1 is the same as the state of counter output 1 (%IWxy.i.3:X10), when enabled, the output is at 1; otherwise, the output is at 0. The output is forced to 0 when it is tripped or when the CTY 2A/4A module is faulty.

---

<b>PHYSICAL OUTPUTS Q0 - Q1 (cont.)</b>	<b>objects</b>	<b>description</b>
Short-circuit fault outputs Q0 and Q1	<b>%MWxy.i.2:X12</b>	<b>state 1</b> : short-circuit on physical output Q0 or Q1, <b>state 0</b> : no short-circuit on physical output Q0 or Q1.
Reactivate outputs Q0 and Q1	<b>%QWxy.i.0:X10</b>	<b>on a rising edge</b> : reactivation of the circuit-breaker for physical outputs Q0 and Q1.

---



## 4 Description of the up/down counting and measurement function (CTY 2C)

### 4.1 Overview

This function of the TSX CTY 2C module is used to up/down count pulses from a sensor :

- in normal mode, (on 24 bits + sign) between the values -16 777 216 and +16 777 215,
- in modulo mode, (on 25 bits) between the values 0 and +33 554 431.

The input interface offers four up/down counting configuration options :

- **IA upcounts** and **IB downcounts**,
- **IA up/down counts** and **IB** gives the **direction**,
- **IA up/down counts** and **the application** gives the **direction**,
- use of an **incremental encoder** (phase-shifted signals),

and 2 measuring options :

- use of an **SSI serial output absolute encoder**,
- use of a **parallel output absolute encoder** (with TELEFAST ABE-7CPA11 adaptor).

The module also provides a speed function in points/second :

- over a **period of measurement** and sampling which can be adjusted by the user, the speed is calculated and updated,
- speed monitoring, using an **overspeed threshold** which can be adjusted by the user, provides a safety measure for the outputs if the overspeed threshold is exceeded (set to 0).

The up/down counter is **enabled** by an auxiliary input/output configured as a physical input **IEna** (I2) or by program. Enabling on a physical input can activate event processing. The current up/down counter value is only accessible in read mode.

The auxiliary input/output which is used for enabling (physical input **IEna**), can also be configured as physical output Q2 (in this case, input IEa is no longer available).

The up/down counter is **preset** by a physical input **IPres** (I0) or by program. It can be configured (7 modes). A preset via the associated physical input may trigger a capture. The preset on the physical input can activate event processing (only for counting pulses or an incremental encoder).

The contents of the up/down counter is **captured** in the capture register by program or according to the configuration of the physical read input **IRead** (I1). The capture can be configured (3 modes). The capture on physical input can activate event processing.

The capture can be configured on input IRead (I1) to save the value of the up/down counter automatically before a physical preset.

In the **Capture before preset on IPres** configuration mode, the physical preset input IPres may trigger consecutively and automatically :

- a capture (saving the value of the up/down counter),
- followed by a preset.

---

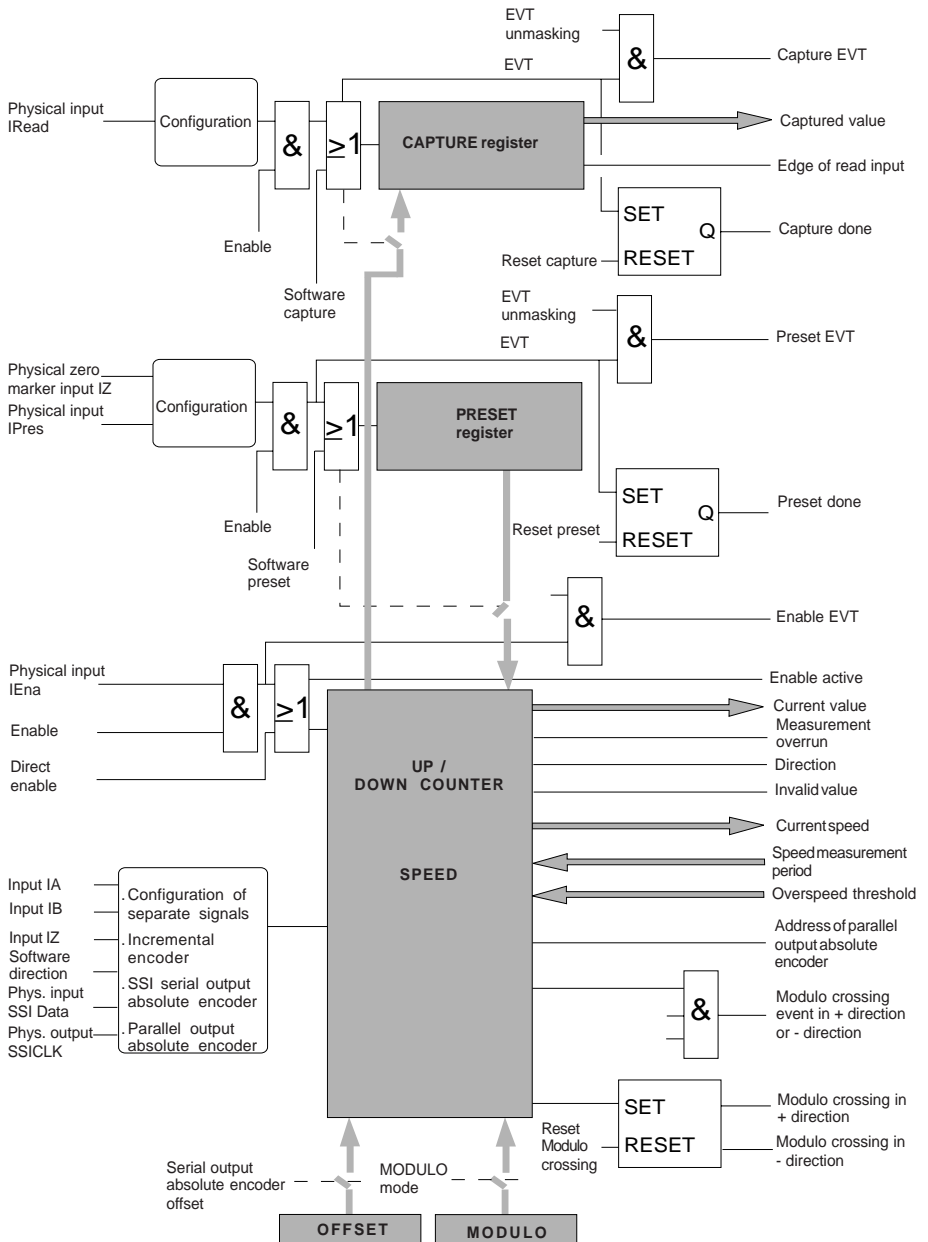
Event processing can also be activated when the current value crosses the **modulo** or the value of **threshold 0** or **1**.

The up/down counting and measurement function also offers **two reflex counter outputs** which can both be applied directly to a physical output associated with the channel.

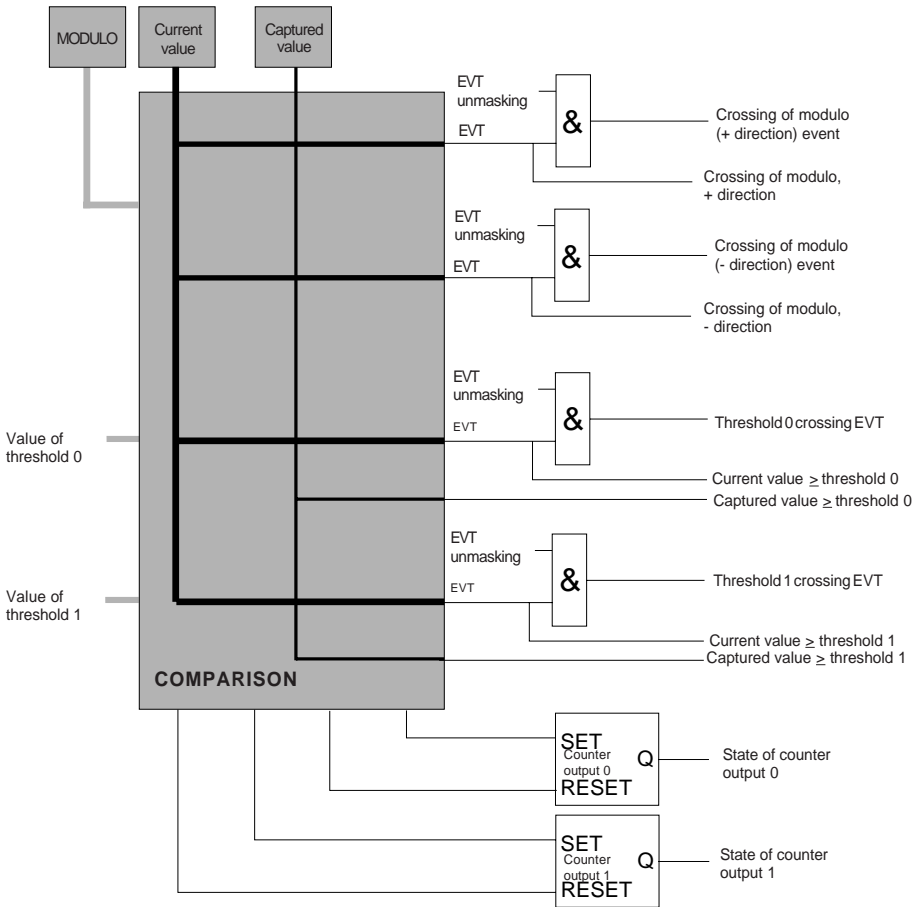
The parameters of the SET and RESET conditions (13 conditions) for counter outputs 0 and 1 can be defined in the adjustment screen.

The TSX CTY 2C module may have an additional physical output **Q2** : one auxiliary input/output per channel can be configured as input IEna or physical output Q2.

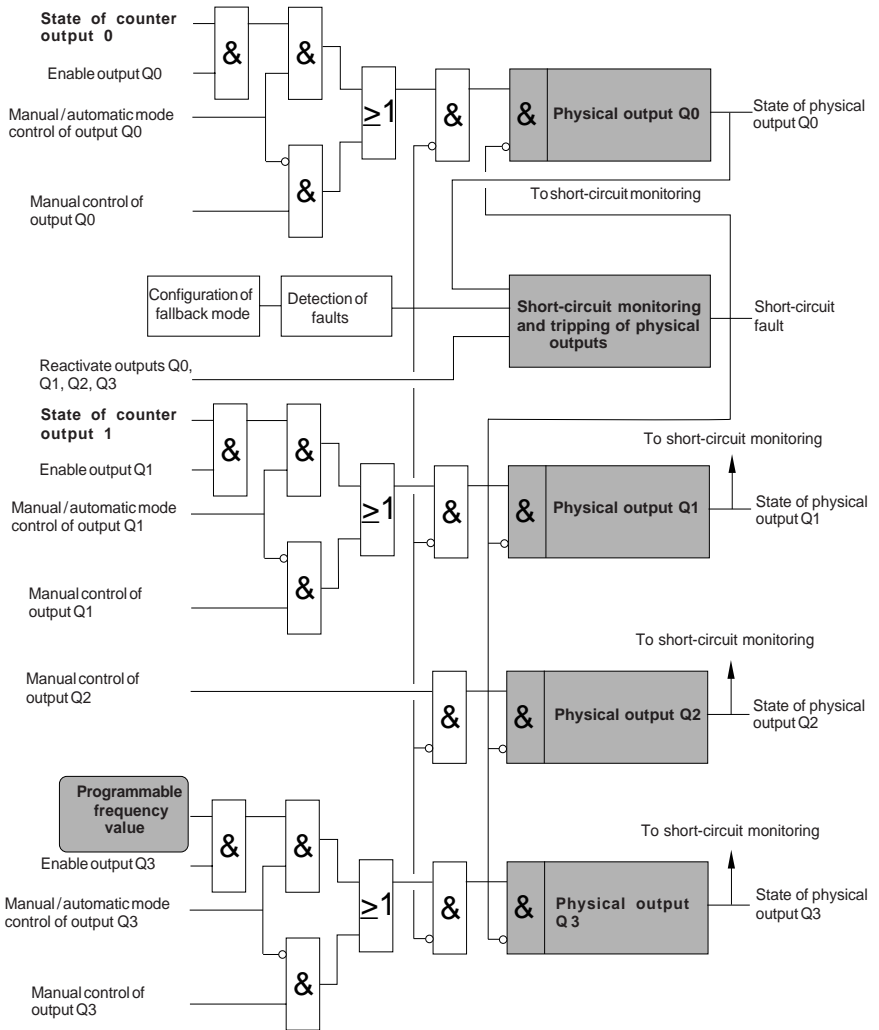
The module has an output Q3 which can be programmed as a **(programmable) frequency output**.



Overview of the up/down counter



### Overview of the comparisons



Overview of the outputs

## 4.2 Functions and timing diagrams

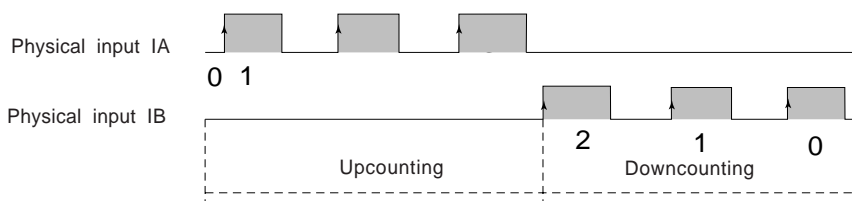
### 4.2-1 Input interface

The input interface offers four up/down counting configuration options :

#### 1 - Separate signals

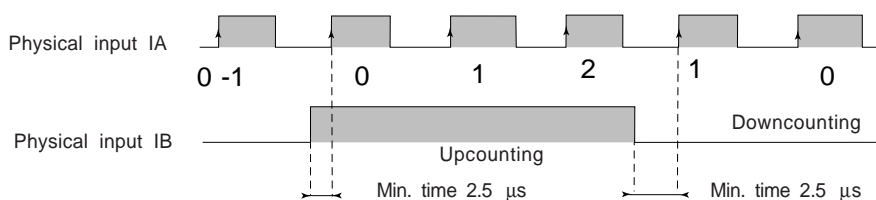
##### • IA upcounts, IB downcounts

Pulses are taken into account by the up/down counter on the rising edges of 2 physical inputs IA and IB. Physical input **IA** is used to increment the up/down counter (upcounting) and physical input **IB** is used to decrement it (downcounting). If the pulses on inputs IA and IB are simultaneous, the up/down counter does not count.



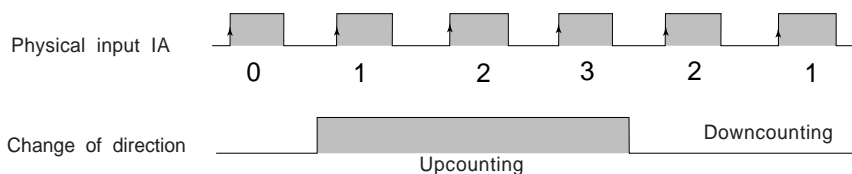
##### • IA up/down counts, IB direction

Physical input **IB** defines the up/down counting direction which is carried out on the rising edges of pulses received on physical input **IA**.



##### • IA up/down counts, application direction :

The **change direction** bit defines the up/down counting direction which is carried out on the rising edges of pulses received on physical input **IA**.



The configuration mode is also used to define :

- the **type of contact**

- mechanical contact (default selection),
- solid state contact < 250 kHz,
- solid state contact < 1 MHz (counting pulses).

- the **modulo mode**

The modulo mode option is used to up/down count in the zone [0, modulo]. The minimum modulo value is 1 and the maximum value is +33 554 432.

In modulo mode, the minimum value of thresholds 0 and 1 is 0 and the maximum value is +33 554 431.

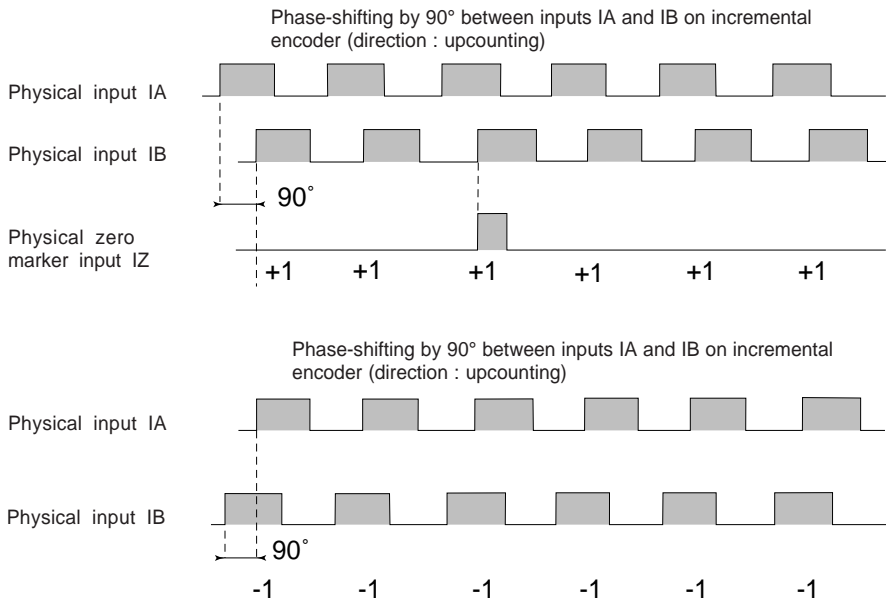
In non-modulo mode, the minimum value of thresholds 0 and 1 is -16 777 216 and the maximum value is +16 777 215.

The default configuration is without modulo mode.

## 2- Incremental encoder (phase-shifted signals)

In this operating mode physical inputs **IA** and **IB** are connected to an incremental encoder which supplies two signals (pulses) IA and IB phase-shifted by 90°. Phase-shifting between inputs IA and IB determines the direction of rotation.

The incremental encoder also provides **zero marker** information on input **IZ**, which enables the up/down counter to be preset.



The configuration mode is also used to define :

- the **type of contact**

- solid state contact < 250 kHz (multiplication by 1) or 125 kHz (multiplication by 4),
- solid state contact < 500 kHz (multiplication by 1) or 250 kHz (multiplication by 4).

- a **measurement inversion**

This option is used to invert the measurement of an incremental encoder; ie. the direction in which the measurement evolves for an encoder direction of rotation. Measurement inversion is not configured by default.

- a **line check**

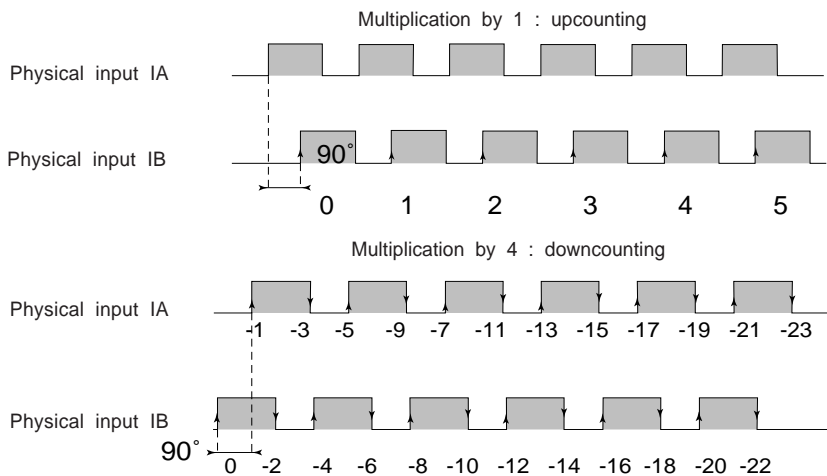
The **line check** may be applied to an incremental encoder connected to the PLC via an RS 422 / 485 link. If this option is configured, the PLC signals a fault if a line break or a short-circuit is detected on one of the physical inputs IA, IB or IZ. Application processing corresponding to the fault detected can then be carried out.

The line check is not configured by default.

- **multiplication by 4**

**Multiplication by 4** is used to improve the precision of the incremental encoder.

When the incremental encoder is configured for **multiplication by 1**, up/down counting is performed on the rising edges of physical input IB. When it is configured for **multiplication by 4**, up/down counting is performed on all rising and falling edges of physical inputs IA and IB.



In multiplication by 1 (default selection), the maximum frequency of the phase-shifted signals is 500 kHz. In multiplication by 4, this is 250 kHz.



### 3 - SSI serial output absolute encoder

In this operating mode the **SSI Data** and **SSICLK** physical inputs are connected to a serial output absolute encoder with the following frame characteristics :

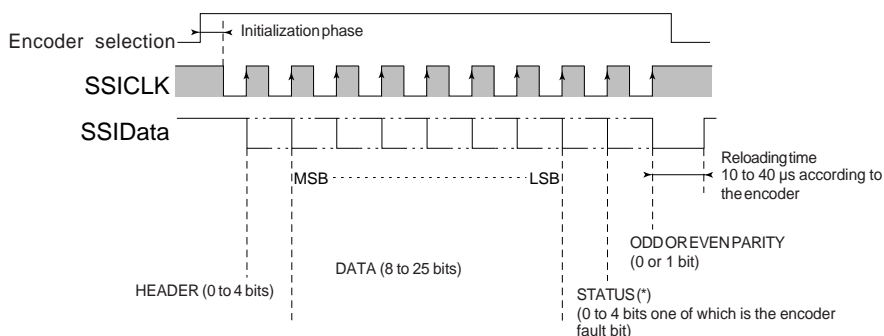
- code : binary (default) or Gray,
- SSICLK transmission frequency : 150 kHz, 200 kHz (default), 375 kHz, 500 kHz, 750 kHz or 1 MHz,
- number of header bits : 0 to 4 bits (0 by default). These bits cannot be read,
- number of encoder data bits : 8 to 25 bits (16 bits by default),
- individual masking of data bits :
  - number of masked most significant bits : 17 bits (0 by default),
  - number of masked least significant bits (reduction of resolution) : 17 bits (0 by default),
 with the following limit : at least 8 encoder data bits.

$$\begin{aligned} &\text{no. of encoder data bits} - \Sigma \text{ no. of masked most significant bits} \\ &- \Sigma \text{ no. of masked least significant bits} \geq 8 \end{aligned}$$

- number of status bits : 0 to 4 (0 by default). These bits can be read.  
The configuration of at least one bit means that it is possible to have an error bit specific to the absolute encoder and to set it :
  - rank : 1 to 4 (1 by default). The rank is counted from the LSB to the MSB.
  - level of activity : 0 or 1 (1 by default).

Level of activity	
0	0 = fault specific to the encoder 1 = no fault specific to the encoder
1	0 = no fault specific to the encoder 1 = fault specific to the encoder

- parity (even) : no parity bit (default), even parity or odd parity (not checked by the module). If parity is odd, the number of status bits is limited to 3.
- line check : with line monitoring by default,
- measurement inversion : no measurement inversion by default,
- absolute encoder offset : this adjustment parameter only relates to absolute encoders. It can be used to shift the zero obtained, by adding the value of the offset to the current value supplied by the absolute encoder.
- modulo value : for signals emitted by a serial output absolute encoder, up/down counting is performed implicitly in modulo mode. The modulo value is given directly by the number of non-masked bits. The up/down counter evolves in the zone [0; modulo[. The minimum value of the modulo is 1 and its maximum value is +33 554 432 (25 data bits with 0 masked most significant data bits).



(\*) Limitation : if parity is odd, there should be a maximum of 3 status bits.

### Configurable frame supplied by an SSI absolute encoder

#### 4 - Parallel output absolute encoder

In this operating mode the **SSI Data** and **SSICLK** physical inputs are connected to a parallel output absolute encoder, via the TELEFAST adaptor ABE-7CPA11. The frame characteristics are as follows :

- code : binary (default) or Gray,
- SSICLK transmission frequency : 150 kHz, 200 kHz (default), 375 kHz, 500 kHz, 750 kHz or 1 MHz,
- number of header bits : fixed at 0. The ABE-7CPA11 / TSX CTY 2C serial frame does not contain a header bit,
- number of encoder data bits : 8 to 24 bits (24 bits by default),
- individual masking of data bits :
  - number of masked most significant bits : 16 bits (0 by default),
  - number of masked least significant bits (reduction of resolution) : 16 bits (0 by default),
 with the following limit :

$\text{no. of encoder data bits} - \sum \text{no. of masked most significant bits} - \sum \text{no. of masked least significant bits} \geq 8$
---

- number of status bits : fixed at 3 bits. The ABE-7CPA11/TSX CTY 2C serial frame contains 3 status bits by default. These bits can be read. It is therefore possible to have an error bit specific to the absolute encoder :
  - rank : fixed at 3. This concerns the first status bit (rank 3 is the MSB of the status bits).
  - level of activity (1 bit) : 0 or 1 (1 by default).

Level of activity	
0	0 = fault specific to the encoder 1 = no fault specific to the encoder
1	0 = no fault specific to the encoder 1 = fault specific to the encoder

- parity : the parity bit is fixed at even parity,
- line check : with line check,
- measurement inversion : no measurement inversion by default,
- absolute encoder offset : this adjustment parameter only relates to absolute encoders. It can be used to shift the zero obtained, by adding the value of the offset to the current value supplied by the absolute encoder,
- modulo value : for signals emitted by a serial output absolute encoder, up/down counting is performed implicitly in modulo mode. The modulo value is given directly by the number of non-masked bits. The up/down counter evolves in the zone [0; modulo]. The minimum value of the modulo is 1 and its maximum value is +16 777 216 (24 data bits with 0 masked most significant data bits),
- multiplexing of parallel output absolute encoders : no multiplexing by default.

## 4.2-2 Invalid value

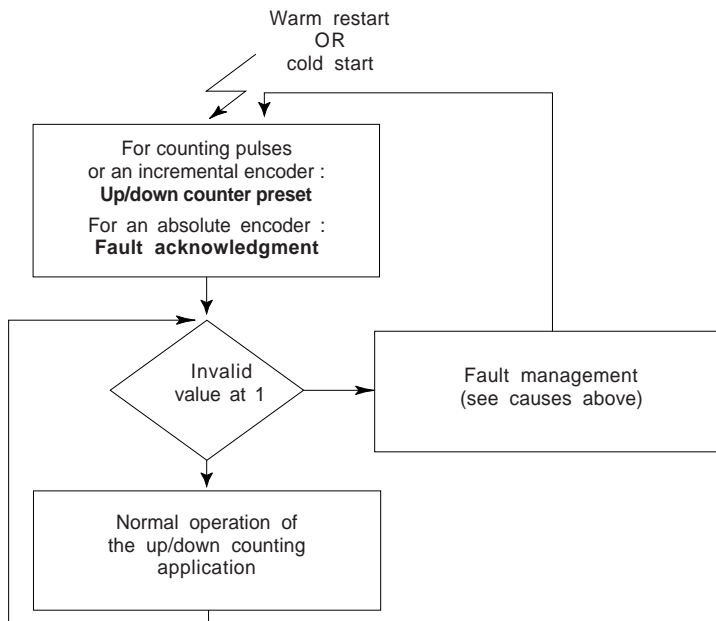
In addition to the diagnostic functions, the user has access to invalid value information. This is used to detect loss of pulses when up/down counting which may have been caused by :

- a cold start or warm restart of the application,
- a fault on the up/down counter input : %MWxy.i.2:X0
  - proximity sensor power supply fault : %MWxy.i.2:X13,
  - encoder line break or short-circuit fault : %MWxy.i.2:X14,
  - serial frame transmission fault (parity error or error on the serial line), when using an SSI serial output absolute encoder or a parallel output absolute encoder : %MWxy.i.2:X15,
  - fault specific to the absolute encoder (if this fault bit is configured in the serial frame status bits) : %MWxy.i.3:X2,
- an overshoot of the measurement in normal mode, for counting pulses or an incremental encoder (the capacity of the up/down counter is 24 bits + sign) : %MWxy.i.3:X1,

In this case, word %IWxy.i.2:X7 is at **state 1**, the contents of the up/down counter cannot be used and the counter outputs are set to 0.

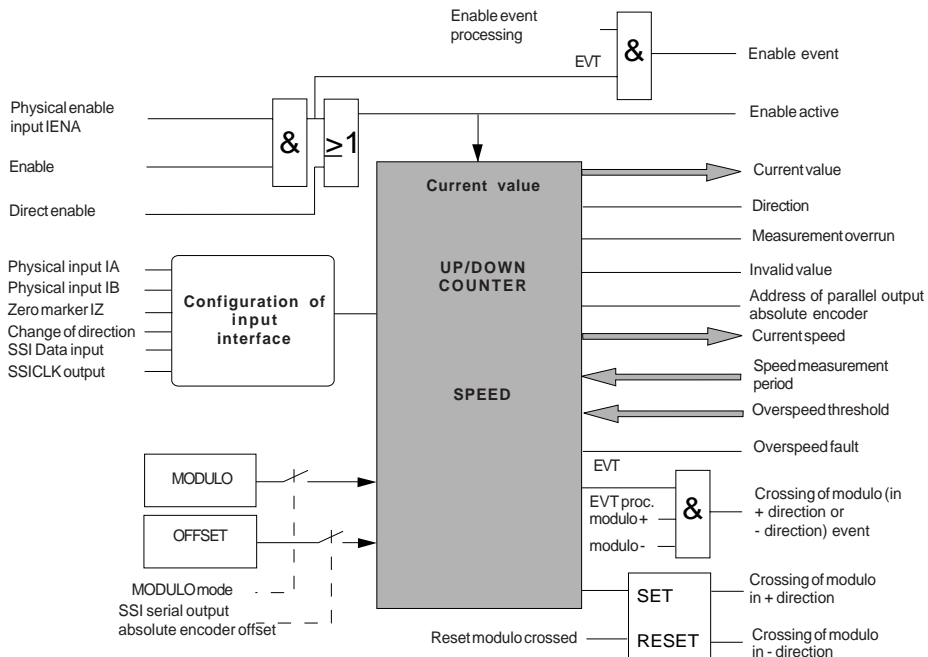
%IWxy.i.2:X7 is at **state 0** when the up/down counter is preset, provided that none of the causes of the loss of pulses are present.

The methodology for managing **invalid values** via the application is as follows :

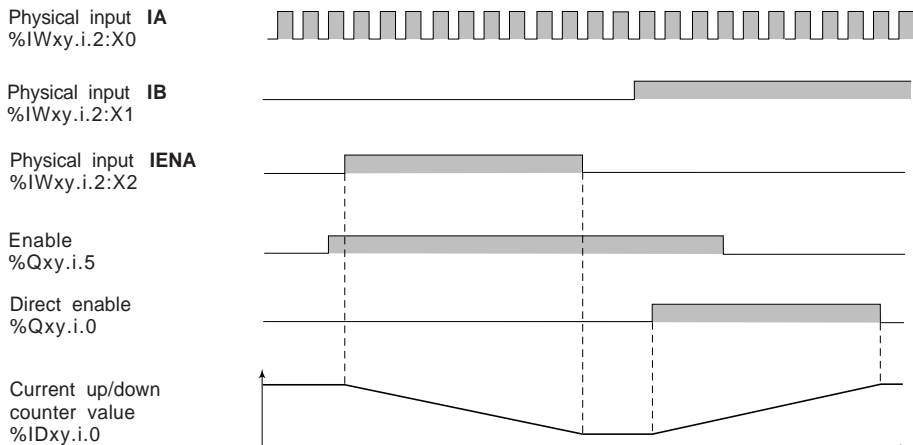


### 4.2-3 Enable

When the up/down counter is enabled, it can count in both directions depending on the physical up/down counter inputs.



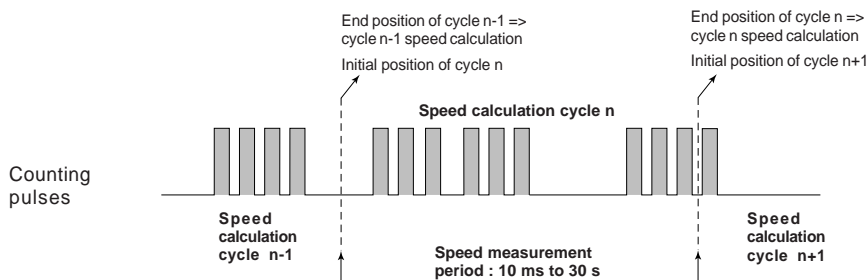
The timing diagram below shows the enable function for the up/down counter :



#### 4.2-4 Speed monitoring

The TSXCTY 2C module offers the speed monitoring function. The principle for calculating the speed is the same as that of the **frequency meter** : over a **measurement and sampling period** which can be adjusted by the user, the speed is calculated and updated, in number of points per second.

The default value for the measurement period is 1 second.



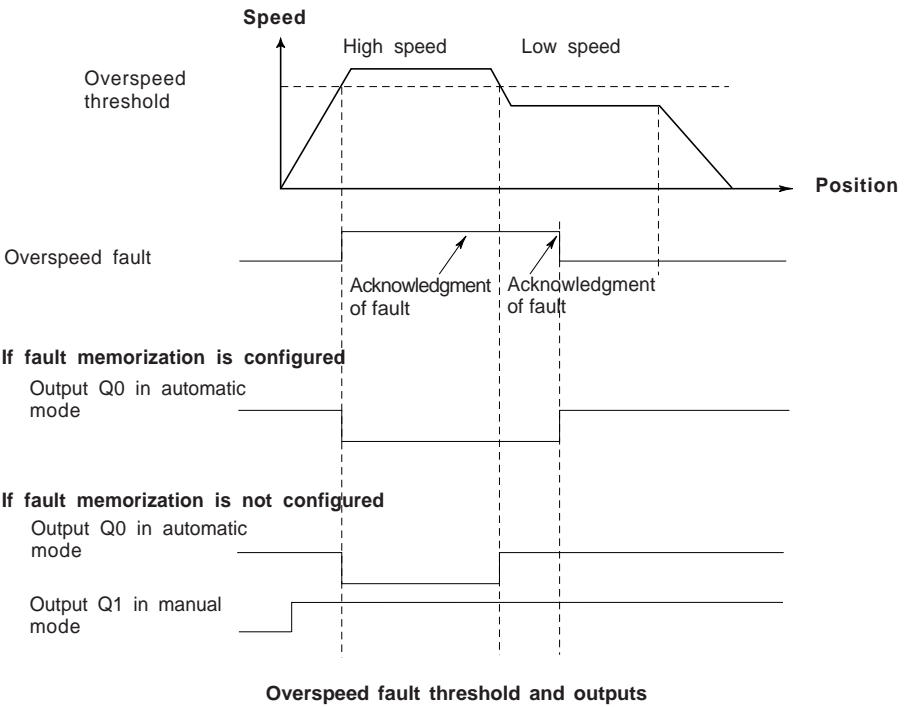
Principle for calculating the speed

The following table gives the minimum period for the measurement, as a function of the speed to be measured, in order to ensure, for example, a precision of 0.1%.

$$\text{Period (in seconds)} \geq 1 / \text{speed (in points/second)} \times \text{precision (in \%)}$$

Speed to be measured (points/second)	Minimum period of the speed measurement (ms)	Precision (%)
250 000...1 000 000	≥ 10	0.1
40 000...250 000	≥ 25	0.1
10 000...40 000	≥ 100	0.1
1 000...10 000	≥ 1000 (1 s)	0.1
100...1 000	≥ 10 000 (10 s)	0.1
10...100	≥ 100 000 (100 s)	0.1
1...10	≥ 1 000 000 (1 000 s)	0.1

Speed monitoring, via an **overspeed threshold** which can be adjusted by the user, is used (in automatic mode only) to provide a safety measure on the outputs if the overspeed threshold is exceeded (the outputs are set to 0).



When the output is in automatic mode :

- if fault memorization is configured, acknowledgment of the overspeed fault once it has disappeared resets the output to the state it was in prior to the fault,
- if fault memorization is not configured, disappearance of the overspeed fault resets the output to the state it was in prior to the fault.

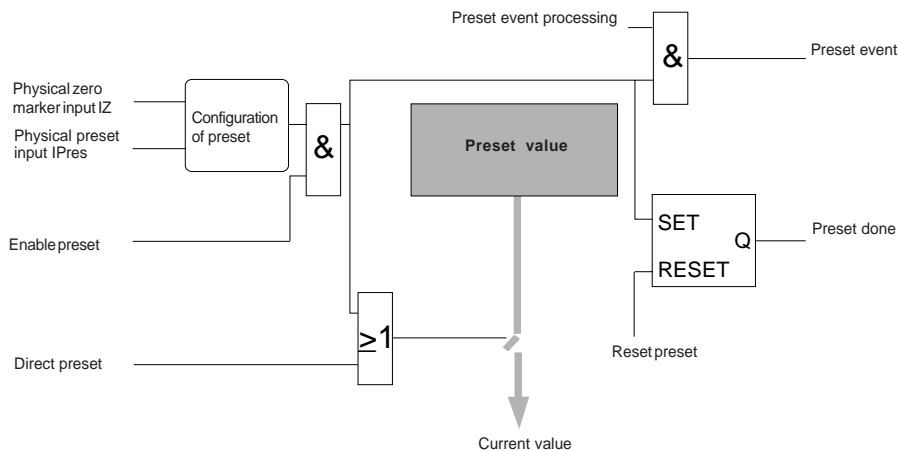
When the output is in manual mode, it is not affected by the overspeed fault and maintains its state.

#### 4.2-5 Preset

The preset is used to initialize the up/down counter to the preset value (only for counting pulses and an incremental encoder). The up/down counter can be preset in 7 modes which are combinations relating to the states, and/or the edges of physical inputs **IPres** and **IZ** :

- rising edge of **IPres**,
- falling edge of **IPres**,
- rising edge of **IPres** + direction / falling edge of **IPres** - direction,
- rising edge of **IPres** - direction / falling edge of **IPres** + direction,
- state of **IPres**,
- short cam reference point (see below),
- long cam reference point (see below).

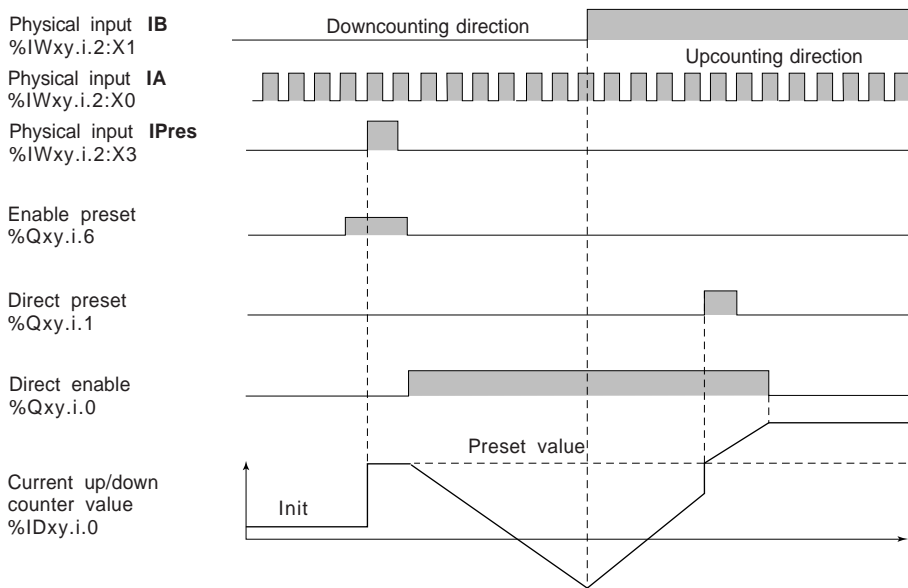
The preset affects the invalid value object (see section 4.2.2).



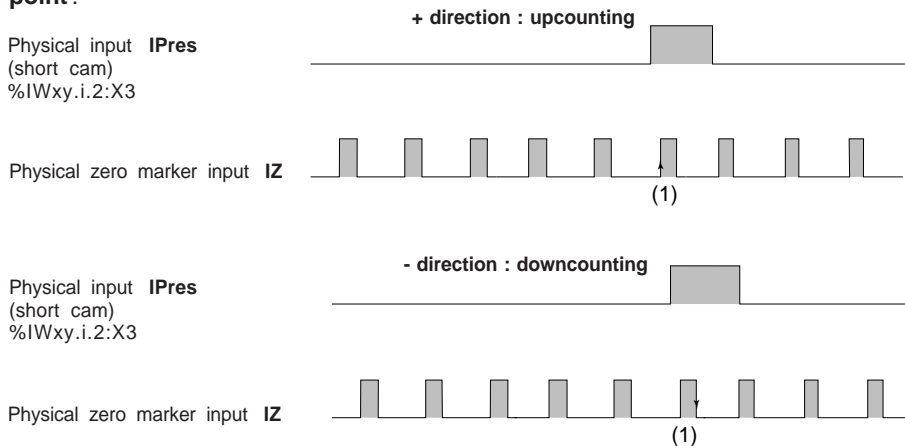
(1) The preset value object %MDxy.i.4 is managed in accordance with the explicit exchange mechanism.



The following timing diagram shows the preset mode on the **rising edge of IPres**:



The following timing diagram shows the preset mode on the **short cam reference point**:



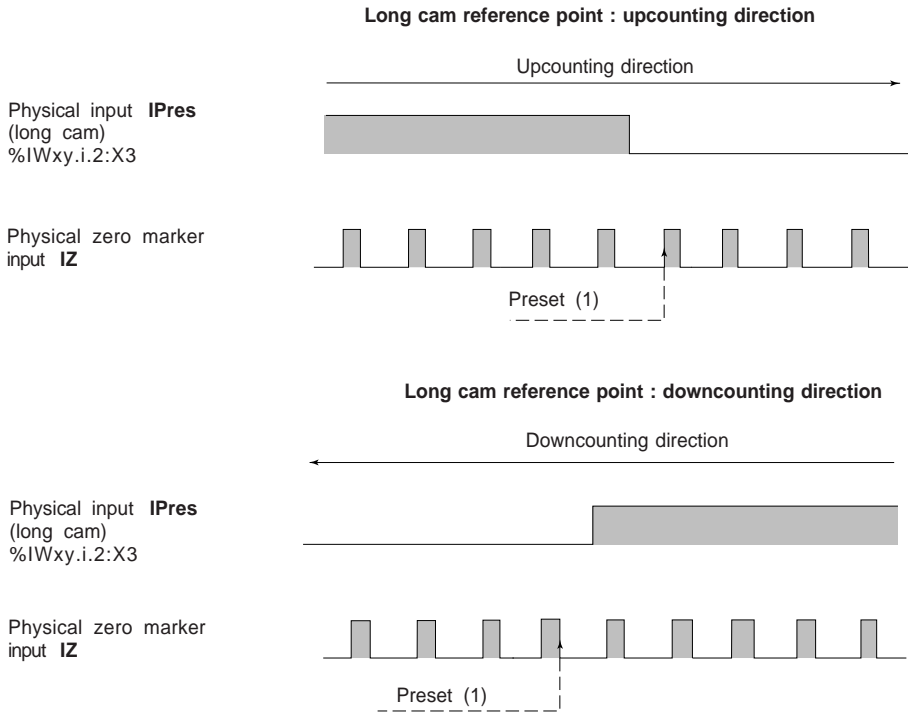
(1) The preset is taken into account :

- in the + direction (upcounting) : input **IPres** at state 1, rising edge of zero marker input **IZ** and software enable,
- in the - direction (downcounting) : input **IPres** at state 1, falling edge of zero marker input **IZ** and software enable,

## Note

In principle, if the short cam is less than one revolution of the incremental encoder, the zero marker will only appear once in the cam.  
 If however there are several revolutions of the incremental encoder in the cam, the last active edge of the zero marker signal triggers a preset.

The following timing diagram shows the preset mode on the **long cam reference point** :



- (1) The preset is taken into account on the first rising edge of the zero marker input **IZ** which follows the changeover to state 0 of input **IPres**, for both the upcounting and the downcounting direction, and software enable.

#### 4.2-6 Capture (read input)

The capture function is used to copy the current up/down counter value to a capture register according to the configuration of the physical read input **IRead** :

- on a **rising edge** of input **IRead**,
- on a **falling edge** of input **IRead**,
- on the **rising and falling edges** of input **IRead**.

Direct capture (by program) is used to save the current value of the up/down counter in a capture register.

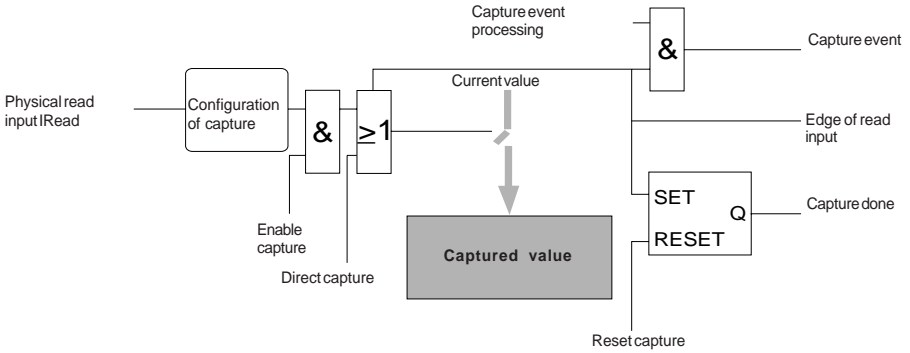
The Capture before preset on input IPres configuration mode is used, via the physical preset input IPres, to trigger consecutively and automatically :

- a capture (saving the value of the up/down counter),
- followed by a preset.

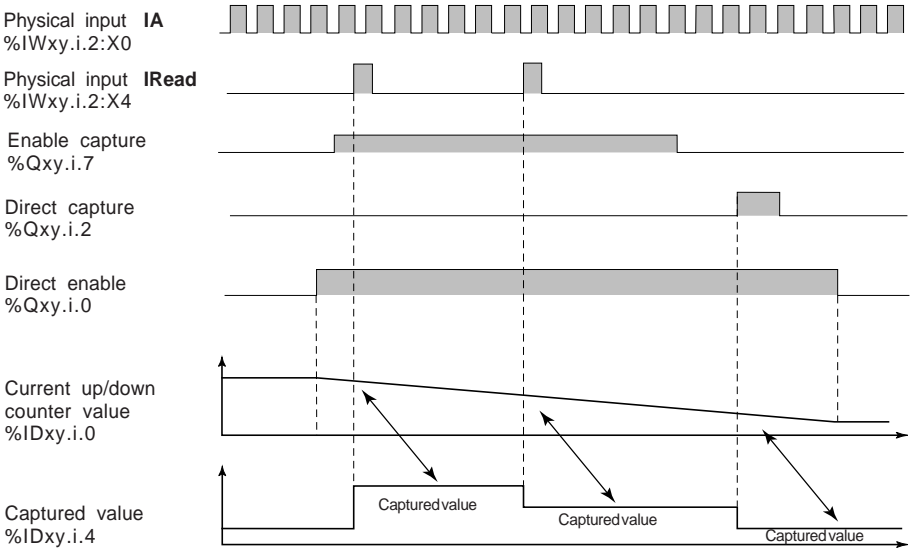
When the read input is configured on a **rising edge** or a **falling edge**, the time performance levels are maximum.

The table below shows the permissible gap between two consecutive edges of the read input, used for example to measure the length of parts (configuration mode : capture on rising and falling edges of the physical read input), as a function of the frequency of the counter inputs. As long as these conditions are observed, the function for measuring the length of parts should operate correctly, ie. all edges present on the physical read input taken into account : Separation  $\geq 0.5$  (ms) x Frequency (kHz)

Frequency of the counter inputs	Minimum separation between two consecutive edges of the read input IRead
125 kHz	63
250 kHz	125
500 kHz	250
1 MHz	500



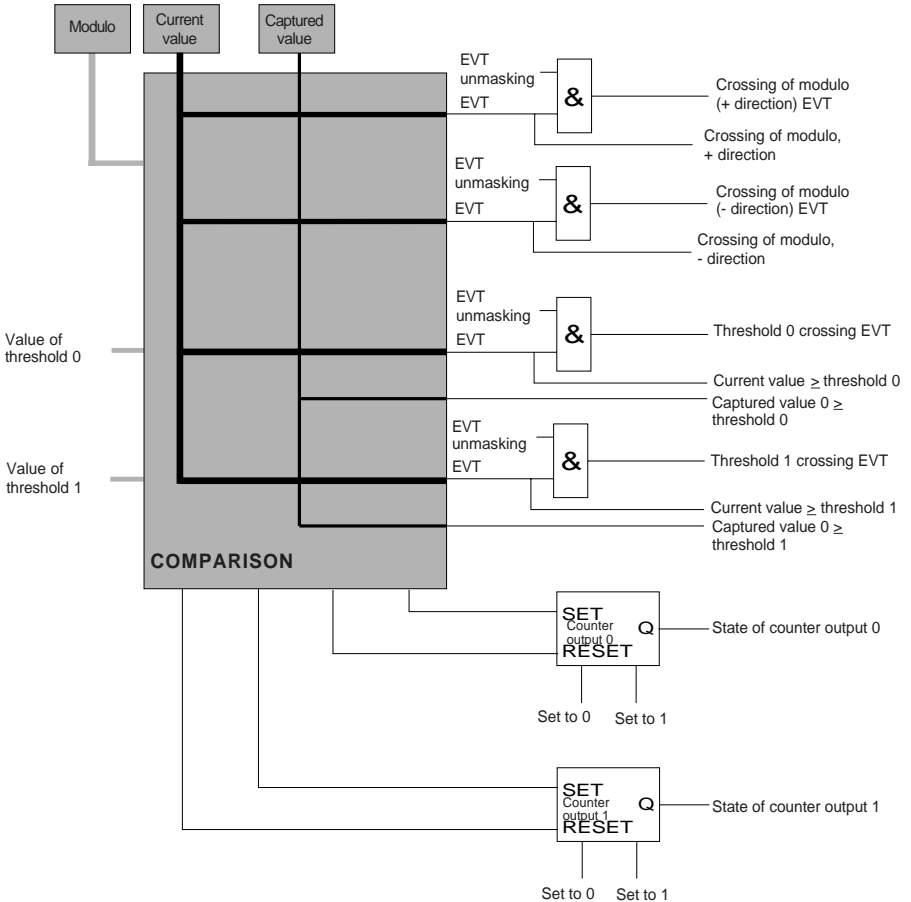
The following timing diagram shows the configuration mode for the capture on the **rising edge of IRead** :



### 4.2-7 Comparison

The comparisons of the captured value and the current value with the thresholds and setpoints are given as language objects.

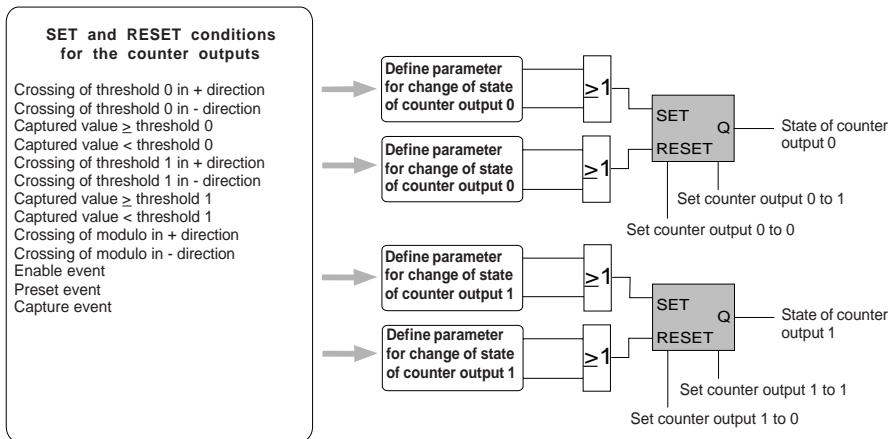
The crossing of the thresholds and the modulo in a positive direction and in a negative direction may generate events.



## 4.2-8 Counter outputs

For each counter output, the parameters of the SET and RESET conditions can be defined. The SET and RESET input logic allows 13 combinations of states relating to :

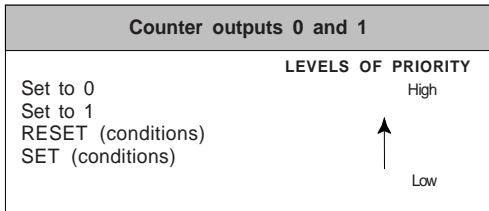
- the crossing of thresholds or modulo by the current up/down counter value,
- the events generated by the up/down counter enable, preset and capture auxiliary inputs.
- the positions of the captured value in relation to the thresholds.



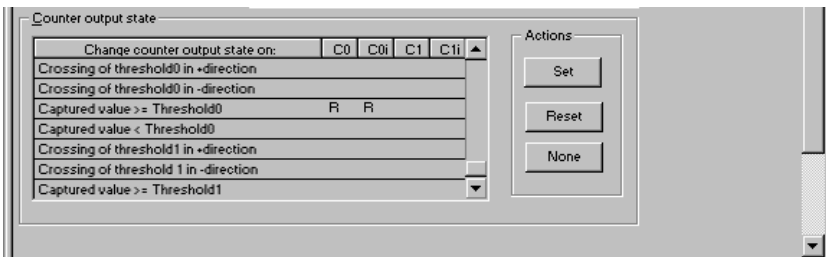
### Caution

**%IWxy.i.2:X7** (invalid value) at **state 1** shows that the contents of the up/down counter cannot be used and counter outputs 0 and 1 are set to 0.

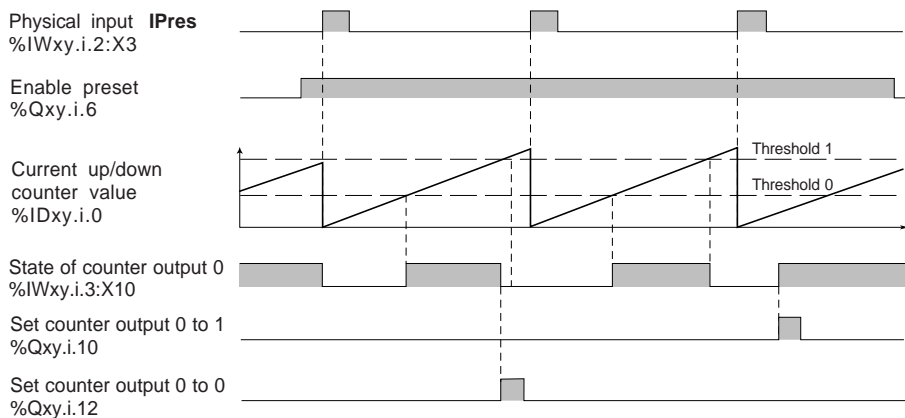
The guidelines for the levels of priority concerning counter outputs 0 and 1 are as follows :



The adjustment screen below shows an example of the parameters for counter outputs 0 and 1 for a channel :



The timing diagram below shows the SET and RESET conditions defined in the previous adjustment screen for counter output 0 :



#### 4.2-9 Outputs

The TSX CTY 2C module has 4 x 24 V solid state outputs : Q0, Q1, Q2 and Q3; Q2 is actually a configurable input/output :

- an auxiliary input/output can be configured either as a physical enable input IEna, or as a physical output Q2,

Each output **Q0**, **Q1**, **Q2** or **Q3** can be used in **manual mode**. The state of the physical outputs Q0 to Q3 is the same as output controls in manual mode.

For each counter channel, two of the four outputs, used in manual mode, can be used to define the addresses of the absolute encoders :

- one to four multiplexed parallel output absolute encoders, via the TELEFAST adaptor, ABE-7CPA11,

Output **Q0** or **Q1** can be used in **automatic mode**. The state of physical outputs Q0 and Q1 is then the same as counter outputs 0 and 1. Automatic mode enables the use of reflex actions at TSX CTY 2C module level : the state of counter outputs 0 and 1 is applied respectively to physical outputs Q0 and Q1, according to the evolution of the up/down counter.

Output **Q3** can be used in programmable **frequency mode** from 1 ms to 4000 s, in steps of 1 ms. The programmable frequency output provides an external synchronization pulse on several channels on several counter modules.



OUTPUT FUNCTIONS			
OUTPUTS	COUNTER OUTPUTS	STANDARD OUTPUTS / ENCODER ADDRESSING	PROGRAMMABLE FREQUENCY OUTPUT
Q0	Automatic mode	Manual mode	
Q1	Automatic mode	Manual mode	
Q2		Manual mode (*)	
Q3		Manual mode	Automatic mode

(\*) Output Q2 is only available in manual mode.

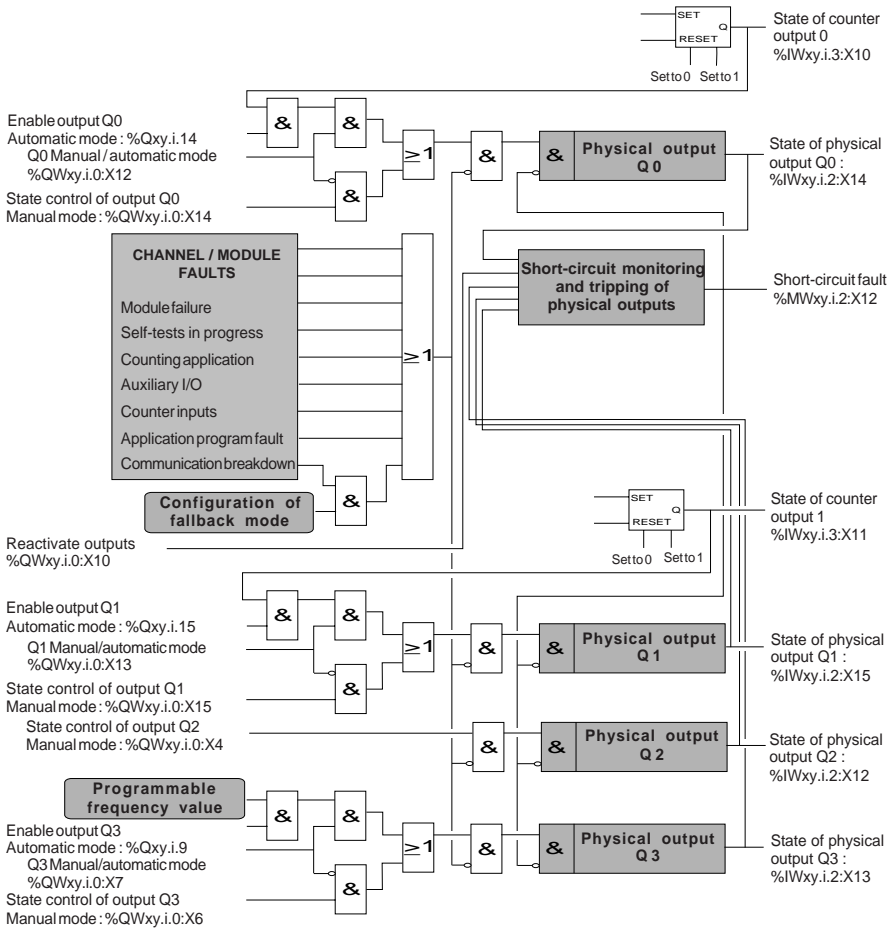
**Outputs Q2 and Q3 in manual mode are specified for use as discrete outputs, thus enabling a parallel output absolute encoder to be addressed, when several parallel output absolute encoders are multiplexed using TELEFAST ABE-7CPA11. Outputs Q0 and Q1 therefore remain available for activation according to the state of the counter outputs.**

Manual control of output Q2 %Qxy.i.20 (address LSB)	Manual control of output Q3 %Qxy.i.21 (address MSB)	Encoder address %IWxy.i.10 %IWxy.i.10:X1 (most significant) or %IWxy.i.2:X9 (status bit, rank 2) %IWxy.i.10:X0 (least significant) or %IWxy.i.2:X8 (status bit, rank 1)
0	0	Address of encoder 0
1	0	Address of encoder 1
0	1	Address of encoder 2
1	1	Address of encoder 3

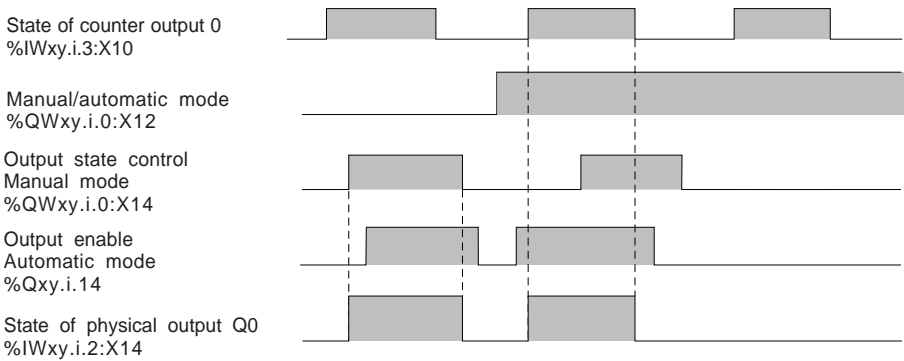
The table below also shows the operation of outputs Qi :

Manual/automatic mode of output Qi	Enable output Qi automatic mode	Manual control of output Qi	State of output Qi
Manual 0	X	0	0
	X	1	1
Auto 1 (*)	0	X	0
	1	X	. State of counter output for Q0 . State of counter output for Q1 . Programmable frequency output for Q3

(\*) Output Q2 is only available in manual mode.



The timing diagram below shows the operation of output Q0 as an example :



The module ensures safety of operation; ie. it resets the outputs to zero, according to the following conditions :

- the type of faults,
- manual or automatic mode control of the outputs.

When the outputs are in **manual mode**, the module resets outputs Q0 to Q3 to 0 during one of the following faults :

- module failure,
- self-test in progress,
- auxiliary I/O fault (auxiliary I/O power supply fault, short-circuit fault on at least one of the 4 outputs),
- application program fault (invalid software configuration or channel reconfiguration).

Outputs in **manual mode** are not reset to 0 for one of the following faults :

- counting application program fault (encoder or proximity sensor power supply fault, encoder line break or short-circuit fault, SSI serial frame fault, fault specific to the absolute encoder),
- measurement overrun fault or overspeed fault,
- adjustment application program fault : faulty adjustment bit (%MWxy.i.2:X2).

When the outputs are in **automatic mode**, they are reset to 0 irrespective of the fault. The module therefore ensures safety of operation, even if the fault is **masked**.

Independent of the manual or automatic mode control of outputs, during a **communication breakdown** :

- if the fallback mode is configured as **reset**, outputs Q0 to Q3 are set to 0,
- if the fallback mode is configured as **maintain**, outputs Q0 to Q3 remain in the state they were in prior to the fault.

---

**Table summarizing the actions for the various faults on the outputs**

Faults	Reset the output	
	Manual mode	Automatic mode
Module failure %MWxy.MOD.2:X0	Reset	Reset
Self-test in progress %MWxy.MOD.2:X3	Reset	Reset
Counter inputs fault %MWxy.i.2:X0	No reset	Reset
Counting application program fault (adjustment error) %MWxy.i.2:X1	No reset	Reset
Counting application program fault (invalid software configuration) %MWxy.i.3:X0	Reset	Reset
Auxiliary I/O fault %MWxy.i.2:X3	Reset	Reset
Auxiliary I/O power supply fault %MWxy.i.2:X11	Reset	Reset
Short-circuit fault on at least one of the 4 outputs %MWxy.i.2:X12	Reset	Reset
Encoder or proximity sensor power supply fault %MWxy.i.2:X13	No reset	Reset
Encoder line break or short-circuit fault %MWxy.i.2:X14	No reset	Reset
SSI serial frame fault %MWxy.i.2:X15	No reset	Reset
Measurement overrun fault %MWxy.i.3:X1	No reset	Reset
Absolute encoder specific fault %MWxy.i.3:X2	No reset	Reset
Overspeed fault %MWxy.i.3:X3	No reset	Reset

**Protection against overloads and short-circuits**

Physical outputs Q0, Q1, Q2 and Q3 (solid state outputs) have an internal electronic protection device which is used to detect an overload (typically a current higher than 625 mA) or a short-circuit at 0V (when the output is at state 1). The occurrence of such a fault causes the following :

- setting physical outputs Q0, Q1, Q2 and Q3 to 0 (%IWxy.i.2:X14, %IWxy.i.2:X15,...),
- setting the short-circuit fault bit to 1 (%MWxy.i.2:X12). This object is updated after a READ\_STATUS explicit exchange or in debug mode on the counter channel).
- flashing of the **CH** indicator lamp associated with the counter channel,
- activation of the **I/O** indicator lamp of the counter module, as a steady red light.

The indication of the short-circuit fault disappears 1 second after the effective reactivation of the physical outputs (the reactivation mode is defined during **configuration**).

**Reactivation of the physical outputs**

When a fault has caused physical output Q0, Q1, Q2 or Q3 to trip, it must be reactivated.

As tripping has an adverse effect on the performance of the process controlled by the PLC, it is recommended to condition the reactivation of physical outputs Q0 to Q3 as a manual operation. Before reactivation, it is then possible for the operator to take all the necessary precautions with regard to the control system and personal safety (for example, requesting a transition to manual operation).

**If allowed by the process controlled by the PLC and at the user's own risk, it is possible to program an automatic reactivation.**

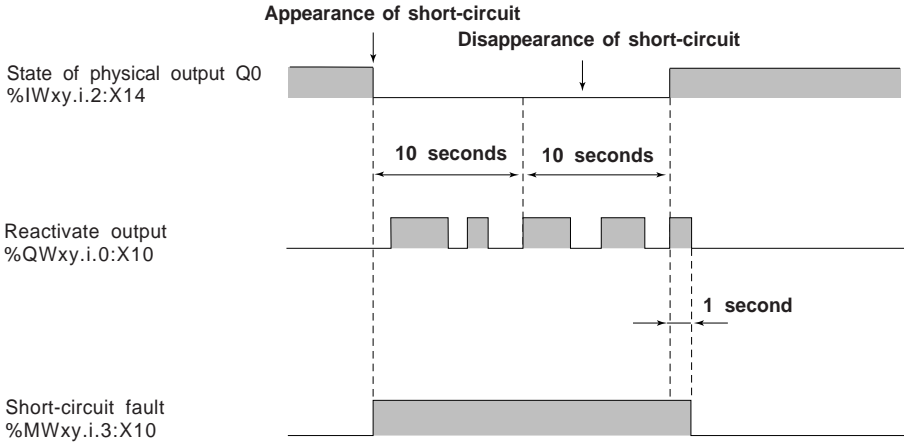
When one of the 4 physical outputs Q0 to Q3 is short-circuited, all 4 outputs are set to 0 by the counter module. While the short-circuit persists, for safety reasons it is necessary to set the 4 physical outputs Q0 to Q3 to 0 by program :

- regardless of the mode (manual or automatic), disable the outputs : set the enable bits for the 4 physical outputs to 0,
- in manual mode : set the manual control objects for the 4 physical outputs to 0.

### Manual reactivation of the physical outputs

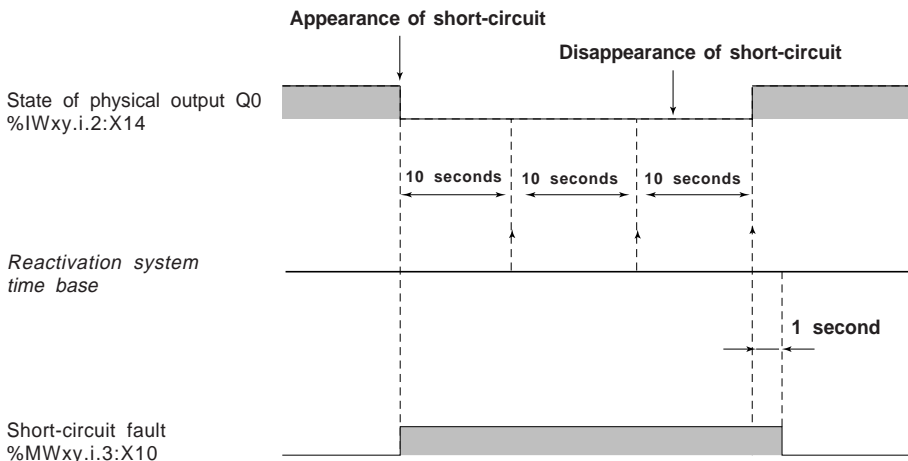
The short-circuit fault bit is set to 1 when the short-circuit appears. It is necessary to activate the output reactivation bit (%QWxy.i.0:X10) to reactivate the physical output on condition that the manual reactivation mode has been configured.

Reactivation will be effective at least 10 seconds after the short-circuit has been detected as long as the short-circuit has disappeared.



### Automatic reactivation of the physical outputs

Reactivation is requested automatically by the TSX CTY 2C module every 10 seconds. The time base of 10 seconds is synchronous with the occurrence of the fault.



#### 4.2-10 Event processing

The user can associate event processing (reflex action) with an upcounter channel during configuration.

If they are unmasked, several events can activate event processing :

- crossing of thresholds, the modulo in + direction or the modulo in - direction (see section 4.2-7),
- counter enable (see section 4.2-3),
- preset (upcounting only, see section 4.2.5),
- capture (see section 4.2.6).

The following timing diagrams give an example of generating internal events in the up/down counter. During event processing, the user must identify the source of the event by testing the event object for 1, and then launch the associated reflex action via the application program.

Threshold 0 crossing  
event processing  
%QWxy.i.1:X5

Threshold 1 crossing  
event processing  
%QWxy.i.1:X6

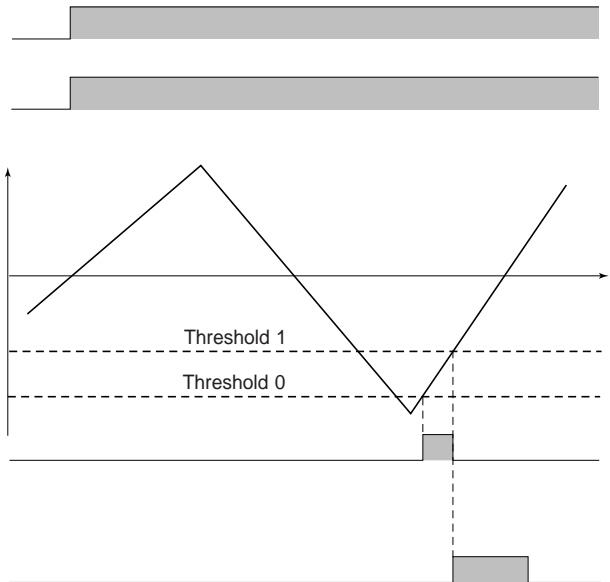
UP/DOWN COUNTER

Current value  
%IDxy.i.0

EVENTS

Crossing event for  
threshold 0 : %IWxy.i.3:X5 and  
+ direction : %IWxy.i.3:X9

Crossing event for  
threshold 1 : %IWxy.i.3:X6 and  
+ direction : %IWxy.i.3:X9



### 4.3 Description of the language objects associated with the function

The language objects associated with up/down counting and measurement function are described in the following tables relating to the **enable**, the **current value**, the **preset**, the **capture**, the **comparison**, the **counter outputs**, the **solid state outputs** and the **frequency output**.

ENABLE	objects	description
Enable event	<b>%IWxy.i.3:X0</b>	object to be tested for <b>state 1</b> in the event-triggered processing (identification of the event) in order to launch the action associated with the enable done.
Enable active	<b>%lxy.i.0</b>	<b>state 1</b> : the up/down counter is enabled, <b>state 0</b> : the up/down counter is inhibited.
Physical enable input <b>IEna</b>	<b>%IWxy.i.2:X2</b>	represents the state of the physical enable input <b>IEna</b> .
Enable	<b>%Qxy.i.5</b>	<b>state 1</b> : enables the physical up/down counter input <b>IEna</b> , <b>state 0</b> : inhibits the physical up/down counter input <b>IEna</b> ,
Direct enable (by program)	<b>%Qxy.i.0</b>	<b>state 1</b> : enables the up/down counter, <b>state 0</b> : inhibits the up/down counter.
Enable event processing	<b>%QWxy.i.1:X0</b>	<b>state 1</b> : the enable event is not masked, <b>state 0</b> : the enable done event is masked (the event is neither processed nor stored).



CURRENT VALUE	objects	description
Current value	%IDxy.i.0	current value of the up/down counter. This <b>word</b> can be read and tested. It is between : -16 777 216 and +16 777 215. In modulo mode, this word is between : 0 and modulo - 1.
Modulo value	%KDxy.i.10	<b>configuration word</b> between 1 and +33 554 432 (default value of the modulo).
Offset value (for an absolute encoder)	%MDxy.i.10	word which can be written, read and tested. It is between -16 777 216 and +16 777 215. In modulo mode, this word is between 0 and the modulo - 1.
Overrun event	%IWxy.i.3:X15	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with an overflow of the stack of PLC events (serious error).
Direction (read)	%lxy.i.9	<b>state 1</b> : The up/down counter upcounts, <b>state 0</b> : The up/down counter downcounts.
Physical counter input <b>IA</b>	%IWxy.i.2:X0	represents the state of the physical up/down counter input <b>IA</b> (for counting pulses or an incremental encoder).
Physical counter input <b>IB</b>	%IWxy.i.2:X1	represents the state of the physical up/down counter input <b>IB</b> (for counting pulses or an incremental encoder).
Physical zero marker input <b>IZ</b>	%IWxy.i.2:X6	represents the state of the incremental encoder physical zero marker input <b>IZ</b> .

CURRENT VALUE (cont.)	objects	description
Change of direction (write)	%QWxy.i.0:X9	<p><b>state 1</b> : the up/down counting direction is positive. The up/down counter upcounts,</p> <p><b>state 0</b> : the up/down counting direction is negative. The up/down counter downcounts.</p>
Status of the serial frame (4 status bits)	%IWxy.i.2:X8 %IWxy.i.2:X9 %IWxy.i.2:X10 %IWxy.i.2:X11	first status bit, second status bit, third status bit, last status bit.
Address of the parallel output encoder	%IWxy.i.10	<p>address of the parallel output encoder, multiplexed via the TELEFAST adaptor ABE-7CPA11.</p> <p>Bit %IWxy.i.10:X1 represents the most significant bit of the address. It is also the rank 2 status bit : %IWxy.i.2:X9 of the serial frame.</p> <p>Bit %IWxy.i.10:X0 represents the least significant bit of the address. It is also the rank 1 status bit : %IWxy.i.2:X10 of the serial frame.</p>

INVALID VALUE	objects	description
Invalid value (channel fault)	%IWxy.i.2:X7	<b>state 1</b> : the current up/down counter value cannot be used, <b>state 0</b> : the current up/down counter value can be used.
Measurement overrun (channel fault)	%MWxy.i.3:X1	<b>Normal mode</b> <b>state 1</b> : the current up/down counter value is less than -16777216 or greater than +16777215, <b>state 0</b> : the current up/down counter value is between -16777216 and +16777215.
Fault acknowledgment	%Qxy.i.3	on <b>a rising edge</b> , this bit is used : <ul style="list-style-type: none"> <li>• if fault memorization has been configured and the faults have disappeared : to reset channel error bit %lxy.i.ERR to zero,</li> <li>• for serial or parallel output absolute encoders (whatever the fault memorization configuration), if the faults have disappeared, to reset the invalid value bit to 0. For counting pulses or incremental encoders, the invalid value bit is reset to zero during preset.</li> </ul>

<b>SPEED</b>	<b>objects</b>	<b>description</b>
Current speed in points / second	<b>%IDxy.i.8</b>	<b>word</b> which can read and tested. It indicates the number of points per second.
Period of measurement in ms	<b>%MDxy.i.27</b>	<b>word</b> which can be read and tested. It is between 10 ms and 30 000 ms. This word is used to define the sampling period at which the speed is calculated and updated.
Overspeed threshold in points / second	<b>%MDxy.i.12</b>	<b>word</b> which can be written, read and tested. It is between 0 and 4 000 000 points/s. This word is automatically compared to the current speed and if this is greater than or equal to the overspeed threshold, the overspeed fault is set to 1. The outputs are forced to 0. <b>The value 0 of this word inhibits the overspeed check.</b>
Overspeed fault	<b>%MWxy.i.3:X3</b>	<b>state 1</b> : the current speed is greater than or equal to the overspeed threshold. <b>state 0</b> : the current speed is less than the overspeed threshold.

PRESET	objects	description
Preset value (for counting pulses or an incremental encoder)	%MDxy.i.4	<b>word</b> which can be written, read and tested. It is between -16 777 216 and +16 777 215. In modulo mode, this word is between 0 and the modulo -1.
Preset event	%IWxy.i.3:X1	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the preset.
Preset done	%lxy.i.1	<b>state 1</b> : when the preset has been done. The preset condition is defined in the configuration (7 modes), <b>state 0</b> : on the rising or falling edge of the preset reset.
Physical preset input <b>IPres</b>	%IWxy.i.2:X3	represents the state of the physical preset input <b>IPres</b> .
Enable preset	%Qxy.i.6	<b>state 1</b> : enables the physical preset input <b>IPres</b> . <b>state 0</b> : inhibits the physical preset input <b>IPres</b> .
Direct preset (by program)	%Qxy.i.1	<b>on a rising edge</b> : sets the current value of the up/down counter to the preset value.
Reset preset	%QWxy.i.0:X1	<b>on a rising or falling edge</b> : sets the preset done bit to 0.
Preset event processing	%QWxy.i.1:X1	<b>state 1</b> : the preset done event is not masked, <b>state 0</b> : the preset done event is masked (the event is neither processed nor stored).

CAPTURE	objects	description
Captured value	%IDxy.i.4	<p>contents of the capture register in the following cases :</p> <ul style="list-style-type: none"> <li>. direct capture,</li> <li>. capture before preset on IPres,</li> <li>. capture on physical input IRead.</li> </ul> <p>This <b>word</b> can be read and tested. It is between -16 777 216 and +16 777 215. In modulo mode, it is between 0 and the modulo -1.</p>
Capture event	%IWxy.i.3:X2	<p>object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the capture.</p>
Edge of read input	%IWxy.i.3:X3	<p><b>state 1</b> : when the capture is made on the falling edge.  <b>state 0</b> : when the capture is made on the rising edge.</p>
Capture done	%Ixy.i.2	<p><b>state 1</b> : when the capture has been done. The capture condition is defined during configuration (3 modes),  <b>state 0</b> : on the rising or falling edge of the capture reset.</p>
Physical read input IRead	%IWxy.i.2:X4	<p>represents the state of the physical read input IRead.</p>
Enable capture	%Qxy.i.7	<p><b>state 1</b> : enables the physical read input IRead,  <b>state 0</b> : inhibits the physical read input IRead,</p>
Direct capture (by program)	%Qxy.i.2	<p><b>on a rising edge</b> : copies the current value of the up/down counter to the capture register.</p>
Reset capture	%QWxy.i.0:X2	<p><b>on a rising or falling edge</b> : sets the capture done bit to 0.</p>
Capture event processing	%QWxy.i.1:X2	<p><b>state 1</b> : the capture done event is not masked,  <b>state 0</b> : the capture done event is masked (the event is neither processed nor stored).</p>

COMPARISON	objects	description
Value of threshold 0	%MDxy.i.6	<b>word</b> which can be written, read and tested. In normal mode, this word is between -16 777 216 and +16 777 215. In modulo mode, this word is between 0 and +33 554 431.
Value of threshold 1	%MDxy.i.8	<b>word</b> which can be written, read and tested. In normal mode, this word is between -16 777 216 and +16 777 215. In modulo mode, this word is between 0 and +33 554 431.
Current value $\geq$ threshold 0	%lxy.i.5	<b>state 1</b> : the current up/down counter value is greater than or equal to the value of threshold 0, <b>state 0</b> : the current up/down counter value is less than the value of threshold 0.
Current value $\geq$ threshold 1	%lxy.i.6	<b>state 1</b> : the current up/down counter value is greater than or equal to the value of threshold 1, <b>state 0</b> : the current up/down counter value is less than the value of threshold 1.
Crossing of modulo, + direction (by the current value)		<b>state 1</b> : the modulo is crossed in the + direction, <b>state 0</b> : on the rising or falling edge of the modulo crossed reset.
Crossing of modulo, - direction (by the current value)		<b>state 1</b> : the modulo is crossed in the - direction, <b>state 0</b> : on the rising or falling edge of the modulo crossed reset.
Reset modulo crossed	%QWxy.i.0:X4	<b>on a rising or falling edge</b> : resets the crossing of the modulo in the + direction and the crossing of the modulo in the - direction.
Captured value $\geq$ threshold 0	%lxy.i.10	<b>state 1</b> : the up/down counter value captured is greater than or equal to the value of threshold 0, <b>state 0</b> : the up/down counter value captured is less than the value of threshold 0.

COMPARISON (cont.)	objects	description
Captured value $\geq$ threshold 1	%Ixy.i.11	<b>state 1</b> : the up/down counter value captured is greater than or equal to the value of threshold 1, <b>state 0</b> : the up/down counter value captured is less than the value of threshold 1.
Direction of threshold crossing	%IWxy.i.3:X9	object to be tested in the event processing, in order to identify the direction of crossing of a threshold. <b>state 1</b> : the threshold is crossed in the + direction <b>state 0</b> : the threshold is crossed in the - direction
Threshold 0 crossing event	%IWxy.i.3:X5	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the crossing of threshold 0.
Threshold 1 crossing event	%IWxy.i.3:X6	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with the crossing of threshold 1.
Modulo crossing event, + direction	%IWxy.i.3:X12	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with crossing of the modulo in the + direction.
Modulo crossing event, - direction	%IWxy.i.3:X13	object to be tested for <b>state 1</b> in the event processing (identification of the event) in order to launch the action associated with crossing of the modulo in the - direction.
Threshold 0 crossing event processing	%QWxy.i.1:X5	<b>state 1</b> : the threshold 0 crossing event is not masked, <b>state 0</b> : the threshold 0 crossing event is masked (the event is neither processed nor stored).
Threshold 1 crossing event processing	%QWxy.i.1:X6	<b>state 1</b> : the threshold 1 crossing event is not masked, <b>state 0</b> : the threshold 1 crossing event is masked (the event is neither processed nor stored).



COMPARISON (cont.)	objects	description
Modulo crossing event processing (+ direction)	<b>%QWxy.i.1:X12</b>	<b>state 1</b> : the modulo crossing event (+ direction) is not masked <b>state 0</b> : the modulo crossing event (+ direction) is masked (the event is neither processed nor stored).
Modulo crossing event processing (- direction)	<b>%QWxy.i.1:X13</b>	<b>state 1</b> : the modulo crossing event (- direction) is not masked <b>state 0</b> : the modulo crossing event (- direction) is masked (the event is neither processed nor stored).
COUNTER OUTPUTS	objects	description
State of counter output 0	<b>%IWxy.i.3:X10</b>	gives the current state of counter output 0 The change of state of counter output 0 is defined in the adjustment.
State of counter output 1	<b>%IWxy.i.3:X11</b>	gives the current state of counter output 1 The change of state of counter output 1 is defined in the adjustment.
Set counter output 0 to 1	<b>%Qxy.i.10</b>	<b>state 1</b> : sets counter output 0 to 1,
Set counter output 1 to 1	<b>%Qxy.i.11</b>	<b>state 1</b> : sets counter output 1 to 1,
Set counter output 0 to 0	<b>%Qxy.i.12</b>	<b>state 1</b> : sets counter output 0 to 0,
Set counter output 1 to 0	<b>%Qxy.i.13</b>	<b>state 1</b> : sets counter output 1 to 0,
SET conditions for counter output 0	<b>%MWxy.i.14</b> :X0 Enable :X1 Preset :X2 Capture  :X4 Crossing of modulo in + direction :X5 Crossing of modulo in - direction	<b>%MWxy.i.15</b> :X0 Crossing of threshold 0 in + direction :X1 Crossing of threshold 0 in - direction :X2 Captured value $\geq$ threshold 0 :X3 Captured value $<$ threshold 0 :X4 Crossing of threshold 1 in + direction :X5 Crossing of threshold 1 in - direction :X6 Captured value $\geq$ threshold 1 :X7 Captured value $<$ threshold 1

COUNTER OUTPUTS (cont.)	objects	description
RESET conditions for counter output 0	<b>%MWxy.i.16</b> :X0 Enable :X1 Preset :X2 Capture  :X4 Crossing of modulo in + direction :X5 Crossing of modulo in - direction	<b>%MWxy.i.17</b> :X0 Crossing of threshold 0 in + direction :X1 Crossing of threshold 0 in - direction :X2 Captured value $\geq$ threshold 0 :X3 Captured value $<$ threshold 0 :X4 Crossing of threshold 1 in + direction :X5 Crossing of threshold 1 in - direction :X6 Captured value $\geq$ threshold 1 :X7 Captured value $<$ threshold 1
SET conditions for counter output 1	<b>%MWxy.i.18</b> :X0 Enable :X1 Preset :X2 Capture  :X4 Crossing of modulo in + direction :X5 Crossing of modulo in - direction	<b>%MWxy.i.19</b> :X0 Crossing of threshold 0 in + direction :X1 Crossing of threshold 0 in - direction :X2 Captured value $\geq$ threshold 0 :X3 Captured value $<$ threshold 0 :X4 Crossing of threshold 1 in + direction :X5 Crossing of threshold 1 in - direction :X6 Captured value $\geq$ threshold 1 :X7 Captured value $<$ threshold 1
RESET conditions for counter output 1	<b>%MWxy.i.20</b> :X0 Enable :X1 Preset :X2 Capture  :X4 Crossing of modulo in + direction :X5 Crossing of modulo in - direction	<b>%MWxy.i.21</b> :X0 Crossing of threshold 0 in + direction :X1 Crossing of threshold 0 in - direction :X2 Captured value $\geq$ threshold 0 :X3 Captured value $<$ threshold 0 :X4 Crossing of threshold 1 in + direction :X5 Crossing of threshold 1 in - direction :X6 Captured value $\geq$ threshold 1 :X7 Captured value $<$ threshold 1

PHYSICAL OUTPUTS Q0 - Q1	objects	description
Enable automatic mode for output Q0 (counter output 0)	%Qxy.i.14	<b>state 1</b> : the state of the physical output Q0 follows the state of counter output 0 in automatic mode (on condition that the output is not tripped and the CTY 2C module is not faulty).
Enable automatic mode for output Q1 (counter output 1)	%Qxy.i.15	<b>state 1</b> : the state of the physical output Q1 follows the state of counter output 1 in automatic mode (on condition that the output is not tripped and the CTY 2C module is not faulty).
Enable automatic mode for output Q3 (programmable frequency output)	%Qxy.i.9	<b>state 1</b> : the state of the physical output Q3 is the same as the programmable frequency output, in automatic mode (on condition that the output has not tripped and the CTY 2C module is not faulty).
Value of the period (programmable frequency output)	%MDxy.i.22	word which can be written, read and tested. It is between 1 and 4 000 000 (ms).
Manual/automatic mode control of output Q0	%QWxy.i.0:X12	<b>state 1</b> : output Q0 is controlled in automatic mode, <b>state 0</b> : output Q0 is controlled in manual mode.
Manual/automatic mode control of output Q1	%QWxy.i.0:X13	<b>state 1</b> : output Q1 is controlled in automatic mode, <b>state 0</b> : output Q1 is controlled in manual mode.
Manual/automatic mode control of output Q3	%QWxy.i.0:X7	<b>state 1</b> : output Q3 is controlled in automatic mode (programmable frequency output). <b>state 0</b> : output Q3 is controlled in manual mode.

PHYSICAL OUTPUTS Q0 - Q1 (cont.)	objects	description
Manual mode state control of output Q0	%QWxy.i.0:X14	<b>state 1</b> : the state of the physical output Q0 is at 1 (on condition that the output is not tripped and the CTY 2C module is not faulty) <b>state 0</b> : output Q0 is at state 0.
Manual mode state control of output Q1	%QWxy.i.0:X15	<b>state 1</b> : the state of the physical output Q1 is at 1 (on condition that the output is not tripped and the CTY 2C module is not faulty) <b>state 0</b> : output Q1 is at state 0.
Manual mode state control of output Q2	%Qxy.i.20	<b>state 1</b> : the state of the physical output Q2 is at 1 (on condition that the output is not tripped and the CTY 2C module is not faulty) <b>state 0</b> : output Q2 is at state 0.
Manual mode state control of output Q3	%Qxy.i.21	<b>state 1</b> : the state of the physical output Q3 is at 1 (on condition that the output is not tripped and the CTY 2C module is not faulty) <b>state 0</b> : output Q3 is at state 0.
State of physical output Q0	%IWxy.i.2:X14	<b>manual mode</b> : the <b>state</b> of physical output Q0 is the same as the manual mode output state control (%QWxy.i.0:X15), <b>automatic mode</b> : the state of physical output Q0 is the same as the state of counter output 0 (%IWxy.i.3:X10), when enabled in automatic mode output Q0 is at 1; otherwise the output is at 0. The output is forced to state 0 when it is tripped or when the CTY 2C module is faulty.

PHYSICAL OUTPUTS Q0 - Q1 (cont.)	objects	description
State of physical output Q1	%IWxy.i.2:X15	<p><b>manual mode</b> : the <b>state</b> of physical output Q1 is the same as the manual mode output state control (%QWxy.i.0:X14),</p> <p><b>automatic mode</b>: the <b>state</b> of physical output Q1 is the same as the state of counter output 1(%IWxy.i.3:X10), when enabled in automatic mode output Q1 is at 1; otherwise the output is at 0.</p> <p>The output is forced to state 0 when it is tripped or when the CTY 2C module is faulty.</p>
State of physical output Q2	%IWxy.i.2:X12	<p><b>manual mode</b> : the <b>state</b> of physical output Q2 is the same as the manual mode output state control (%QWxy.i.0:X4),</p> <p>The output is forced to state 0 when it is tripped or when the CTY 2C module is faulty.</p>
State of physical output Q3	%IWxy.i.2:X13	<p><b>manual mode</b> : the <b>state</b> of physical output Q3 is the same as the manual mode output state control (%QWxy.i.0:X6),</p> <p><b>automatic mode</b>: the <b>state</b> of physical output Q3 is the same as the programmable frequency(%IWxy.i.3:X10), when enabled in automatic mode output Q3 is at 1; otherwise the output is at 0.</p> <p>The output is forced to state 0 when it is tripped or when the CTY 2C module is faulty.</p>
Short-circuit fault on at least one of the outputs Q0, Q1, Q2 and Q3 (channel fault)	%MWxy.i.2:X12	<p><b>state 1</b> : short-circuit on at least one of the physical outputs Q0, Q1, Q2 and Q3.</p> <p><b>state 0</b> : no short-circuits on physical outputs Q0 to Q3.</p>
Reactivate outputs Q0, Q1, Q2 and Q3	%QWxy.i.0:X10	<p><b>on a rising edge</b> : reactivation of the circuit-breaker for physical outputs Q0, Q1, Q2 and Q3.</p>



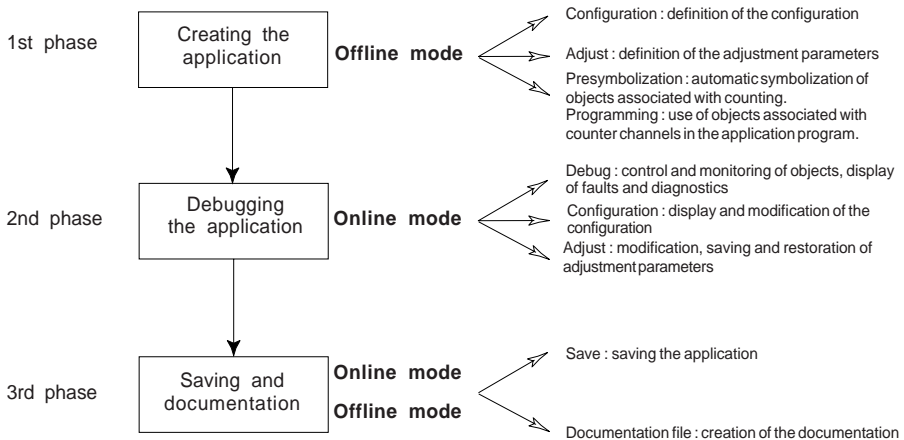
## 5.1 Introduction

Before creating an application program, the physical operating context in which it will be executed must be defined; i.e. the racks and the modules located in the racks : power supply, processor, discrete I/O and application-specific modules (counting, analog, communication, etc).

When up/down counters are used, the counter channel parameters must also be defined (input interface, preset, reactivation of outputs, etc).

The counting setup for an application therefore comprises 3 distinct phases :

- creating the application, which consists of configuring the counter modules, defining the adjustment parameters and setting up the counting in the application program. The presymbolization of objects associated with counter channels simplifies programming, by using mnemonics rather than addresses.
- debugging the application, which consists of optimizing the application, the configuration and adjustment parameters in online mode,
- saving and creating the documentation file.



To start a counting application, the user must :

- perform a preset or reset of the up/down counter (for counting pulses or incremental encoders with a TSX CTY 2A / 4A module),
- acknowledge faults (for an absolute encoder with a TSX CTY 2C module),
- enable the up/down counter,
- unmask the events if necessary.

## 5.1-1 Presymbolization

Application-specific modules enable automatic symbolization of the objects associated with them. The user gives the generic symbol for channel %CHxy.i and all the symbols for objects associated with this channel can then be generated automatically on request. This presymbolization operation, performed in the variables editor, simplifies programming by using mnemonics rather than addresses, which are more difficult to handle.

These objects are symbolized using the following syntax :

### User\_prefix\_Manufacturer\_suffix

where

**User\_prefix** is the generic symbol given by the user to channel %CHxy.i (12 characters maximum),

**Manufacturer\_suffix** is the part of the symbol which corresponds to the channel bit or word and is given by the system (20 characters maximum).

In addition to the symbol, a manufacturer comment is generated automatically which gives a brief description of the role of the object.

**Example :** Motor\_aux2 is the user prefix for channel 0

Variables				
Parameters		#0	Adr: 5:TSX CTY 4A	Entry field
	Address	Type	Symbol	Comment
	%CH5.MOD	CH		
	%IS.MOD.ERR	EBOOL		
	%Mv5.MOD	WORD		
	%Mv5.MOD.1	WORD		
	%Mv5.MOD.2	WORD		
P	%CH5.0	CH		
	%Iv5.0.2	WORD		
	%Iv5.0.3	WORD		
	%ID5.0	DWORD	Motor_aux2_cur_meas	Current counter measurement value
	%ID5.0.4	DWORD	Motor_aux2_capt	Captured counter value
	%Qv5.0	WORD		
	%Qv5.0.1	WORD		
	%Kv5.0	WORD		
	%Kv5.0.1	WORD		
	%Kv5.0.2	WORD		
	%IS.0	EBOOL	Motor_aux2_enab_activ	Counter enable active
	%IS.0.1	EBOOL	Motor_aux2_pres_done	Preset done
	%IS.0.2	EBOOL	Motor_aux2_capt_done	Capture done
	%IS.0.3	EBOOL		
	%IS.0.4	EBOOL		

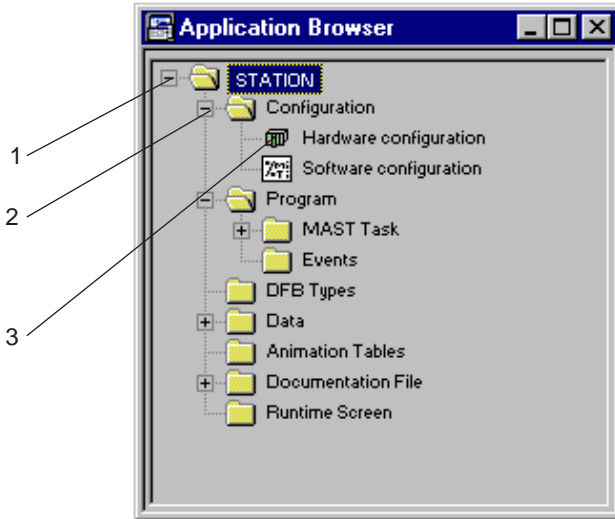


## 5.2 The configuration editor

### 5.2-1 Accessing the configuration editor

The **Application Browser** is used to access the configuration editor. To do this :

- 1 Open the **Station** folder (double-click on the icon or click on its plus sign),
- 2 Open the **Configuration** folder (double-click on the icon or click on its plus sign),
- 3 Double-click on the **Hardware Configuration** icon.



If the **Application Browser** window is not open on the screen :

- Pull down the **Tools** menu and activate the **Application Browser** command,

or

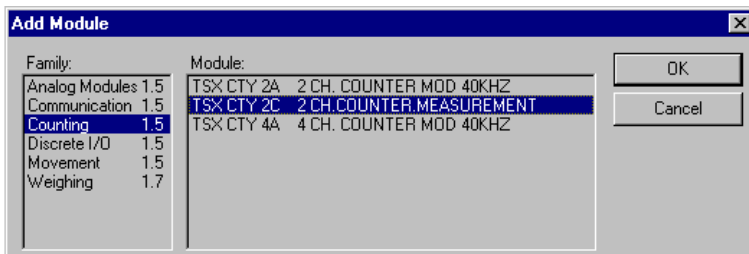
- In the toolbar, click on the Application Browser icon :



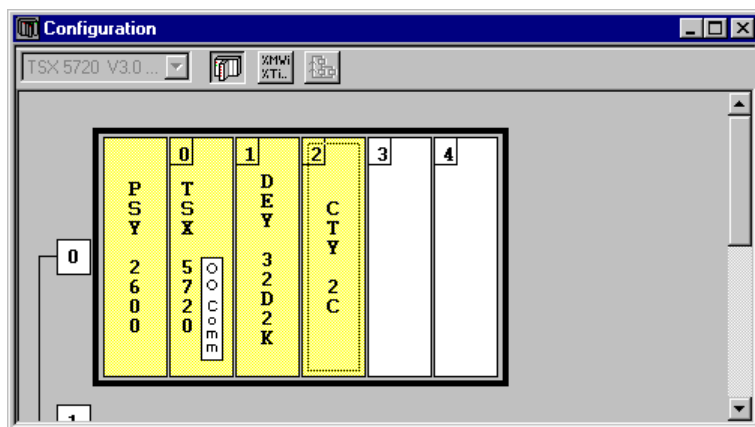
## 5.3 Configuring counter modules

### 5.3-1 Selecting the modules

The module is selected by double-clicking on the position to be configured (for example 2), which displays the following dialog box :



Select the type of module (for example, Counting) from the **Family** field, then select the reference of the module to be configured (for example, TSX CTY 2C) from the **Module** field. By clicking on **OK**, the module is declared in its position (this is framed and contains the module reference).



**Caution** : the maximum number of counter channels which can be installed in a configuration is as follows :

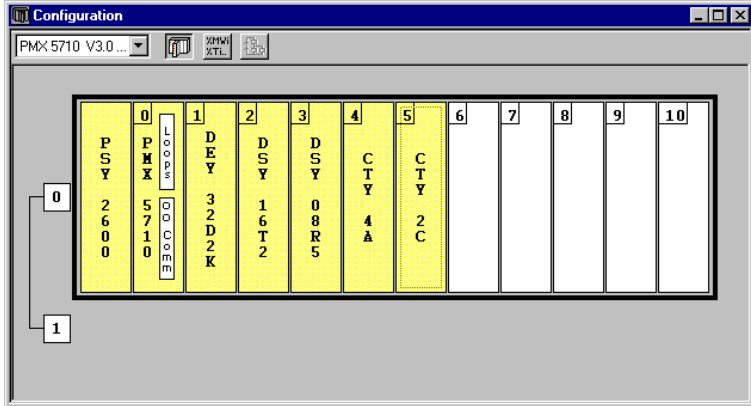
- TSX P57 102 / TPMX P57 102 / TPCX 57 1012 : 8 application-specific channels,
- TSX P57 2•2 / TPMX P57 202 : 24 application-specific channels,
- TSX P57 3•2 / TPMX P57 352 / TPCX 57 3512 : 32 application-specific channels,
- TSX P57 4•2 / TPMX P57 452 : 48 application-specific channels.

An "application-specific" channel is any channel on an intelligent module (counter module, axis control module, etc).

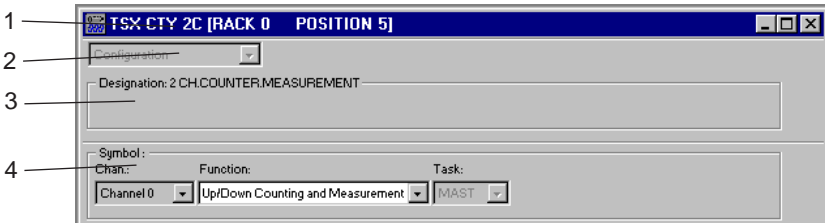
**Note** : to delete a module from its position, click on it to select it and then press the <Del> button, which will display a dialog box. The confirm the deletion of the module.

### 5.3-2 Accessing the parameter settings for a counter module

This is accessed by double-clicking on the representation of the module (for example, TSX CTY 2C module, positioned in slot 2 of rack 0). The parameter settings for a module which has been selected by clicking on it can also be accessed using the **Open Module** command in the **Edit** pulldown menu.



The following parameter setting screen is then displayed :



- 1 This title gives the module catalog reference and its geographic address in the PLC (rack number and position in the rack).
- 2 This command field indicates the current mode (**Configuration** mode) and can be used to select other accessible modes from a pulldown list: **Adjust** mode and **Debug** mode.
- 3 This module zone contains the short title of the module (for example, MEASUREMENT COUNTER MODULE 2 CHANNELS).

---

4 This channel zone can be used to select the counter channel to be configured and the associated counting function :

- **Symbol** : symbol defined by the user (via the variables editor) and associated with the language object of the channel (%CHxy.i).
- **Counter** : used to access the configuration of counter channels : **Counter 0** (channel 0), **Counter 1** (channel 1), etc.
- **Function** : used to define the counting function of the channel and to provide access to the selection of associated parameters : **Upcounting**, **Downcounting** or **Up/down counting** (TSX CTY 2A/ 4A module) and **Up/down counting and measuring** (TSX CTY 2C module). By default, **None** is selected.
- **Task** : used to define the task in which the implicit exchange objects of the channel are exchanged : **MAST** or **FAST** task.

**Note** : the **Function** and **Task** field cannot be modified in online mode.

## 5.4 Channel configuration mode (TSX CTY 2A / 4A / 2C)

This mode can be used to display and modify the configuration parameters of the selected channel relating to the associated counting function.

After modifying the parameters, the new configuration must be confirmed. To do this, pull down the **Edit** menu and select **OK**.

It is also possible to exit the function without confirming the parameters, which will display a dialog box where you can confirm the new configuration.

### 5.4-1 Downcounting, upcounting and up/down counting functions (TSX CTY 2A / 4A)

#### Downcounting function

The screenshot shows the configuration interface for the Downcounting function. At the top, there are three dropdown menus: "Counter 0", "Downcounting", and "MAST". The main interface is divided into several sections:

- Input interfaces:** Contains two dropdown menus. The first is "1 IA input" and the second is "Solid State contact", which is currently selected and highlighted in blue.
- Event:** Contains a checkbox labeled "EVT", which is currently unchecked.
- Reset Output Q0:** Contains two radio buttons: "Manual" (which is selected) and "Automatic".
- Fallback mode:** Contains two radio buttons: "RESET" (which is selected) and "Maintain".
- Preget on IPres:** Contains a dropdown menu set to "Rising Edge IPres" and a small square icon with a red arrow pointing up.
- Operates on Changing to 0:** Contains two radio buttons: "No Downcounter Preset" (which is selected) and "With Downcounter Preset".

#### Upcounting function

The screenshot shows the configuration interface for the Upcounting function. At the top, there are three dropdown menus: "Counter 0", "Upcounting", and "MAST". The main interface is divided into several sections:

- Input interfaces:** Contains two dropdown menus. The first is "1 IA input" and the second is "Solid State contact".
- Event:** Contains a checkbox labeled "EVT", which is currently unchecked.
- Reset Outputs:** Contains two radio buttons: "Manual" (which is selected) and "Automatic".
- Fallback mode:** Contains two radio buttons: "RESET" (which is selected) and "Maintain".
- Preget on IReset:** Contains a dropdown menu set to "Rising Edge IReset" and a small square icon with a red arrow pointing up.
- Operates on Crossing of Setpoint:** Contains two radio buttons: "No Counter Reset" (which is selected) and "With Counter Reset".

## Up/down counting function

The screenshot shows a configuration window for the 'Up/down counting function'. At the top, there are three dropdown menus: 'Counter' set to 'Counter 0', 'Function' set to 'Up/Down Counting', and 'Task' set to 'MAST'. Below these are several sections:

- Input interfaces:** A dropdown menu is set to 'IA Up/Down counts, Application direction'. Below it is a 'Solid State contact' dropdown and a 'Line check' checkbox.
- Multiplication:** Two radio buttons are present: 'By 1' (selected) and 'By 4'.
- Preget on IPres:** A dropdown menu is set to 'Rising Edge IPres' with a small timing diagram icon to its right.
- Read on IRead:** A dropdown menu is set to 'Rising Edge IRead' with a small timing diagram icon to its right.
- Event:** A checkbox for 'EVT' is present.
- Reset Outputs:** Two radio buttons: 'Manual' (selected) and 'Automatic'.
- Fallback mode:** Two radio buttons: 'RESET' (selected) and 'Maintain'.

- **Input interface** (see section 3.2-1)  
Used to define the up/down counter physical input interface.

### The first field (only for up/down counting)

- **IA upcounts, IB downcounts** : the upcounter input is connected to IA and the downcounter input is connected to IB.
- **IA up/down counts, IB direction** : the up/down counter input is connected to IA. The direction, upcounting or downcounting, is defined by the state of input IB,
- **IA up/down counts, application direction** : the up/down counter input is connected to IA. The direction, upcounting or downcounting, is defined by the application program (state of a bit).
- **incremental encoder** : inputs IA, IB and IZ are connected to an incremental encoder. The rotational direction is defined by phase shifting between inputs IA and IB.

### The second field

- **mechanical contact** : the physical input filtering is compatible with this type of sensor (anti-bounce).
- **solid state contact** : reduced filtering of the physical input.

### Line check (for an incremental encoder)

In the case of an RS-422/485 link encoder, this option can be used to signal a channel fault if there is a break in the line between the encoder and inputs IA, IB and IZ.

### Multiplication by 1 or by 4 (for an incremental encoder)

Multiplication by 4 is used to improve the precision of the encoder.




- **Preset on IPres / reset on IReset** (see sections 2.2-3 and 3.2-4)  
Used to define the acceptance of the initialization of the up/down counter :
  - Rising edge of IPres (rising edge of IReset)** : the up/down counter is initialized to the preset value (or value 0) on the rising edge of input IPres (or IReset).
  - **Falling edge of IPres (Falling edge of IReset)** : the up/down counter is initialized to the preset value (or value 0) on the falling edge of input IPres (or IReset).
  - **Rising edge of IPres, + direction / falling edge of IPres, - direction** : the up/down counter is initialized to the preset value :
    - on the rising edge of input IPres (upcounting)
    - on the falling edge of input IPres (downcounting)
  - **Rising edge of IPres - direction / falling edge of IPres + direction** : the up/down counter is initialized to the preset value :
    - on the rising edge of input IPres (downcounting)
    - on the falling edge of input IPres (upcounting)
  - **IPres** : the up/down counter is initialized to the preset value when input IPres is at state 1 (the up/down counter does not evolve as long as IPres remains at 1).
  - **Short cam reference point** : input IPres serves as the reference point. The up/down counter is initialized to the preset value on each rising edge (upcounting) or falling edge (downcounting) of input IZ which follows the change to state 1 of input IPres (see timing diagram in section 3.2-4).
  - **Long cam reference point** : input IPres serves as the reference point. The up/down counter is initialized to the preset value on the first rising edge of input IZ which follows the falling edge of input IPres (see timing diagram in section 3.2-4).
- **Operation on change of state to 0 / Operation on setpoint crossing** (see section 2.2-3)  
Used to define the behavior of the downcounter on change to state 0, or of the upcounter on crossing of the high setpoint.
  - **No downcounter preset/ no upcounter reset**: the downcounter (or the upcounter) is not initialized.
  - **With downcounter preset / with upcounter reset** : the downcounter (or the upcounter) is initialized to the preset value (or to value 0).
- **Capture on IRead** (see section 3.2-5)  
Used to configure the capture; i.e. to copy the current value of the up/down counter to a register on the fly.
  - **Rising edge of IRead** : the current up/down counter value is captured on the rising edge of input IRead.
  - **Falling edge of IRead** : the current up/down counter value is captured on the falling edge of input IRead.

- **Event** (see sections 2.2-7 and 3.2-9)  
Used to associate event processing with the counter channel. If the **EVT** box is checked, the number of the event program which will be executed when the event occurs must be defined:
  - 0 to 31 with a TSX P57 102 / TPMX P57 102 or TPCX 57 1012 processor,
  - 0 to 63 with a TSX P57 2•2 / TSX P57 3•2 / TSX P57 4•2 / TPMX P57 202 / TPMX P57 352 / TPMX P57 452 or TPCX 57 3512 processor.
- **Reactivation of outputs** (see sections 2.2-6 and 3.2-8)  
Used to define the reactivation mode for tripped outputs.
  - **Manual** : physical outputs are reactivated by setting the reactivation bit to 1.
  - **Automatic** : the reactivation of physical outputs is requested by the module.
- **Fallback mode** (see sections 2.2-6 and 3.2-8)  
Used to define the output fallback mode when a fault occurs.
  - **Reset** : the outputs are forced to 0.
  - **Maintain** : the outputs are maintained in the state they were in prior to the fault.

## 5.4-2 Up/down counting and measuring with a TSX CTY 2C module

Chan:	Function:	Task:
Channel 0	Up/Down Counting and Measurement	MAST

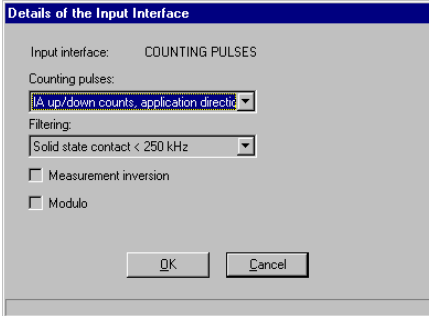
<b>Input interfaces</b> Counting pulses  <input type="button" value="Configure..."/>	<b>Reactivation of outputs</b> <input checked="" type="radio"/> Manual <input type="radio"/> Automatic
<b>Preset on IPres</b> Rising edge IPres 	<b>Fallback mode</b> <input checked="" type="radio"/> RESET <input type="radio"/> Maintain
<b>Read on IRead</b> Rising edge IRead 	<b>Event</b> <input type="checkbox"/> EVT
<input type="checkbox"/> Capture before preset on IPres	<b>Special functions</b> Num.: 0 0 0 0 0 Parameter: <input type="text"/>
<input checked="" type="checkbox"/> Enable on IEna or output Q2 <input checked="" type="checkbox"/> Enable input on IEna <input type="checkbox"/> Output Q2	<b>Faults</b> <input type="checkbox"/> Latch <input type="button" value="Mask..."/>



- **Input interface** (see section 4.2-1)

Used to define the up/down counter physical input interface. The **Configuration** key accesses the configuration screen for the selected interface.

- **Counting pulses** : up/down counting is performed by pulses on physical inputs IA and IB (separate signals).

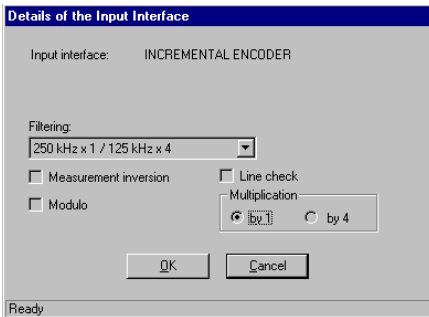


- **Counting pulses** : used to define the acceptance of counting pulses : **IA upcounts, IB downcounts** or **IA up/down counts, IB direction** or **IA up/down counts, application direction** (see previous section).

- **Filtering** : used to define filtering on physical inputs, depending on the type of sensors used : **mechanical contact, solid state contact < 250 kHz, solid state contact < 500 kHz** or **solid state contact < 1 MHz**.

- **Measurement inversion** : used to invert the measurement evolution direction.
- **Modulo** : used to activate modulo mode and to enter its value. Up/down counting is performed in the zone [0, modulo]. The modulo value is between 1 and 33 554 432.

- **Incremental encoder** : the physical inputs are connected to an incremental encoder which supplies two signals phase-shifted by 90°.



- **Filtering** : **250 kHz x1 / 125 kHz x4** (solid state contact < 250 kHz, for multiplication by 1 or < 125 kHz, for multiplication by 4), **500 kHz x1 / 250 kHz x4** (solid state contact < 500 kHz, for multiplication by 1 or < 250 kHz, for multiplication by 4).

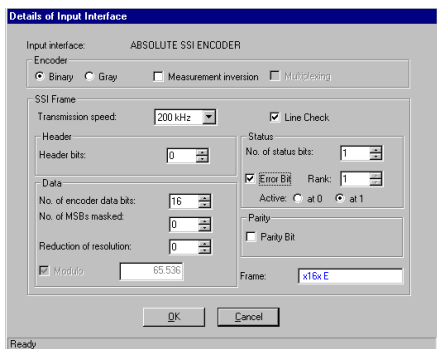
- **Measurement inversion** : used to invert the measurement evolution direction.

- **Modulo** : as for counting pulses.

- **Line check** : used to signal a channel fault if there is a line break or short-circuit.

- **Multiplication by 1 or by 4** : used to improve the precision of the encoder (see previous section).

- **SSI absolute encoder** (only on TSX CTY 2C) : the physical input SSIData and the physical output SSICLK are connected to an SSI serial output absolute encoder.



. **Encoder** : used to define the type of coding (**Binary** or **Gray**).

**Measurement inversion** : used to invert the encoder measurement.

. **SSI frame** : used to define the characteristics of the serial frame :

**Transmission speed** : SSICLK signal frequency : 150 kHz, 200 kHz (by default), 375 kHz, 500 kHz, 750 kHz or 1 MHz.

**Header** : number of serial output absolute encoder header bits : 0 to 4 (0 by default).

**Data** : **Number of encoder data bits** : 8 to 25 (16 by default),  
**Number of MSBs masked** : 17 most significant bits (0 by default),  
**Reduction of resolution** (number of masked least significant bits) :  
 17 least significant bits (0 by default),  
 limited by (minimum of 8 bits of encoder data) :

$$\text{Number of bits of encoder data} - \Sigma \text{ no. of masked most significant bits} - \Sigma \text{ no. of masked least significant bits} \geq 8$$

**Modulo** : value of modulo. For signals emitted by an SSI serial output absolute encoder, up/down counting is performed implicitly in modulo mode. The modulo value is given directly by the number of non-masked encoder data bits.

The up/down counter evolves in the zone [0; modulo[. The minimum modulo value is 1 and the maximum value is +33 554 432.

**Status** : number of serial output absolute encoder status bits : 0 to 4 (0 by default).

**Error bit** : encoder-specific error bit ("None" by default) The "With" option requires that the number of status bits is at least equal to 1 :

. **Rank** : significance of the fault bit specific to the absolute encoder in the status word : rank 1 to 4 (1 by default),

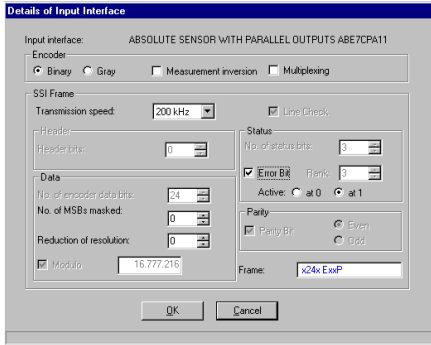
. **Active at 0 / at 1** : level of activity of the fault bit specific to the absolute encoder. By default, this bit is active at 1 (most absolute encoders).

**Parity** : presence or absence of an even or odd parity bit. For odd parity, the number of status bits is limited to a maximum of 3.

**Line check** : presence or absence of line check. It is used to signal a channel fault if there is a short-circuit or a break in the line.

**Frame** : summarizes the characteristics defined for the SSI serial frame.

- **Parallel output absolute sensor with ABE-7CPA11** : the physical input SSIData and the physical output SSICLK are connected to a parallel output absolute encoder (via the ABE-7CPA11 adaptor).



- . **Encoder** : used to define the type of coding (**Binary** or **Gray**).

**Measurement inversion** used to invert the encoder measurement.

**Multiplexing** used to define whether the parallel output encoders are multiplexed.

- . **SSI frame** : used to define the characteristics of the serial frame :

**Transmission speed** : SSICLK signal frequency : 150 kHz, 200 kHz (by default), 375 kHz, 500 kHz, 750 kHz or 1 MHz.

**Header** : number of parallel output absolute encoder header bits : fixed at 0.

**Data** : number of parallel output absolute encoder header bits : 8 to 24 bits (24 bits by default),

**Number of MSBs masked** : 16 most significant bits (0 by default),

**Reduction of resolution** (number of masked least significant bits) : 16 least significant bits (0 bits by default), limited by (minimum of 8 bits of encoder data) :

$$\text{Number of bits of encoder data} - \sum \text{no. of masked most significant bits} - \sum \text{no. of masked least significant bits} \geq 8$$

**Modulo** : value of modulo. For signals emitted by a parallel output absolute encoder, up/down counting is performed implicitly in modulo mode. The modulo value is given directly by the number of non-masked encoder data bits.

The up/down counter evolves in the zone  $[0; \text{modulo}]$ . The minimum modulo value is 1 and the maximum value is +16 777 216.

**Status** : number of parallel output absolute encoder status bits : fixed at 3.

**Error bit** : encoder-specific error bit ("None" by default)

. **Rank** : significance of the fault bit specific to the absolute encoder in the status word : fixed at 3,

. **Active at 0 / at 1** : level of activity of the fault bit specific to the absolute encoder. By default, this bit is active at 1 (most absolute encoders).

**Parity** : presence or absence of an even parity bit.

**Line check** : presence or absence of line check. It is used to signal a channel fault if there is a short-circuit or a break in the line.

**Frame** : summarizes the characteristics defined for the serial frame.

## Multiplexing of parallel output absolute encoders

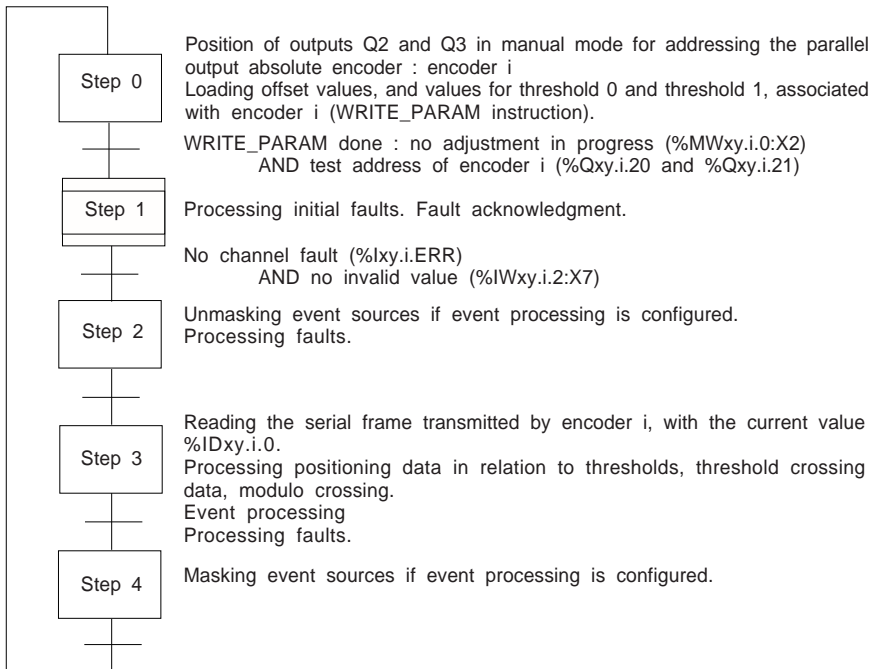
Each channel of the TSX CTY 2C module can be used to acquire, via a serial frame, the signals provided by a parallel output absolute encoder via the ABE-7CPA11 TELEFAST adaptor. Using several TELEFAST adaptors enables multiplexing of up to 4 parallel output absolute encoders on a single counter channel.

Encoder multiplexing address is provided by two discrete outputs (from the TSX CTY 2C module : outputs Q2 and Q3 or from a discrete I/O module) which are looped on to the dedicated TELEFAST inputs. TELEFAST then returns the measurement read and the value of the multiplexing address to the module.

However, the context linked to the encoder (offset value, threshold values, counter output SET and RESET conditions), which must change when a new encoder is addressed, is governed by the application program.

In addition, data relating to the position/crossing of thresholds or the modulo, the speed value and the overspeed fault must not be taken into account when changing the encoder. To perform these operations, the following actions must be performed for each absolute encoder and before each measurement :

- 1 the application program must load the context of the interrogated encoder (offset value, threshold values, counter output SET and RESET conditions),
- 2 the application program must address the parallel output absolute encoder via physical outputs Q2 and Q3 in manual mode or via discrete outputs,
- 3 the data must be read.



- 
- **Preset on IPres** (see sections 4.2-4 and 5.4-1)  
Used to define the acceptance of the preset : Rising edge of IPres, Falling edge of IPres, Rising edge of IPres + direction / Falling edge of IPres - direction, Rising edge of IPres - direction / Falling edge of IPres + direction, IPres, Short cam reference point or Long cam reference point.
  - **Capture on IRead** (see sections 4.2-5 and 5.4-1)  
Used to configure capture : Rising edge of IRead, Falling edge of IRead or Rising and falling edges of IRead.
  - **Capture before preset on IPres** (see section 4.2-5)  
Used to perform a capture before a preset on physical input IPres
  - **Enable on IEna or output Q2** (see section 4.2-8)  
Used to configure a physical auxiliary I/O as enable input IEna or output Q2.
  - **Faults (see section 5.6-3)**
    - **Latch** is used to store the occurrence of a fault. If the option has been configured, channel error bit (%Ixy.i.ERR) is reset to 0 on fault acknowledgment (if the faults are no longer present). This enables transient faults to be stored (without memorizing the cause).
    - **Masking** accesses a list of faults in order to select the faults to be masked when they occur.
  - **Reactivation of outputs** (see sections 4.2-8 and 5.4-1)  
Used to define the reactivation mode for tripped outputs : Manual or Automatic.
  - **Fallback mode** (see sections 4.2-8 and 5.4-1)  
Used to define the output fallback mode when a fault occurs : Reset or Maintain.
  - **Event** (see sections 4.2-9 and 5.4-1)  
Used to associate event processing with the counter channel. If the **EVT** box is checked, the number of the event program (0 to 31) which will be executed when the event occurs must be defined.
  - **Special functions**  
Used to define a special function (not performed by the standard up/down counting and measurement function).
    - **Num** used to enter the numbers of the simultaneous and non-exclusive special function (4 entry fields),
    - **Parameter** used to define the configuration parameter linked to the special functions.
-

## 5.5 Channel adjust mode (TSX CTY 2A / 4A / 2C)

This mode can be used to enter or modify **initial** parameters in **offline** mode. In **online** mode, it can be used to modify the **current** parameters and, if necessary, the **initial** parameters via the **Save Parameters** function.

### TSX CTY 2A / 4A

Adjust

Designation: 4 CH. COUNTER MOD 40KHZ

Symbol:

Counter: Counter 0 Function: Up/Down Counting  Counter output state

Preget value: 100

Initial value: 0

Threshold value:

Threshold0: 200 Initial value: 0

Threshold1: 300 Initial value: 0

Setpoint Values:

High: 500 Initial value: 0

Low: 50 Initial value: 0

Counter output state:

Change counter output state on: C0 C0i C1 C1i

Crossing of threshold0 in +direction R R

Crossing of threshold0 in -direction R R

Captured value >= Threshold0

Captured value < Threshold0

Crossing of threshold1 in +direction S S

Crossing of threshold1 in -direction S S

Captured value >= High setpoint

Crossing of low setpoint in +direction

Crossing of low setpoint in -direction

Actions: Set Reset None

### TSX CTY 2C

Preget value: 100 Initial value: 100

Threshold value:

Threshold0: 200 Initial value: 200

Threshold1: 300 Initial value: 300

Speed monitoring:

Overspeed monitoring: 200 pulses/s Initial value: 200

Measurement period: 10 ms Initial value: 10

Frequency output:

Period: 2 ms Initial value: 2

Counter output state:

Change counter output state on: C0 C0i C1 C1i

Crossing of threshold0 in +direction R R

Crossing of threshold0 in -direction R R

Crossing of threshold1 in +direction S S

Crossing of threshold1 in -direction S S

Actions: Set Reset None

- **Symbol**  
Displays the symbol associated with object %CHxy.i
- **Counter**  
Used to select the channel to be adjusted : Counter 0 (channel 0), Counter 1 (channel 1), etc.
- **Function**  
Provides a reminder of the channel function : Upcounting, Downcounting, etc. This field cannot be modified.
- **Counter output state**  
This checkbox displays the **counter output state** table, for setting the parameters of counter outputs 0 and 1.
- **Preset value** (see sections 2.2-3, 3.2-4 and 4.2-4) (see note below)  
Used to enter the **preset value** (for counting pulses or an incremental encoder). This value must be between -16 777 216 and +16 777 215 in normal mode and between 0 and +33 554 431 in modulo mode. Otherwise, an error is signaled.
- **Offset value** (see note below)  
Used to enter the **offset value** for an absolute encoder (always in modulo mode). It must be between 0 and +33 554 431. Otherwise, an error is signaled.
- **Frequency output** (see note below)  
When output Q3 is used as a programmable frequency output (output Q3 in automatic mode), this field can be used to enter the **value of the period** of output Q3 (100 ms by default). It must be between 1 and 4 000 000 ms, in steps of 1 ms. Otherwise, an error is signaled.
- **Threshold value** (see note below)  
Used to enter the **value of thresholds 0** and **1**. This value must be between -16 777 216 and +16 777 215 in normal mode and between 0 and +33 554 431 in modulo mode or for an absolute encoder. Otherwise, an error is signaled.
- **Setpoint value** (see note below)  
Used to enter the **value of high and low setpoints**. This value must be between -16 777 216 and +16 777 215, otherwise an error is signaled.

**Note**

**In offline mode** : after confirmation (**Edit/Confirm** command), the value entered becomes the **initial value** and appears in the corresponding field. On transfer, the initial value becomes the **current value**.

**In online mode** : after confirmation (**Edit/Confirm** command), the value entered becomes the **current value**. The **Utilities/Save Adjustment Parameters** command is used to copy the current value to the **initial value**.

---

- **Speed monitoring** (see note below)

**Overspeed threshold** : used to enter the overspeed threshold value. If the current speed equals or exceeds this value, an overspeed fault is signaled and the outputs are forced to 0. The value of this threshold must be between 1 and 4 000 000 points/s. Otherwise, an error is signaled.

Value 0 inhibits the overspeed check.

**Measurement period** : used to define the time interval (period) during which the measurement will be taken. The average speed can be deduced from the points acquired during this period.

- **Counter output state**

When this box is checked, the **counter output state** table is displayed, in order to set the parameters for **counter output 0** and **counter output 1** (see sections 2.2-5, 3.2-7 and 4.2-7).

**Counter output state table** : used to define the conditions which will modify the state of **counter outputs 0** and **1**. To do this, complete each field for columns **C0** and **C1**, using the **Set**, **Reset** and **None** keys.

Columns **C0i** and **C1i** show the initial values for changes of state.

- **Set** : sets the counter output to 1 for the selected condition. The letter **S** appears in the selected box **C0** or **C1**.

- **Reset** : resets the counter output to 0 for the selected condition. The letter **R** appears in the selected box **C0** or **C1**.

- **None** : no action on the counter output for the selected condition. The selected box **C0** or **C1** remains empty.

**In offline mode** : after confirmation (**Edit / Confirm** command), the value entered becomes the **initial value**. The symbol **S**, **R** or "no symbol" is then displayed in the corresponding **C0i** and **C1i** boxes.

**Note**

**In offline mode** : after confirmation (**Edit/Confirm** command), the value entered becomes the **initial value** and appears in the corresponding field. On transfer, the initial value becomes the **current value**.

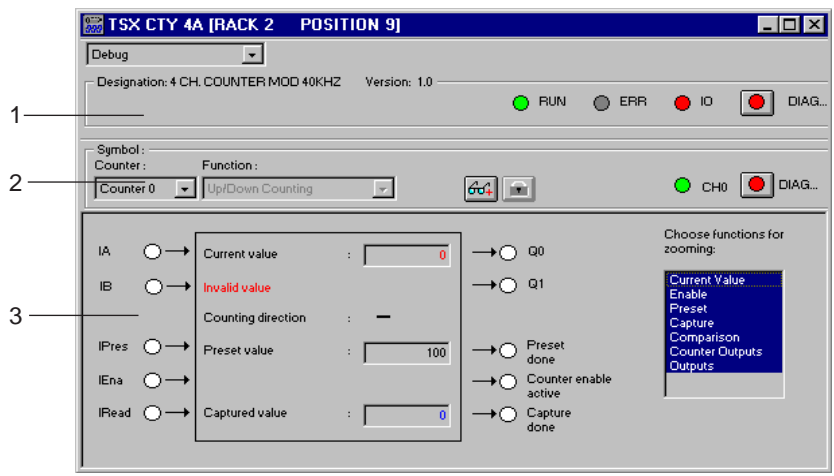
**In online mode** : after confirmation (**Edit/Confirm** command), the value entered becomes the **current value**. The **Utilities / Save Adjustment Parameters** command is used to copy the current value to the **initial value**.



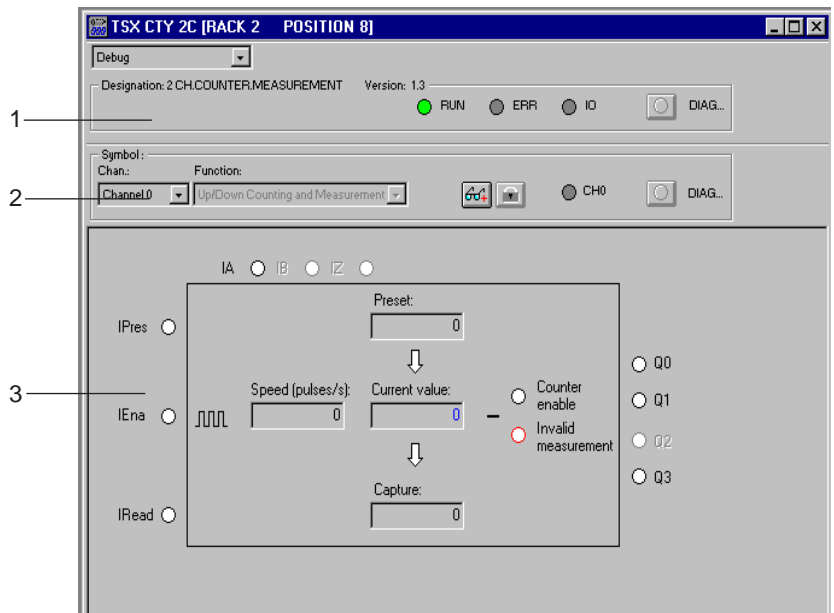
### 5.6 Debug mode (TSX CTY 2A / 4A / 2C)

Debug mode can only be accessed **in online mode**. It can be used to display the state of the inputs for a selected channel, with any faults, and can be used to control language objects (set a bit to 0 / 1, force / unforce an input/output, etc).

#### TSX CTY 2A / 4A



#### TSX CTY 2C



1 This module zone gives the short title and version of the module and displays the state of its indicator lamps in realtime :

- **RUN** : this indicator is on (green), when the module is operating. It is off when the module is faulty or switched off.
- **ERR** : this indicator is on (red), when the module is faulty. It flashes when there is no communication with the processor and is off during normal operation.
- **I/O** : this indicator is on (red) to indicate an "external" fault on one of the channels or an application fault. It is off during normal operation.



DIAG... this button accesses the module diagnostics screen.

2 This channel zone gives the channel symbol (associated with the %CHxy.i object in the data editor) and accesses the debugging function for the various module channels (debug and channel diagnostics screens) :

- **Counter** or **Channel** : used to select the channel to be debugged, which accesses the associated parameters.
- **Function** : reminder of the function configured for the current channel : upcounting, up/down counting and measurement, etc
- **CHx** : this indicator displays the channel status in realtime.



DIAG... this button accesses the channel diagnostics screen.



this button accesses the debug screen in extended mode.




this button is used to unforce all forced I/O.




this icon indicates that certain faults are masked.

3 This channel zone is a parameter zone :

- **IA to IRead** : these indicators show the state of the corresponding physical inputs, when they are configured :

 (indicator on) : the input is at state 1,

 (indicator off) : the input is at state 0,

- counter inputs :

. **IA** : state of physical input IA,

. **IB** : state of physical input IB,

. **IZ** : state of physical input IZ,



- auxiliary inputs :

. **IPres** : state of physical preset input IPres,

. **IEna** : state of physical enable input IEna,

. **IRead** : state of physical capture input IRead.

- **Current value** : this field displays the current value of the up/down counter. This value appears in red when it cannot be used.

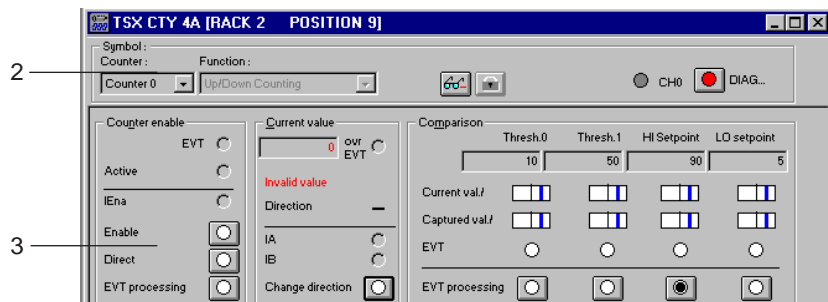
- 
- **Speed** : this field displays the current speed in points/s. This value appears in red when it cannot be used.
  - **Invalid value** : this text (TSX CTY 2A / 4A) or the corresponding indicator (TSX CTY 2C) appears in red when the current value (or the current speed for TSX CTY 2C only) cannot be used.
  - **Counting direction** : signals the module counting direction. The - sign indicates that the channel is downcounting and the + sign that it is upcounting. With a TSX CTY 2C module, only the sign (+ or -) is displayed.
  - **Preset value** or **Preset** : this field displays the current preset value (for counting pulses or an incremental encoder).
  - **Offset** : this field displays the current offset value (for an absolute encoder).
  - **Captured value** or **Capture** : this field displays the captured value.
  - **Address** : this field displays the address of the parallel output absolute encoder when one or more multiplexed parallel output absolute encoders are configured on the channel via the ABE-7CPA11 TELEFAST adaptor.
  
  - **Q0 to Q3** : these indicators display the state of the corresponding physical outputs :
    -  (indicator on) : the output is at state 1,
    -  (indicator off) : the output is at state 0,
    - **Q0** : state of physical output Q0,
    - **Q1** : state of physical output Q1,
    - **Q2** : state of physical output Q2 (when configured),
    - **Q3** : state of physical output Q3.
  - **Preset done** : this indicator shows the state "Preset done".

With a TSX CTY 2C module, the arrow between the Preset and Measurement fields acts as the indicator. The activation edge appears to the left of this arrow and the "done" text appears to the right. This arrow is displayed in blue.
  - **Counter enable active** : this indicator shows the "Enable Active" state.
  - **Capture done** : this indicator shows the "Capture done" state.

With a TSX CTY 2C module, the arrow between the Measurement and Capture fields acts as the indicator. The activation edge appears to the left of this arrow and the "done" text appears to the right. This arrow is displayed in blue.
  - **Choose functions for zooming** (TSX CTY 2A / 4A modules) : this field can be used to select the functions which will be displayed in the debug screen in extended mode (Measurement, Enable, Preset, etc.).
-

### 5.6-1 Debug screen in extended mode (TSX CTY 2A / 4A / 2C)

In this second debug screen, module zone 1 is not displayed, in order to make the screen easier to read. However, it is still possible to use the **View** menu to display this zone or to delete channel zone 2.



- 2 This channel zone is identical to that in the debug screen in reduced mode (see previous pages).



this button returns to the debug screen in reduced mode.

- 3 This channel zone is a zone for displaying parameters and for control :

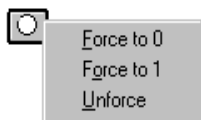
The indicator lamps show the state of the associated bits (physical I/O, event bit, etc) :


- (indicator on) : the bit is at state 1,
- (indicator off) : the bit is at state 0.


The buttons are used to set the associated bits to 1 or 0 (click on the button to change state) :

- clicking on this button sets the associated bit to 0
- clicking on this button sets the associated bit to 1

If an object can be forced, a click with the right mouse button on the corresponding button displays a menu which accesses the forcing commands :



Force to 0, which displays the following button 

Force to 1, which displays the following button 

Unforce, which displays/deletes the letter F from the button.

For manual commands Q2 and Q3, which are used to address multiplexed parallel output absolute encoders, the menu which accesses the forcing commands is extended by 4 additional options :



Force to 0  
Force to 1  
Unforce

Encoder 0  
Encoder 1  
Encoder 2  
Encoder 3

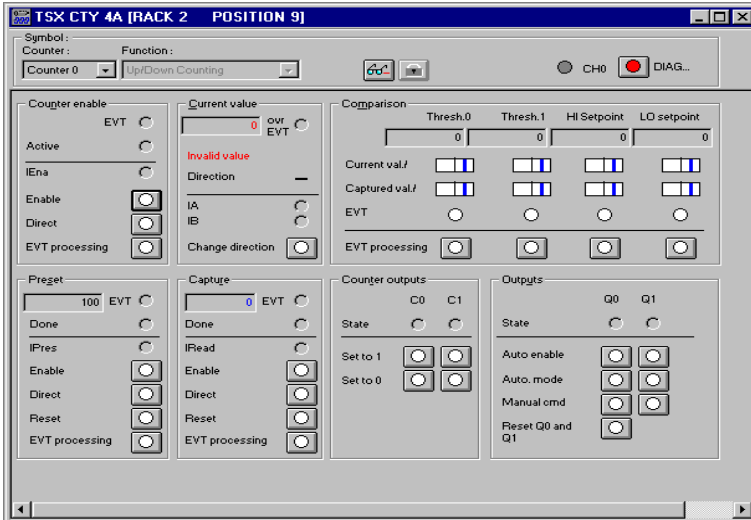
**Encoder 0** : forces outputs **Q2** and **Q3**, to **0** and **0** respectively, in order to address encoder 0,

**Encoder 0** : forces outputs **Q2** and **Q3**, to **1** and **1** respectively, in order to address encoder 1,

**Encoder 0** : forces outputs **Q2** and **Q3**, to **1** and **2** respectively, in order to address encoder 2,

**Encoder 3** : forces outputs **Q2** and **Q3**, to **1** and **1** respectively, in order to address encoder 3,

## TSX CTY 2A / 4A



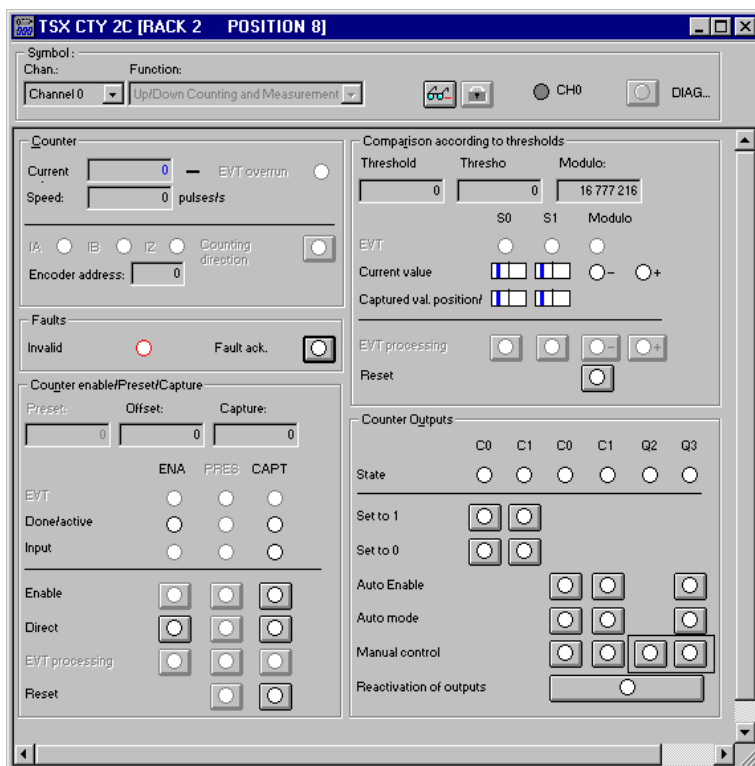
- **Counter enable** functional group
  - **EVT** : state of the enable event,
  - **Active** : state of the enable : active or inactive,
  - **IEna** : state of the physical enable input,
  - **Enable** : command to enable the physical input IEna (can be forced),
  - **Direct** : direct enable command (can be forced),
  - **EVT processing** : command to unmask the enable event (cannot be forced).
- **Preset** functional group (downcounting) or **Reset** functional group (upcounting)
  - **Preset** field : preset value,
  - **EVT** : state of the preset event,
  - **Done** : state of the preset or reset : done or not done,
  - **IPres** or **IReset** : state of the physical preset or reset input,
  - **Enable** : command to enable the physical IPres or IReset input (can be forced),
  - **Direct** : direct preset or reset command (can be forced),
  - **Reset** : command to reset preset or reset counter output (cannot be forced),
  - **EVT Processing** : command to unmask the preset or reset event (cannot be forced).

- **Current value** functional group
  - **Current value** field : current value of the counter. This value appears in red when it cannot be used.
  - **Ovr EVT** : state of event which indicates the overrun of PLC events,
  - **Invalid value** : this text appears in red when the current value cannot be used,
  - **Direction** : indicates the module counting direction : - for downcounting and + for upcounting,
  - **IA** : state of physical counter input IA.
  - **IB** : state of physical counter input IB,
  - **Change direction** : command from the software counting direction input (cannot be forced).
- **Read input (capture)** functional group
  - **Capture** field : captured value,
  - **EVT** : state of the captured event,
  - **Done** : state of the capture : done or not done,
  - **IRead** : state of physical read input,
  - **Enable** : command to enable the physical input IRead (can be forced),
  - **Direct** : direct capture command (can be forced),
  - **Reset** : command to reset the capture counter output (cannot be forced),
  - **EVT processing** : command to unmask the capture event (cannot be forced).
- **Comparison** functional group
  - **Threshold 0** : field containing the current value of threshold 0,
  - **Threshold 1** : field containing the current value of threshold 1,
  - **LO setpoint** : field containing the current value of the high setpoint.
  - **HI setpoint** : field containing the current value of the low setpoint,
  - **Current value/** : position of the measurement in relation to thresholds and setpoints :
    - the current upcounter value is greater than threshold 0, threshold 1, HI setpoint or LO setpoint,
    - the current upcounter value is less than or equal to threshold 0, threshold 1, HI setpoint or LO setpoint,
  - **Capture value/** : position of the capture in relation to thresholds and setpoints
    - the captured value is greater than threshold 0, threshold 1, HI setpoint or LO setpoint,
    - the captured value is less than or equal to threshold 0, threshold 1, HI setpoint or LO setpoint,
  - **EVT** : state of threshold 0, threshold 1, HI setpoint or LO setpoint crossing events,
  - **EVT processing** : commands to unmask the threshold 0, threshold 1, HI and LO setpoint crossing events (cannot be forced).
- **Counter outputs** functional group
  - **State** : state of counter outputs C0 and C1,
  - **Set to 1** : commands to set counter outputs C0 and C1 to 1 (cannot be forced),
  - **Set to 0** : commands to set counter outputs C0 and C1 to 0 (can be forced).

- **Outputs** functional group

- **State** : state of physical outputs Q0 and Q1,
- **Auto enable** : commands to enable outputs Q0 and Q1 in automatic mode (can be forced),
- **Auto mode** : commands to set outputs Q0 and Q1 to automatic mode (cannot be forced),
- **Manual command** : command buttons in manual mode for outputs Q0 and Q1 (cannot be forced),
- **Reset Q0 and Q1** : command to reactivate outputs Q0 and Q1. This key is only active if reactivation is configured in manual mode (cannot be forced).

## TSX CTY 2C





- **Counterfunctional group**
  - **Current** : field containing the current value of the up/down counter. This value appears in red when it cannot be used.
  - **Speed** : field containing the current speed in points/s. This value appears in red when it cannot be used.
  - **+ or -** : shows the counting direction (+ for upcounting or - for downcounting),
  - **EVT Overrun** : state of event which indicates the overrun of PLC events,
  - **IA** : state of physical counter input IA,
  - **IB** : state of physical counter input IB,
  - **IZ** : State of physical zero marker input IZ,
  - **Counting direction** : command from the counting direction input (cannot be forced),
  - **Encoder address** : field containing the address of the parallel output absolute encoder, when one or more parallel output absolute encoders are configured on the channel via the ABE-7CPA11 TELEFAST adaptor.
- **Faultsfunctional group**
  - **Invalid** : this indicator appears in red when the current value or the current speed cannot be used,
  - **Fault ack.** : fault acknowledgment command :
    - . if the fault latching option has been configured and if the faults concerned are no longer present, this command can be used to reset channel error bit %Ixy.i.ERR to 0,
    - . for an SSI serial or parallel output encoder, whatever the state of the fault latching option and if the faults concerned are no longer present, this command can be used to reset the "invalid value" bit to 0.

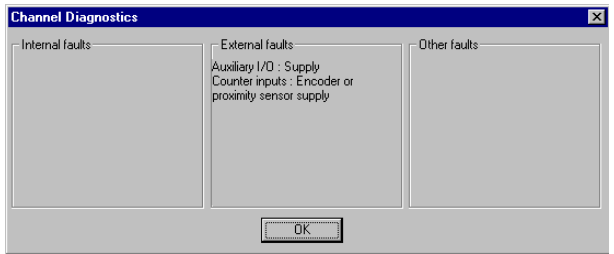
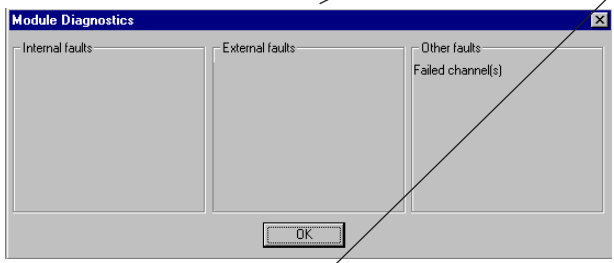
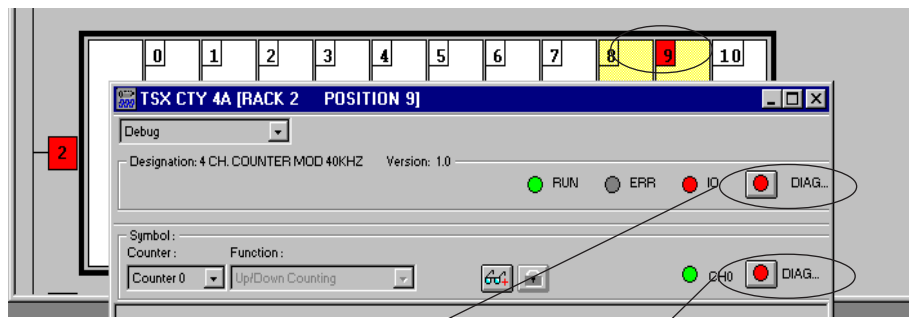
For counting pulses or incremental encoders, this bit is reset to 0 via a preset (preset done or direct preset).
- **Counter enable/Preset/Capture functional group**
  - **Preset** : field containing the preset value (for counting pulses or an incremental encoder),
  - **Offset** : field containing the current offset value (absolute encoder),
  - **Capture** : field containing the captured value,
  - **EVT** : state of enable, preset and capture events,
  - **Done/Active** : enable state (active or inactive), preset state (done or not done) and capture state (done or not done),
  - **Input** : state of physical enable, preset and read inputs,
  - **Enable** : enable commands for physical enable IE<sub>na</sub>, preset IP<sub>res</sub> and read I<sub>read</sub> inputs (can be forced),
  - **Direct** : direct enable, direct preset and direct capture commands (can be forced),
  - **EVT processing** : commands to unmask the enable, preset and capture event commands (cannot be forced),
  - **Reset** : reset to zero (counter output reset) commands for the preset done and capture done (cannot be forced).

- 
- **Comparison according to thresholds** functional group
    - **Threshold 0** : field containing the current value of threshold 0,
    - **Threshold 1** : field containing the current value of threshold 1.
    - **Modulo** : field containing the current value of the modulo,
    - **EVT** : state of crossing of threshold 0, threshold 1 and modulo events,
    - **Current value** : position of the measurement in relation to thresholds and setpoints or to the modulo :
      - the current counter value is greater than threshold 0, threshold 1 or the modulo as appropriate
      - the current counter value is less than or equal to threshold 0, threshold 1 or the modulo as appropriate
    - **Capture val. position/** : position of the capture in relation to thresholds or to the modulo :
      - the captured value is greater than threshold 0, threshold 1 or the modulo as appropriate ,
      - the captured value is less than or equal to threshold 0, threshold 1 or the modulo as appropriate,
    - **EVT processing** : commands to unmask threshold 0, threshold 1, modulo - direction and modulo + direction crossing events (cannot be forced),
    - **Reset** : command for resetting the crossed modulo - direction to 0 (cannot be forced).
  - **Counter Outputs** functional group
    - **State** : State of counter outputs C0 and C1 and physical outputs Q0, Q1, Q2 (when configured) and Q3,
    - **Set to 1** : commands to set counter outputs C0 and C1 to 1 (cannot be forced),
    - **Set to 0** : commands to set counter outputs C0 and C1 to 0 (can be forced).
    - **Auto enable** : commands to enable outputs Q0, Q1 and Q3 in automatic mode (can be forced),
    - **Auto mode** : commands to set outputs Q0, Q1 and Q3 to automatic mode (cannot be forced),
    - **Manual control** : buttons for control in manual mode of outputs Q0, Q1, Q2 (when configured) and Q3 (cannot be forced),
    - **Reactivation of outputs** : command to reactivate outputs Q0, Q1, Q2 (when configured) and Q3. This key is only active if reactivation is configured in manual mode (cannot be forced).

### 5.6-2 Diagnostic screens (TSX CTY 2A / 4A / 2C)

The module and channel diagnostic screens can only be accessed in online mode. A module error is indicated as follows :

- on the module, via the central display unit,
- via dedicated language objects : %lxy.i.ERR, %lxy.i.MOD.ERR, %MWxy.i.MOD.2, etc.
- in the module configuration screen, by the presence of a red square on the faulty counter module position
- in the debug screen via the DIAG button in the channel or module zones, which turns red. Pressing this button accesses the fault diagnostics.



### 5.6-3 Faults and diagnostics (TSX CTY 2A / 4A / 2C)

#### Module

- %lxy.MOD.ERR (at state 1) : indicates the presence of a module fault,
- %MWxy.MOD.2 : this word can be analyzed to diagnose the fault,
- READ\_STS %CHxy.MOD : updates the module diagnostics word.

Address	Error indicated	Action
%MWxy.MOD.2:X0	Module failure : the module is faulty, which causes it to stop functioning.	Change the module
%MWxy.MOD.2:X1	Faulty channel(s) : one or more channels are faulty.	Refer to the channel diagnostics
%MWxy.MOD.2:X3	Self-test : the module is running a self-test.	Wait for the self-test to end
%MWxy.MOD.2:X5	Different hardware and software configuration : there is inconsistency between the configured module and the module in the rack.	Check the correspondence between the configuration and the software configuration
%MWxy.MOD.2:X6	Module missing or switched off : . module missing . module switched off	Insert the module Tighten the fixing screw

#### Channel

- %lxy.0.ERR (at state 1) : indicates the presence of a fault on channel 0,
- %lxy.1.ERR (at state 1) : indicates the presence of a fault on channel 1,
- %MWxy.i.2 and %MWxy.i.3 : these words can be analyzed to diagnose the fault,
- READ\_STS %CHxy.i : updates the channel diagnostics words.

Address	Error indicated	Action
%MWxy.i.2:X0	External fault or counter input fault : . encoder supply fault or proximity sensor fault, . line break or short-circuit fault on at least one of the differential signals of the encoder (IA, IB or IZ), . SSI serial frame fault . Specific absolute encoder fault.  In automatic mode, the outputs are set to 0.	Diagnose the fault more precisely (see %MWxy.i.2:X13 to %MWxy.i.2.15 and %MWxy.i.3:X2)  The up/down counter measurement becomes invalid (%MWxy.i.2:X7)
%MWxy.i.2:X1	Counting application fault : . measurement overrun fault, . overspeed fault.  In automatic mode, the outputs are set to 0.	Diagnose the fault more precisely (see %MWxy.i.3:X1 and %MWxy.i.3:X3)

Address	Error indicated	Action
%MWxy.i.2:X3	Auxiliary I/O fault: <ul style="list-style-type: none"> <li>. 24 V power supply fault on auxiliary I/O,</li> <li>. short-circuit fault on at least one of the outputs.</li> </ul>	Diagnose the fault more precisely (see %MWxy.i.2:X11 and %MWxy.i.2.12)
%MWxy.i.2:X4	Internal fault or channel self-test (module fault has reached the channel) : <ul style="list-style-type: none"> <li>. module failure,</li> <li>. module missing or switched off,</li> <li>. module self-test in progress.</li> </ul>	Refer to the module diagnostics.
%MWxy.i.2:X5	Different hardware and software configurations (module fault returned to the channel) :	Refer to the module diagnostics.
%MWxy.i.2:X6	Communication fault	Check the connections between racks.
%MWxy.i.2:X7	Application fault : channel cannot be configured or adjusted.	Diagnose the fault more precisely (see %MWxy.i.3:X0)
%MWxy.i.2:X8 and %MWxy.i.2:X9	State of the channel (green indicator) : <ul style="list-style-type: none"> <li>. 00 = indicator off</li> <li>. 01 = indicator flashing,</li> <li>. 11 = indicator on.</li> </ul>	
%MWxy.i.2:X11	Auxiliary I/O power supply fault.	Check the 24 V supply to inputs IEna, IPres, IRead and to outputs Q0, Q1 and Q2, Q3 (CTY 2C)
%MWxy.i.2:X12	Short-circuit fault on at least one output.	Check the connection of the auxiliary outputs. Delete the fault and acknowledge if latching of faults is configured
%MWxy.i.2:X13	Encoder or proximity sensor power supply fault.  In automatic mode, the outputs are set to 0.	Check the supply to the encoder and the proximity sensors. Delete the fault and acknowledge if latching of faults is configured. The up/down counter value becomes invalid (%IWxy.i.2:X7) : <ul style="list-style-type: none"> <li>. counting pulses or incremental encoder : preset the up/down counter,</li> <li>. absolute encoder : acknowledge the faults.</li> </ul>

Address	Error indicated	Action
%MWxy.i.2:X14	Counting inputs (encoder line break or short-circuit) : . fault on inputs IA, IB or IZ of an incremental encoder, . fault on SSIData and SSICLK I/O (SSI absolute encoder).  In automatic mode, the outputs are set to 0.	Change the encoder cable. Delete the fault and acknowledge if latching of faults is configured. The up/down counter value becomes invalid (%IWxy.i.2:X7) : . counting pulses or incremental encoder : preset the up/down counter, . absolute encoder : acknowledge the faults.
%MWxy.i.2:X15 (only for TSX CTY 2C)	SSI serial frame fault : . parity error : there is inconsistency between the parity bit transmitted in the frame and the parity calculation, . the frame transmission time is greater than the watchdog.  In automatic mode, the outputs are set to 0.	Delete the fault and acknowledge if latching of faults is configured. The up/down counter becomes invalid (%IWxy.i.2:X7) : . counting pulses or incremental encoder : preset the up/down counter, . absolute encoder : acknowledge the faults.
%MWxy.i.3:X0	Invalid software configurations : . incorrect constant, . combination of bits not associated with any configuration.	Check and modify the configuration constants.
%MWxy.i.3:X1	Measurement overrun fault : . the up/down counter value cannot be used.  In automatic mode, the outputs are set to 0.	Delete the fault and acknowledge if latching of faults is configured. The up/down counter value becomes invalid (%IWxy.i.2:X7) : . counting pulses or incremental encoder : preset the up/down counter, . absolute encoder : the fault cannot occur.

Address	Error indicated	Action
%MWxy.i.3:X2 (only for TSX CTY 2C)	Fault specific to the absolute encoder : error bit set to 1 in the SSI frame  In automatic mode, the outputs are set to 0.	Check the encoder (contamination, increased temperature, etc.). Delete the fault and acknowledge if latching of faults is configured. The up/down counter value becomes invalid (%IWxy.i.2:X7) : . counting pulses or incremental encoder : preset the up/down counter, . absolute encoder : acknowledge the faults.
%MWxy.i.3:X3 (only for TSX CTY 2C)	Overspeed fault : . current speed $\geq$ overspeed threshold. In automatic mode, the outputs are set to 0.	Delete the fault and acknowledge if latching of faults is configured. The up/down counter value becomes invalid (%IWxy.i.2:X7) : . counting pulses or incremental encoder : preset the up/down counter, . absolute encoder : acknowledge the faults.

### Non-latched channel faults

If latching faults is not configured, transient faults will not be detected by the application program (when the fault disappears, the channel error bit is reset to 0). The following faults are however latched using the invalid value fault :

- counter input fault :
  - encoder or proximity sensor power supply,
  - encoder line break or short-circuit fault,
  - SSI frame fault,
  - fault specific to the absolute encoder (if configured).
- counting application fault :
  - measurement overrun,
  - overspeed fault.

For counting pulses or incremental encoders, the invalid value fault is reset to 0 by a preset done or direct preset (via the program).

For absolute encoders, the invalid value fault is reset to 0 by a fault acknowledgment.

## Latched channel faults

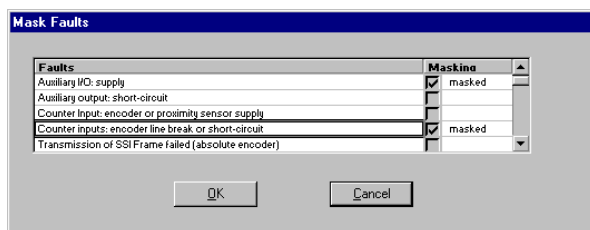
If latching of faults is configured, transient faults will not be detected or stored by the application program. But in this case, the channel error bit remains set to 1 when the fault disappears.

The channel error fault is reset to 0 by a fault acknowledgment.

## Masking channel faults

When a fault is masked, the module ensures the operational safety (resets outputs to 0 in automatic mode - see section 4.2-9) and activates the invalid value fault. However, the fault is not signaled using module and channel error bits, indicators and status words.

Faults are masked using the **Masking** button in the configuration screen which accesses the following dialog box :



The fault is masked by clicking the relevant checkbox. **Masked** is then displayed.

Fault	Fault masking	Comment
1 Counter input fault	No	$\Sigma$ for faults 6, 7, 8 and 10
2 Counter application fault	No	$\Sigma$ for faults 9 and 11
3 Auxiliary I/O fault	No	$\Sigma$ for faults 4 and 5
4 Auxiliary I/O supply fault	Yes	
5 Short circuit fault on at least one of the 4 outputs	No	
6 Encoder or proximity sensor supply fault	Yes	
7 Line break or short-circuit fault on the encoder	No	This fault can be inhibited in the configuration (line check)
8 SSI frame fault	No	
9 Measurement overrun fault	No	
10 Fault specific to the absolute encoder	No	This fault can be inhibited in the configuration of the SSI frame (errorbit)
11 Overspeed fault	No	This fault can be inhibited during adjustment (value of overspeed at 0)



---

## 6.1 Presentation

---

Event processing is used to minimize the reaction time when installing TSX CTY 2A/4A/2C modules :

- extension of (**reflex**) physical outputs Q0 and Q1 to other output module physical outputs,
- extension of physical outputs Q2 and Q3 to output module physical outputs (TSX CTY 2C only),
- programming of actions in event processing.

A counter channel has a corresponding specific application program. This program may be executed in the MAST or FAST tasks : depending on the selection made during configuration. Event processing can be associated with each up/down counter channel; its priority depends on the number of the event.

The occurrence of an event from the counting function diverts the application program (executed in the MAST or FAST task) to the event processing associated with the up/down counter channel.

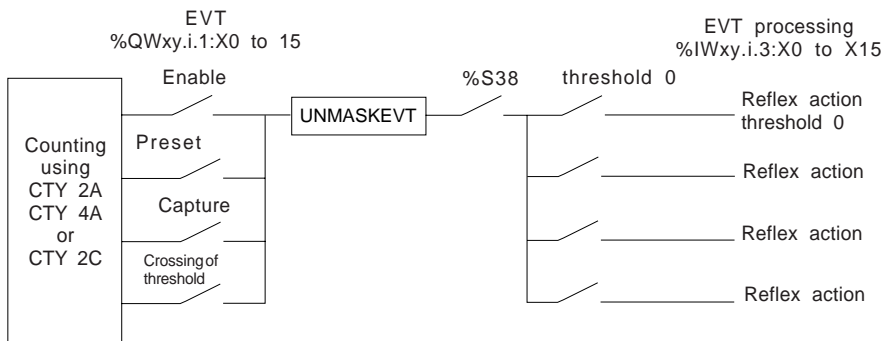
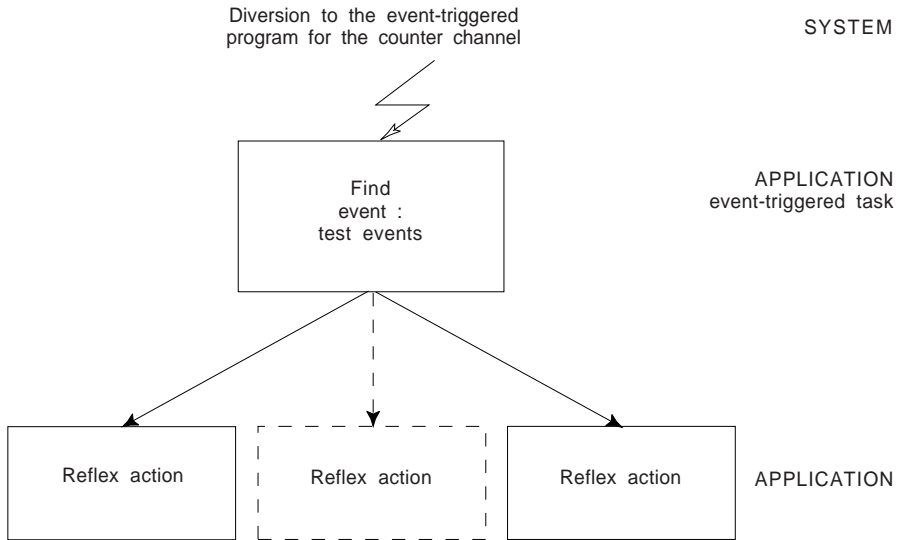
The bits indicating the origin of the event (%IWxy.i.3:X0 to %IWxy.i.3:X15) and the captured value (%IDxy.i.4) are updated before the execution of the event processing. **The other counter objects are not updated.**

During event processing, the user must identify the source of the event by testing the direction of threshold crossing or counter output state events for 1 (%IWxy.i.3:X0 to X15). The user can then launch the associated reflex action via the application program.

Event processing is enabled when :

- bit %S38 for unmasking PL7 events is at state 1,
- the UNMASKEVT instruction for MAST or FAST tasks is used,
- the up/down counter channel events are unmasked (EVT processing objects %QWxy.i.1:X0 to X13).

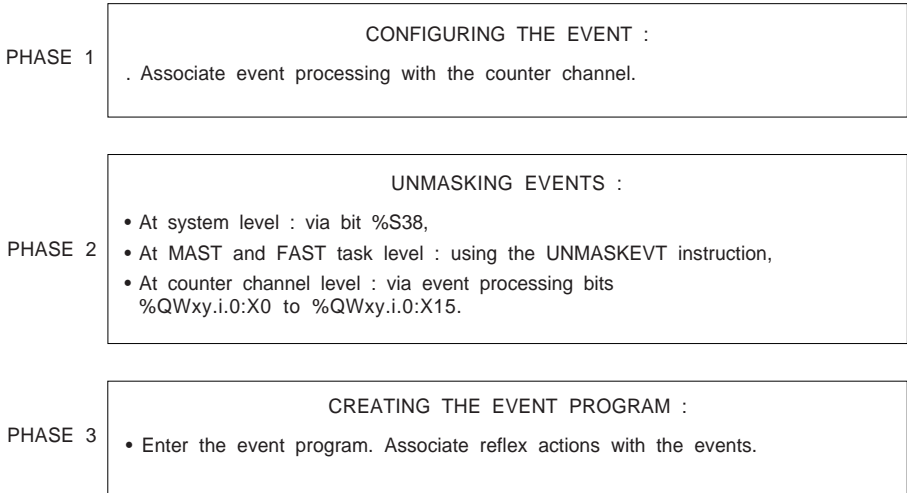
(\*) The direction of crossing of the counter output state.



For more information on event processing, refer to the documentation entitled "Description of PL7 Languages" or "Common Features of Application-Specific Functions".

## 6.2 Methodology for programming event processing

The phases and precautions required to program and execute event processing are as follows :



### 1 Configuration phase

In offline mode, the user must select event processing and an event number for the up/down counting channel.

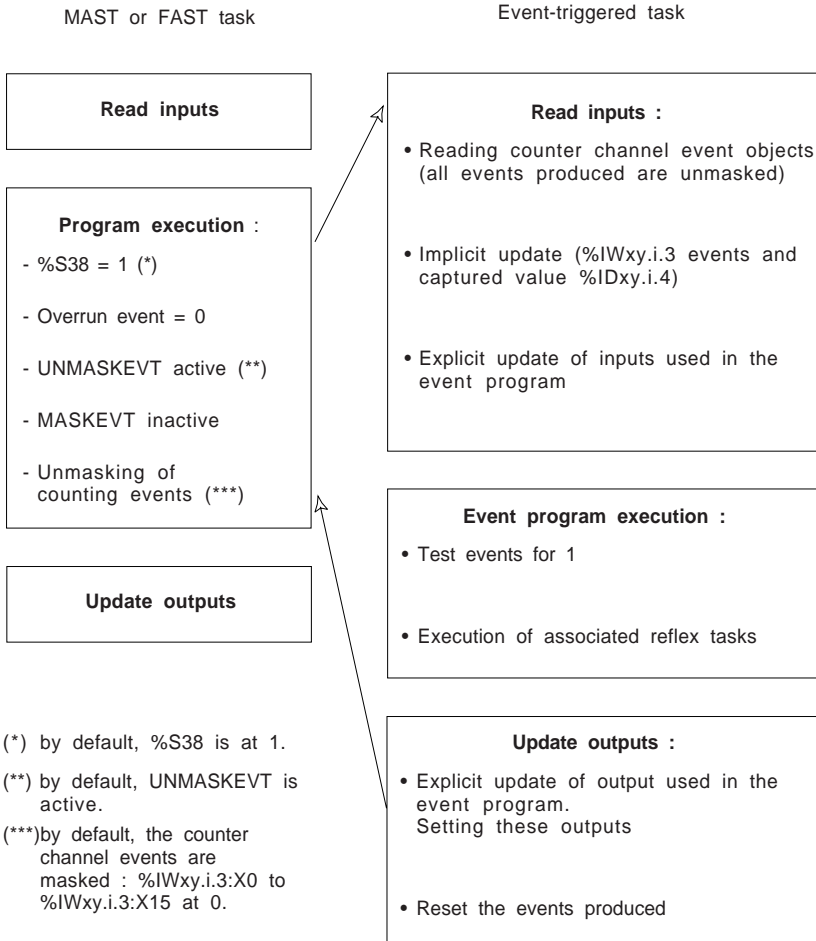
### 2 Event unmasking phase

- Unmasking events using system bit %S38  
For event processing to be executed, bit %S38 must be set to 1. By default, %S38 is at 1.
- Unmasking events using the UNMASKEVT instruction in the MAST and FAST tasks  
For event processing to be taken into account as soon as the event occurs, the program executed in the MAST or FAST task must use the UNMASKEVT instruction. If the MASKEVT instruction is used, event processing will be postponed (the events are memorized) until the next UNMASKEVT instruction. By default, the events are unmasked (UNMASKEVT is active).  
Take note of an event overrun which is indicated by %S39.
- Unmasking counter channel events by program  
For events to trigger event processing on the counter channel, they must be unmasked (%QWxy.i.1:X0 to %QWxy.i.1:X13). By default, the events are masked : %QWxy.i.1:X0 to %QWxy.i.1:X13 are at 0.

### 3 Event program creation phase

Select EVT in the program editor and enter the event program.

## 6.3 Event program execution



## 7 Operating modes (TSX CTY 2A / 4A / 2C)

---

### 7.1 Processing on power breaks and power returns

---

During a power break, the application context and time of the power break are saved. When the power returns, the saved context is compared to the current one :

- if the application context has changed (loss of the system context or new application), the PLC initializes the application : **cold start**,
- if the application context is identical, the PLC performs a restart without data initialization : **warm restart**.

For more detailed information on this subject, see the TSX Premium PLC setup manual.

---

### 7.2 Processing on a warm restart

---

Program execution recommences at the program element where the power break occurred, but the outputs are not set in the first cycle.

The values of objects in counting functions are not modified by a warm restart :

- but when power returns, the current counter value **cannot be used**(invalid value at state 1) because pulses may have been lost during the power break,
- the current and initial values are maintained :
  - the setpoints, the thresholds and the preset value (TSX CTY 2A / 4A),
  - the thresholds, the preset value, the speed measurement period, the overspeed threshold, the programmable frequency output period (TSX CTY 2C).
- counter outputs Q0 and Q1 are reset to 0 (as a result of the invalid value),
- objects %Qxy.i.r and %QWxy.i:X0 to X15 maintain their state; forced objects remain forced.

If the module did not have a power break, the current value %IDxy.i.0 is not changed.

The user must define the processing to be performed on a warm restart (see invalid value).

---

### 7.3 Processing on a cold start

---

During a cold start, the data and the system are initialized.

The values of objects in counting functions are initialized :

- the current counter value **cannot be used** (invalid value at state 1),
- the current values are initialized with the initial values :
  - the setpoints, the thresholds and the preset value (TSX CTY 2A / 4A),
  - the thresholds, the preset value, the speed measurement period, the overspeed threshold, the programmable frequency output period (TSX CTY 2C).
- counter outputs Q0 and Q1 are reset to 0 (as a result of the invalid value),
- the physical outputs Q0, Q1, Q2 (if it is configured) and Q3 are set to 0,
- objects %Qxy.i.r and %QWxy.i:X0 to X15 are set to 0; forced objects are unforced.

If the module did not have a power break, the current value %IDxy.i.0 is not changed (even if the invalid value equals 1).

The user must define the processing to be performed on a cold start (see invalid value).

---

### 7.4 Processing in STOP mode

---

The user program is not executed in STOP mode, the counting function of the CTY 2A/4A/2C module is operational :

- the counter changes as a function of the state of the physical inputs (IA, IB, IPres or IReset, IEna, IRead),
- the physical outputs are in fallback mode (if reset fallback the outputs are at 0, if maintain fallback the outputs maintain their state),
- objects %Qxy.i.r and %QWxy.i,r are not transmitted to the counter module by the TSX 57 processor,
- the counter module transfers objects %lxy.i.r, %IWxy.i.r and %IDxy.i.r to the TSX 57 processor.

---

### 7.5 Reconfiguration in online mode

---

During reconfiguration in online mode :

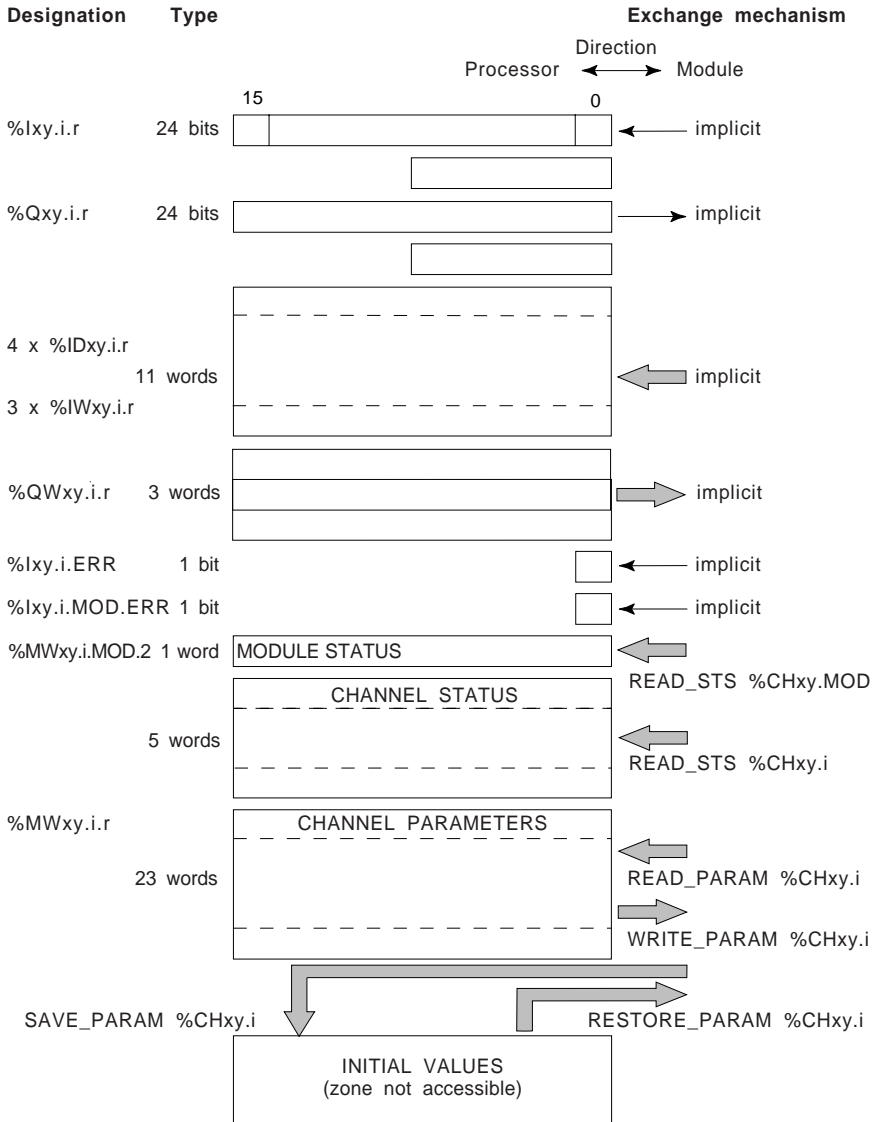
- the current counter value **cannot be used** (invalid value at state 1),
- the current values are initialized with the initial values :
  - the setpoints, the thresholds and the preset value (TSX CTY 2A / 4A),
  - the thresholds, the preset value, the speed measurement period, the overspeed threshold, the programmable frequency output period (TSX CTY 2C).
- objects %lxy.i.r, %lwxy.i.r, %IDxy.i.r, %Qxy.i.r and %QWxy.i.r maintain their state; forced objects remain forced.

## 8 Summary of language objects

## 8.1 Language objects associated with counting

Counter modules configured at a given position automatically generate a set of language objects which enable this counter module to be programmed and diagnosed.

**Summary of objects exchanged per channel and the mechanisms used :**



---

**Implicit exchange objects** : (exchange performed automatically in each scan of the task in which the module channels are configured). These objects are used to access the inputs and software data relating to counting :

- **%Ixy.i.0** to **%Ixy.i.23** : software data (see following sections).
- **%IWxy.i.2:X0** to **%IWxy.i.2:X15** : software data indicating the state of the physical counter inputs (see following sections).
- **%IWxy.i.3:X0** to **%IWxy.i.3:X15** : data indicating the source of the events and the state of the software counter outputs (see following sections).
- **%Qxy.i.0** to **%Qxy.i.23** : software commands (see following sections).
- **%QWxy.i.0:X0** to **%Qxy.i.0:X15** : commands for resetting the stored data, the software direction command (see following sections).
- **%QWxy.i.1:X0** to **%QWxy.i.1:X15** : commands for unmasking or processing events (see following sections).
- **%QWxy.i.2:X0** to **%QWxy.i.2:X15** : reserved for special functions (TSX CTY 2C).
- **%IDxy.i.0** : current counter value.
- **%IDxy.i.4** : captured counter value.
- **%IDxy.i.8** : speed (number of pulses per second).
- **%IDxy.i.10** : multiplexed address of parallel output absolute encoder (CTY 2C).
- **%IDxy.i.11** : reserved for special functions.
- **%CHxy.MOD** : address of the counter module. In particular, allows explicit exchanges to be sent.
- **%Ixy.i.ERR** : counter channel fault. At state 1 indicates that channel **i** of the module in position **xy** is faulty. The causes of the fault are listed in words **%MWxy.i.2** and **%MWxy.i.3**.
- **%Ixy.MOD.ERR** : module fault. At state 1 indicates that the module in position **xy** is faulty. The causes of the fault are listed in word **%MWxy.MOD.2**.

**Event-triggered exchange objects only** :

- **%IDxy.i.6** : reserved for special functions (TSX CTY 2C).



**Explicit exchange objects** : (exchange via instructions initiated by the application) : these objects are used to access the diagnostics and configuration relating to counting :

- **%MDxy.i.4** : preset value.
- **%MDxy.i.6** : value of threshold 0.
- **%MDxy.i.8** : value of threshold 1.
- **%MDxy.i.10** : value of the high setpoint (TSX CTY 2A/4A) or the offset value for an absolute encoder (TSX CTY 2C).
- **%MDxy.i.12** : value of the low setpoint (TSX CTY 2A/4A) or of the overspeed (TSX CTY 2C).
- **%MDxy.i.22** : time base period of frequency output Q3 (TSX CTY 2C).
- **%MDxy.i.24** : reserved for special functions (TSX CTY 2C).
  
- **%MWxy.i.0:X0** to **%MWxy.i.0:X15** : indicators for the execution during an explicit exchange (see following sections).
- **%MWxy.i.1:X0** to **%MWxy.i.1:X15** : indicators for the execution errors during an explicit exchange (see following sections).
- **%MWxy.i.2:X0** to **%MWxy.i.2:X15** : diagnostics of faults on counter channel *i* of the module in position *xy* (see following sections).
- **%MWxy.i.3:X0** to **%MWxy.i.3:X15** : diagnostics of faults on counter channel *i* of the module in position *x* (see following sections).
- **%MWxy.i.14:X0** to **%MWxy.i.14:X15** : SET conditions of counter output 0 (see following sections).
- **%MWxy.i.15:X0** to **%MWxy.i.15:X15** : SET conditions of counter output 0 (see following sections).
- **%MWxy.i.16:X0** to **%MWxy.i.16:X15** : RESET conditions of counter output 0 (see following sections).
- **%MWxy.i.17:X0** to **%MWxy.i.17:X15** : RESET conditions of counter output 0 (see following sections).
- **%MWxy.i.18:X0** to **%MWxy.i.18:X15** : SET conditions of counter output 1 (see following sections).
- **%MWxy.i.19:X0** to **%MWxy.i.19:X15** : SET conditions of counter output 1 (see following sections).
- **%MWxy.i.20:X0** to **%MWxy.i.20:X15** : RESET conditions of counter output 1 (see following sections).
- **%MWxy.i.21:X0** to **%MWxy.i.21:X15** : RESET conditions of counter output 1 (see following sections).
- **%MWxy.i.26** : reserved for special functions (TSX CTY 2C).
- **%MWxy.i.27** : speed measurement period (TSX CTY 2C).
- **%MWxy.i.28** : reserved for special functions (TSX CTY 2C).
  
- **%MWxy.MOD.2:X0** to **%MWxy.MOD.2:X15** : diagnostics of faults on module in position *xy* (see following sections).

### 8.1-1 Implicit exchange objects

<b>Objects : %IWxy.i.2:X0 to %IWxy.i.2:X15</b>	
<b>Bit</b>	<b>Software data indicating the state of the physical counter I/O</b>
0	State of physical counter input <b>IA</b>
1	State of physical counter input <b>IB</b>
2	State of physical enable input <b>IEna</b>
3	State of the physical preset input <b>IPres</b> or <b>IReset</b>
4	State of physical read input <b>IRead</b>
6	State of physical counter input <b>IZ</b>
7	Invalid value
8	Status bit, rank 1 of the SSI frame, or For an SSI absolute encoder, odd parity (not checked by the module) : odd parity bit, or For a parallel output absolute encoder with ABE-7CPA11 adaptor sub-base : least significant bit of the encoder address.
9	Status bit, rank 2 of the SSI frame, or For a parallel output absolute encoder with ABE-7CPA11 adaptor sub-base : most significant bit of the encoder address.
10	Status bit, rank 3 of the SSI frame, or For a parallel output absolute encoder with ABE-7CPA11 adaptor sub-base : fault bit specific to the absolute encoder.
11	Status bit, rank 4 of the SSI frame
12	State of output Q2
13	State of output Q3
14	State of output Q0
15	State of output Q1

Objects : %lxy.i.0 to %lxy.i.23		
Rank	Software data	
0	Enable active	
1	Preset done	
2	Capture done	
5	Current value $\geq$ threshold 0 ( $\leq 0$ in downcounting)	(TSX CTY 2A/4A)
6	Current value $\geq$ threshold 1	
7	Current value $\geq$ high setpoint	(TSX CTY 2A/4A)
8	Current value $\geq$ low setpoint	(TSX CTY 2A/4A)
9	Counting direction	
10	Captured value $\geq$ threshold 0	
11	Captured value $\geq$ threshold 1	
12	Captured value $\geq$ high setpoint Modulo crossing, + direction	(TSX CTY 2A/4A) (TSX CTY 2C)
13	Captured value $\geq$ low setpoint Modulo crossing, - direction	(TSX CTY 2A/4A) (TSX CTY 2C)
16...21	Reserved for special functions	(TSX CTY 2C)

<b>Objects : %Qxy.i.0 to %Qxy.i.23</b>	
<b>Rank</b>	<b>Software commands</b>
0	Direct enable by program
1	Direct preset by program
2	Direct capture by program
3	Fault acknowledgment (TSX CTY 2C)
5	Enable physical enable input
6	Enable physical preset input
7	Enable physical capture input
9	Enable output Q3 in automatic mode (TSX CTY 2C)
10	Set counter output 0 to 1
11	Set counter output 1 to 1
12	Set counter output 0 to 0
13	Set counter output 1 to 0
14	Automatic enable output Q0
15	Automatic enable output Q1
16...19	Reserved for special functions (TSX CTY 2C)
20	Manual control, output Q2 (TSX CTY 2C)
21	Manual control, output Q3 (TSX CTY 2C)

<b>Objects : %QWxy.i.0:X0 to %QWxy.i.0:X15</b>	
<b>Bit</b>	<b>Reset + change direction commands</b>
1	Reset preset
2	Reset capture
4	Reset crossing of modulo (TSX CTY 2C)
9	Counting direction
10	Reactivation of outputs Q0, Q1 and outputs Q2, Q3 (TSX CTY 2C)
11	Manual/automatic mode, output Q3 (frequency) (TSX CTY 2C)
12	Manual/automatic mode, output Q0
13	Manual/automatic mode, output Q1
14	Manual control, output Q0
15	Manual control, output Q1

<b>Objects : %IWxy.i.3:X0 to %IWxy.i.3:X15</b>	
<b>Bit</b>	<b>Data on the events + counter output state</b>
0	Enable event
1	Preset or reset event
2	Capture event
3	Direction of capture edge (TSX CTY 2C only)
5	Threshold 0 crossing event (or crossing of value 0 in downcounting with TSX CTY 2A/4A)
6	Threshold 1 crossing event
7	High setpoint crossing event (TSX CTY 2A/4A) Reserved for special functions (TSX CTY 2C)
8	Low setpoint crossing event (TSX CTY 2A/4A) Reserved for special functions (TSX CTY 2C)
9	Direction when crossing threshold or setpoint (TSX CTY 2A/4A)
10	State of counter output 0
11	State of counter output 1
12	Crossing of modulo in + direction event (TSX CTY 2C)
13	Crossing of modulo in - direction event (TSX CTY 2C)
15	Overrun event

<b>Objects : %QWxy.i.1:X0 to %QWxy.i.1:X15</b>	
<b>Bit</b>	<b>Event unmasking or event processing commands</b>
0	Enable event processing
1	Preset or reset event processing
2	Capture event processing
5	Threshold 0 event processing
6	Threshold 1 event processing
7	High setpoint event processing (TSX CTY 2A/4A)
8	Low setpoint event processing (TSX CTY 2A/4A)
12	Crossing of modulo in + direction event processing (TSX CTY 2C)
13	Crossing of modulo in - direction event processing (TSX CTY 2C)

## 8.1-2 Explicit exchange objects

Bit	Objects : %MWxy.i.0:X0 to %MWxy.i.0:X15 Explicit exchange execution indicators
0	Reading channel status in progress
2	Adjustment in progress
15	Reconfiguration in progress

Bit	Objects : %MWxy.i.1:X0 to %MWxy.i.1:X15 Explicit exchange execution errors
2	Adjustment error
15	Reconfiguration error

Bit	Objects : %MWxy.i.14:X0 to %MWxy.i.14:X15 SET conditions for counter output 0
0	Enable
1	Preset or reset
2	Capture
4	Crossing of modulo in + direction (TSX CTY 2C)
5	Crossing of modulo in - direction (TSX CTY 2C)

<b>Objects : %MWxy.i.15:X0 to %MWxy.i.15:X15 SET conditions for counter output 0 (continued)</b>		
<b>Bit</b>		
0	Crossing of threshold 0 in increasing direction	
1	Crossing of threshold 0 in decreasing direction	
2	Captured value $\geq$ threshold 0	
3	Captured value $<$ threshold 0	
4	Crossing of threshold 1 in increasing direction	
5	Crossing of threshold 1 in decreasing direction	
6	Captured value $>$ threshold 1	
7	Captured value $<$ threshold 1	
8	Crossing of high setpoint in increasing direction	(TSX CTY 2A/4A)
9	Crossing of high setpoint in decreasing direction	(TSX CTY 2A/4A)
10	Captured value $\geq$ high setpoint	(TSX CTY 2A/4A)
12	Crossing of low setpoint in increasing direction	(TSX CTY 2A/4A)
13	Crossing of low setpoint in decreasing direction	(TSX CTY 2A/4A)
15	Captured value $<$ low setpoint	(TSX CTY 2A/4A)

<b>Objects :%MWxy.i.16:X0 to %MWxy.i.16:X15 RESET conditions for counter output 0</b>		
<b>Bit</b>		
0	Enable	
1	Preset or reset	
2	Capture	
4	Crossing of modulo in + direction	(TSX CTY 2C)
5	Crossing of modulo in - direction	(TSX CTY 2C)

<b>Objects : %MWxy.i.17:X0 to %MWxy.i.17:X15</b>		
<b>Bit</b>	<b>RESET conditions for counter output 0 (continued)</b>	
0	Crossing of threshold 0 in increasing direction	
1	Crossing of threshold 0 in decreasing direction	
2	Captured value $\geq$ threshold 0	
3	Captured value $<$ threshold 0	
4	Crossing of threshold 1 in increasing direction	
5	Crossing of threshold 1 in decreasing direction	
6	Captured value $\geq$ threshold 1	
7	Captured value $<$ threshold 1	
8	Crossing of high setpoint in increasing direction	(TSX CTY 2A/4A)
9	Crossing of high setpoint in decreasing direction	(TSX CTY 2A/4A)
10	Captured value $\geq$ high setpoint	(TSX CTY 2A/4A)
12	Crossing of low setpoint in increasing direction	(TSX CTY 2A/4A)
13	Crossing of low setpoint in decreasing direction	(TSX CTY 2A/4A)
15	Captured value $<$ low setpoint	(TSX CTY 2A/4A)

<b>Objects : %MWxy.i.18:X0 to %MWxy.i.18:X15</b>		
<b>Bit</b>	<b>SET conditions for counter output 1</b>	
0	Enable	
1	Preset or reset	
2	Capture	
4	Crossing of modulo in + direction	(TSX CTY 2C)
5	Crossing of modulo in - direction	(TSX CTY 2C)



<b>Objects : %MWxy.i.19:X0 to %MWxy.i.19:X15</b>		
<b>Bit</b>	<b>SET conditions for counter output 1 (continued)</b>	
0	Crossing of threshold 0 in increasing direction	
1	Crossing of threshold 0 in decreasing direction	
2	Captured value $\geq$ threshold 0	
3	Captured value $<$ threshold 0	
4	Crossing of threshold 1 in increasing direction	
5	Crossing of threshold 1 in decreasing direction	
6	Captured value $\geq$ threshold 1	
7	Captured value $<$ threshold 1	
8	Crossing of high setpoint in increasing direction	(TSX CTY 2A/4A)
9	Crossing of high setpoint in decreasing direction	(TSX CTY 2A/4A)
10	Captured value $\geq$ high setpoint	(TSX CTY 2A/4A)
12	Crossing of low setpoint in increasing direction	(TSX CTY 2A/4A)
13	Crossing of low setpoint in decreasing direction	(TSX CTY 2A/4A)
15	Captured value $<$ low setpoint	(TSX CTY 2A/4A)

<b>Objects : %MWxy.i.20:X0 to %MWxy.i.20:X15</b>		
<b>Bit</b>	<b>RESET conditions for counter output 1</b>	
0	Enable	
1	Preset or reset	
2	Capture	
4	Crossing of modulo in + direction	(TSX CTY 2C)
5	Crossing of modulo in - direction	(TSX CTY 2C)

<b>Objects : %MWxy.i.21:X0 to %MWxy.i.21:X15</b>		
<b>Bit</b>	<b>RESET conditions for counter output 1 (continued)</b>	
0	Crossing of threshold 0 in increasing direction	
1	Crossing of threshold 0 in decreasing direction	
2	Captured value $\geq$ threshold 0	
3	Captured value $<$ threshold 0	
4	Crossing of threshold 1 in increasing direction	
5	Crossing of threshold 1 in decreasing direction	
6	Captured value $\geq$ threshold 1	
7	Captured value $<$ threshold 1	
8	Crossing of high setpoint in increasing direction	(TSX CTY 2A/4A)
9	Crossing of high setpoint in decreasing direction	(TSX CTY 2A/4A)
10	Captured value $\geq$ high setpoint	(TSX CTY 2A/4A)
12	Crossing of low setpoint in increasing direction	(TSX CTY 2A/4A)
13	Crossing of low setpoint in decreasing direction	(TSX CTY 2A/4A)
15	Captured value $<$ low setpoint	(TSX CTY 2A/4A)

<b>Objects : %MWxy.MOD.2:X0 to %MWxy.MOD.2:X15</b>		
<b>Bit</b>	<b>Module fault diagnostics</b>	
0	Module failure	
1	Faulty channel(s)	
5	Different hardware or software configurations	
6	Module missing or switched off	

<b>Objects : %MWxy.i.2:X0 to %MWxy.i.2:X15</b>		
<b>Bit</b>	<b>Counter channel fault indicators</b>	
0	External fault : counter inputs fault	
1	Counting application fault	
3	External fault : auxiliary I/O	
4	Internal fault (or channel performing self-test)	
5	Module hardware and software configuration fault	
6	Communication fault	
7	Application program fault (adjustment or configuration fault)	
8-9	Channel status (CH indicator lamp)	
11	Auxiliary input supply fault	
12	Short-circuit fault on at least one of the two channels	
13	Encoder or proximity sensor power supply fault	
14	Break on at least one of the signals	
15	SSI frame transmission fault (parity)	(TSX CTY 2C)

<b>Objects : %MWxy.i.3:X0 to %MWxy.i.3:X15</b>		
<b>Bit</b>	<b>Counter channel fault indicators</b>	
0	Channel configuration fault	
1	Measurement overrun fault	
2	Fault specific to the absolute encoder	(TSX CTY 2C)
3	Overspeed fault	(TSX CTY 2C)
8-9	Reserved for special functions	(TSX CTY 2C)

## 8.2 Addressing objects

The addressing of the main counter module bit and word objects is defined in the following sections.

The principle of addressing is as follows:

%	I, Q, M, K	W, D	xy	.	i	.	r
IEC1131 sign	<b>Object type</b> I = input Q = output M = internal word K = internal constant	<b>Format</b> W = word D = double word	<b>Position</b> x = rack number y = slot number	<b>Channel no.</b> i = 0 to 127 or MOD	<b>Rank</b> r = 0 to 255 or ERR		

### Additional information

- **Object type :**

**I, Q** : implicit exchange I/O objects.

**M** : read or write data exchanged at the request of the application (explicit exchanges).

**K** : configuration data only available in read mode.

Example : **%MD4.1.4** : preset value, downcounting, CTY 2A module **position 4, channel 1**.

- **Channel no. :** (i= 0 to 1 on CTY 2A/2C, i= 0 to 3 on CTY 4A).

**MOD**: channel reserved for the management of the module and parameters common to the channels it supports.

Example : **%I4.MOD.ERR**: information about a fault on the counter module at **position 4**.

- **Rank :**

Used to address different objects of the same type associated with a single channel.

Example : **%IW4.1.3:X10** : word extract bit 10 **%IW4.1.3** which represents the state of the comparison counter output for downcounting on a CTY 2A module at **position 4, channel 1, rank 3** of this word.

**ERR** : indicates a module or channel fault.

Example : **%I4.1.ERR** : information about a fault on the counter module at **position 4, channel 1**.

The variables editor provides access to all objects associated with a module, by selecting "I/O" from the pulldown list and the module position from the "Module Address" zone **if the counter channels are configured**.

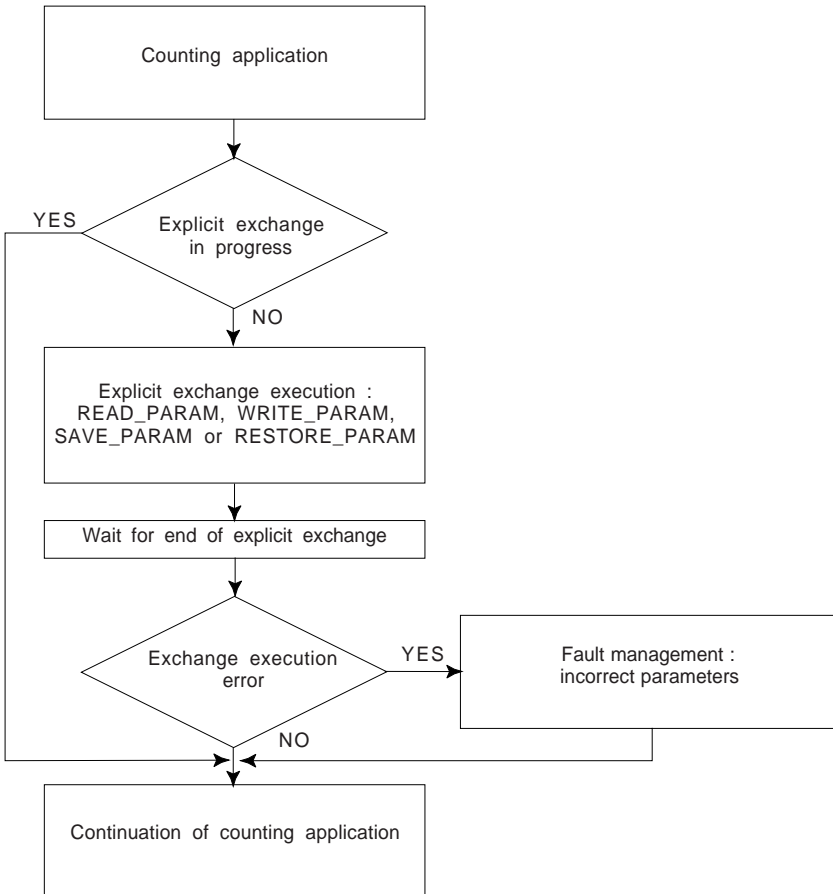
All variables associated with counting can be allocated a symbol.

Variables			
Parameters: I/O		Adr: 5: TSX CTY 4A	
<input type="checkbox"/> Entry field			
Address	Type	Symbol	Comment
%CH5.MOD	CH		
%I5.MOD.ERR	EBOOL		
%Mw5.MOD	\WORD		
%Mw5.MOD.1	\WORD		
%Mw5.MOD.2	\WORD		
%CH5.0	CH		
%Iw5.0.2	\WORD		
%Iw5.0.3	\WORD		
%ID5.0	D\WORD	Motor_aux2_cur_meas	Current counter measurement value
%ID5.0.4	D\WORD	Motor_aux2_capt	Captured counter value
%QV5.0	\WORD		
%QV5.0.1	\WORD		
%Kw5.0	\WORD		
%Kw5.0.1	\WORD		
%Kw5.0.2	\WORD		
%I5.0	EBOOL	Motor_aux2_enab_activ	Counter enable active
%I5.0.1	EBOOL	Motor_aux2_pres_done	Preset done
%I5.0.2	EBOOL	Motor_aux2_capt_done	Capture done
%I5.0.3	EBOOL		
%I5.0.4	EBOOL		

### 8.3 Explicit exchanges

Explicit exchanges are performed at the request of the user program. These exchanges are used to read and write the memory objects (%MW) associated with the application-specific functions. To check whether the explicit exchange was completed successfully, the corresponding status bits must be tested: exchange in progress and exchange execution error (%MWxy.i.0:X0 to X15 and %MWxy.i.1:X0 to X15).

The user must program his application as shown below if he uses the read/write adjustment parameter instructions : READ\_PARAM, WRITE\_PARAM, SAVE\_PARAM and RESTORE\_PARAM.



For more information, refer to the following documentation on explicit exchanges :

- Description of PL7 language,
- Common features of application-specific functions.

### 8.3-1 Reading the status word

The **READ\_STS** instruction is used to read the counter module or channel status word.

- To read the counter module status word (module diagnostics), use the following syntax :

**READ\_STS %CHxy.MOD**      xy : module position

READ\_STS %CH4.MOD : updates the contents of word %MW4.MOD.2.

- To read the counter channel status words (channel diagnostics), use the following syntax :

**READ\_STS %CHxy.i**      xy : module position, i: channel number

READ\_STS %CH4.1 : updates the contents of words %MW4.1.2 and %MW4.1.3.

### 8.3-2 Reading adjustment parameters

- To read the current value of the counter module adjustment parameters, use the following syntax :

**READ\_PARAM %CHxy.i**      xy : module position, i: channel number

READ\_PARAM %CH4.1 : updates the contents of the words associated with adjustment, ie. %MD4.1.4, %MD4.1.6, %MD4.1.8, %MD4.1.10, %MD4.1.12 and %MW4.1.14 to %MW4.1.21 and %MW4.1.27.

---

### 8.3-3 Writing adjustment parameters

- To modify the current value of the counter module adjustment parameters, use the following syntax :

**WRITE\_PARAM %CHxy.i**      xy : module position, i : channel number

WRITE\_PARAM %CH4.1 : sends the contents of the words associated with adjustment to the CTY channel, ie. words %MD4.1.4, %MD4.1.6, %MD4.1.8, %MD4.1.10, %MD4.1.12 and %MW4.1.14 to %MW4.1.21 and %MW4.1.27.

The indicator for execution of exchanges in progress is the object **adjustment in progress** : %MWxy.i.0:X2, and the indicator for an error during an exchange is the object **adjustment error** : %MWxy.i.1:X2.

If the parameter setting fails, the actual values used by the module must be re-read by READ\_PARAM. If the values are outside the range, they will be limited.

---

### 8.3-4 Saving adjustment parameters

- To save the adjustment parameters of a counter module in the application, use the following syntax :

**SAVE\_PARAM %CHxy.i**      xy : module position, i : channel number

SAVE\_PARAM %CH4.1 : saves the current parameters in the PLC application (replaces the initial values with the current values). This affects objects %MD4.1.4, %MD4.1.6, %MD4.1.8, %MD4.1.10, %MD4.1.12 and %MW4.1.14 to %MW4.1.21 and %MW4.1.27.

The indicator for execution of exchanges in progress is the object **adjustment in progress** : %MWxy.i.0:X2, and the indicator for an error during an exchange is the object **adjustment error** : %MWxy.i.1:X2.

---

### 8.3-5 Restoring adjustment parameters

- To restore the adjustment parameters of a counter module to the application, use the following syntax :

**RESTORE\_PARAM %CHxy.i**      xy : module position, i : channel number

RESTORE\_PARAM %CH4.1 : replaces the current parameters in the PLC application with the initial values. This affects the following objects : %MD4.1.4, %MD4.1.6, %MD4.1.8, %MD4.1.10, %MD4.1.12 and %MW4.1.14 to %MW4.1.21 and %MW4.1.27.

The indicator for execution of exchanges in progress is the object **adjustment in progress** : %MWxy.i.0:X2, and the indicator for an error during an exchange is the object **adjustment error** : %MWxy.i.1:X2.

---



---

### 8.3-6 Execution conditions

- An adjustment operation launched while another adjustment operation is already in progress (%Mwxy.i.0:X2 to X1) is ignored.
- A reconfiguration operation cannot be launched during an adjustment operation.

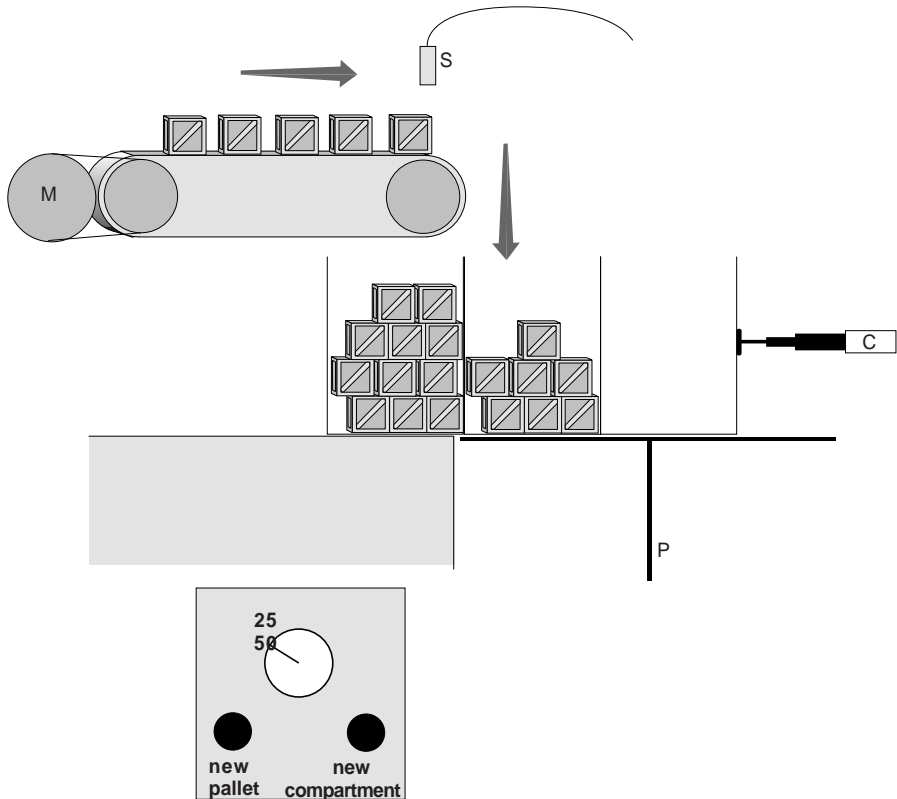
Status words (STATUS) are only updated when a READ\_STS instruction is launched or if the debug screen is open.

The adjustment %MD and %MW only correspond to the parameters used by the module when an explicit exchange has been carried out and has not ended in an error (%MWxy.i.1:X2 to X0).



### 9.1 External specifications of the example

The machine to be controlled groups batches of objects into a pallet with three compartments each containing 25 or 50 objects. The principle is as follows :



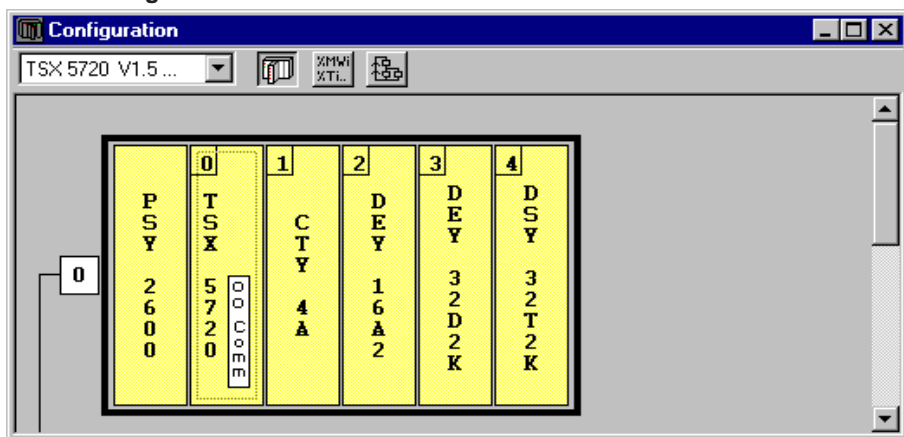
- a motor M drives a conveyor belt on which the objects are carried,
- a sensor S counts the objects before they are placed in the pallet,
- a telescopic cylinder C moves the pallet in order to present a new compartment when the current one is full,
- a platform P is used to change the pallet.

The man-machine interface consists of the following checks :

- new pallet : forces the pallet to change (on a rising edge),
- 25/50 : used to select the type of compartment. Only taken into account for the next pallet (on a state),
- new compartment : forces the compartment to change (on a rising edge),
- starting the motor physically enables the counter.

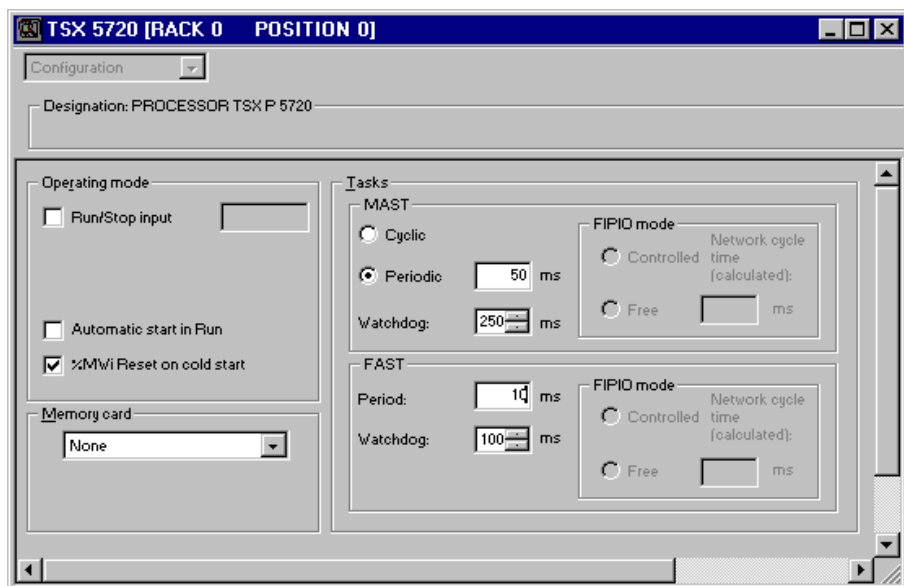
## 9.2 Internal specifications of the application

### 9.2-1 Configuration



### 9.2-2 Processor

TSX 57-20, periodic MAST, 50ms



### 9.2-3 Counting

The downcounting function is used in automatic preset on a CTY.  
Channel configured for the MAST task.

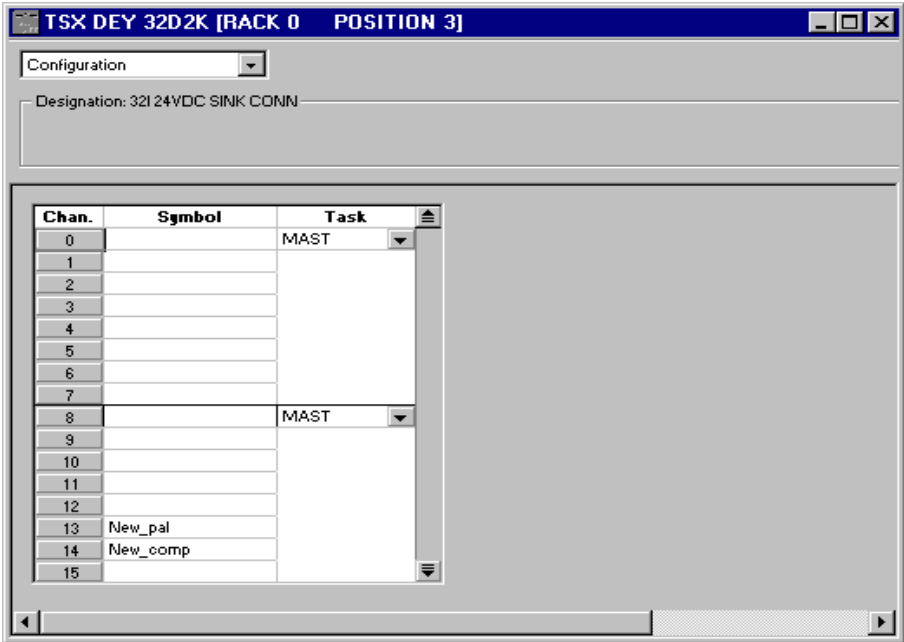
Use of event 0.

The screenshot shows the configuration window for a TSX CTY 4A module. The window title is "TSX CTY 4A [RACK 0 POSITION 1]". The configuration is as follows:

- Configuration:** Designation: 4 CH. COUNTER MOD 40KHZ
- Symbol:**
  - Counter: Counter 0
  - Function: Downcounting
  - Task: MAST
- Input interfaces:**
  - 1 IA input
  - Solid State contact
- Preset on IPres:** Rising edge IPres
- Operates on changing to 0:**
  - No downcounter preset
  - With downcounter preset
- Event:**
  - EVT 0
- Reset output Q0:**
  - Manual
  - Automatic
- Fallback mode:**
  - Reset
  - Maintain

## 9.2-4 Discrete I/O

An input module and an output module configured for the MAST task. Only the input module is used in the program.



## 9.2-5 Internal bits and words

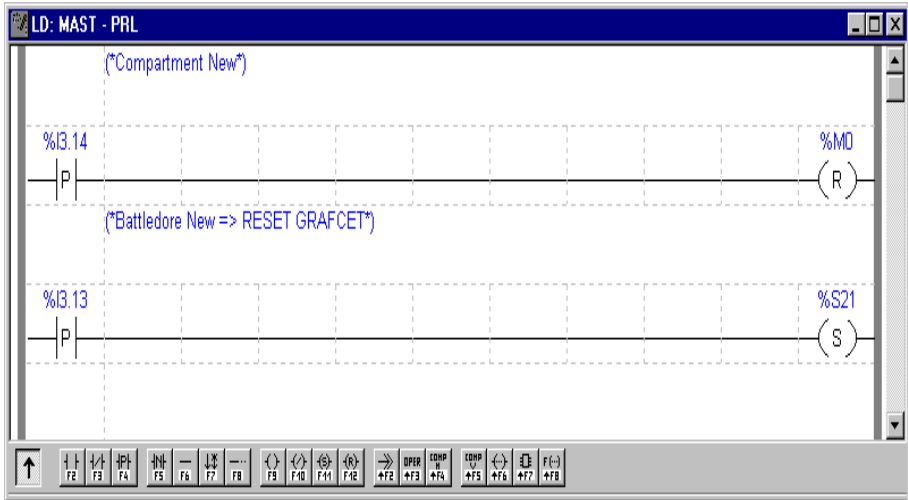
- %M0 at 1 : starts the motor M  
at 0 : stops the motor M
- %M1 at 1 : the pallet is being changed,  
at 0 : the pallet is ready.
- %MW0 : position of the cylinder C (1, 2 or 3, 4 = exit of pallet),
- %MW1 : memorization of the compartment size

## 9.3 Program

### 9.3-1 Preprocessing

Management of the forced operating modes :

- change of pallet
- change of compartment

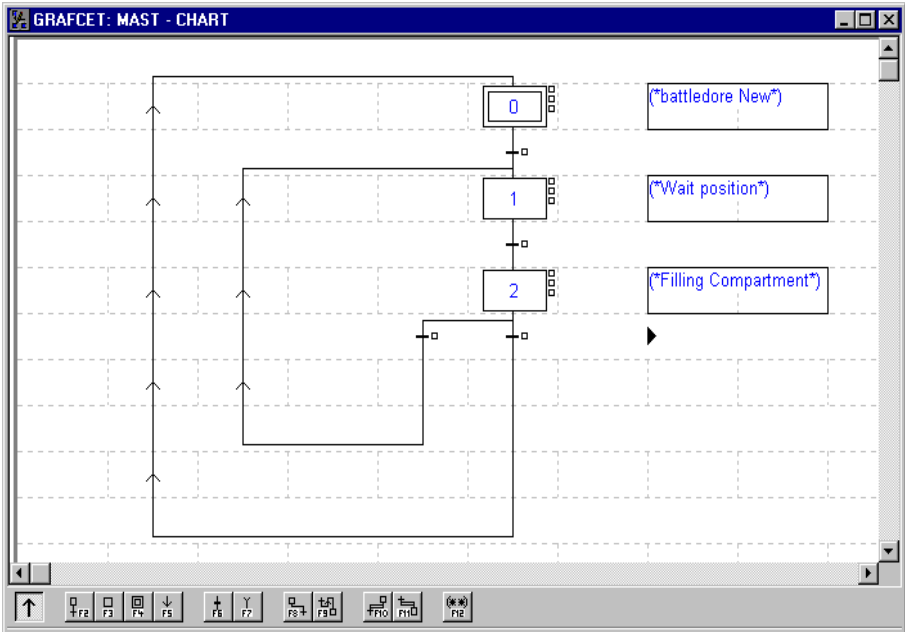


### 9.3-2 Sequential processing

Step 0 : Initializations

- stopping the motor,
- enabling the auxiliary counter inputs,
- unmasking the threshold crossing event.

Step 2 : filling the compartment

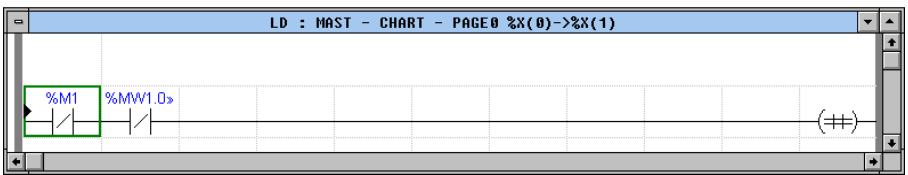




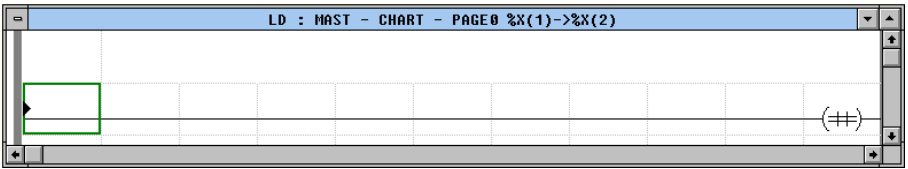
### 9.3-3 Grafcet actions and transitions

Grafcet transitions :

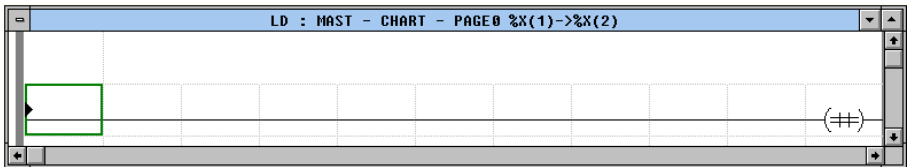
- Transition between action 0 and action 1 :



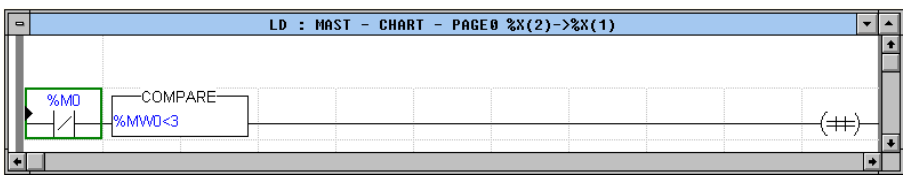
- Transition between action 1 and action 2 :



- Transition between action 2 and action 0 :

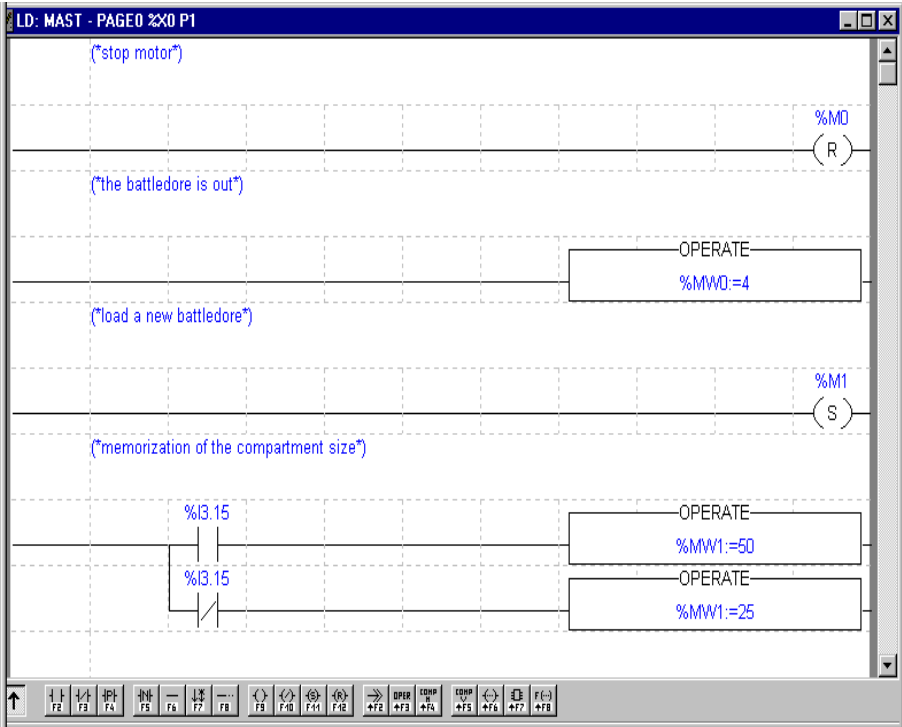


- Transition between action 2 and action 1 :

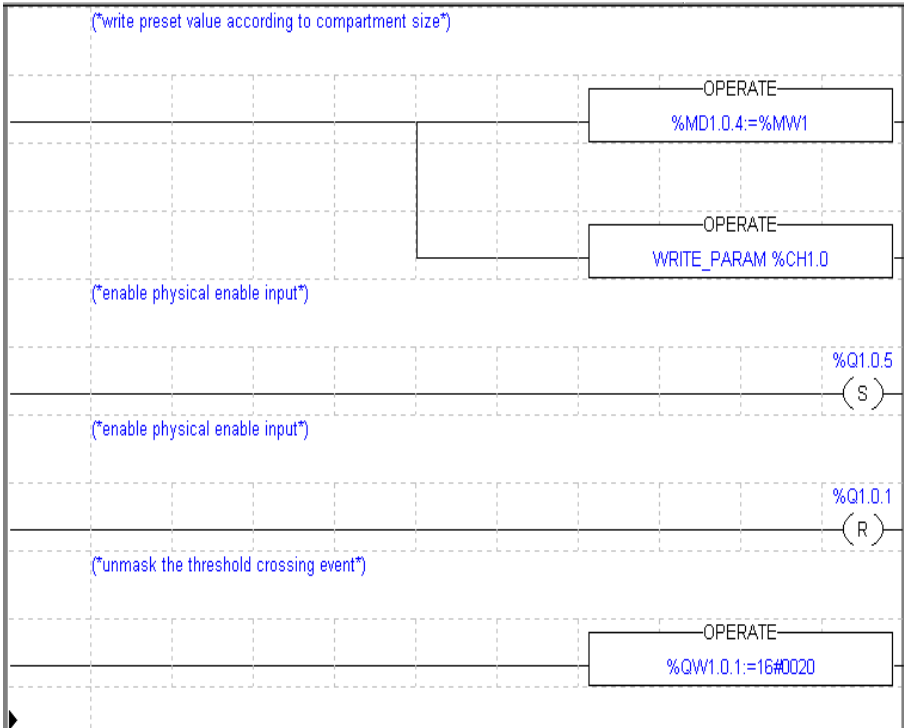


## Grafcet steps :

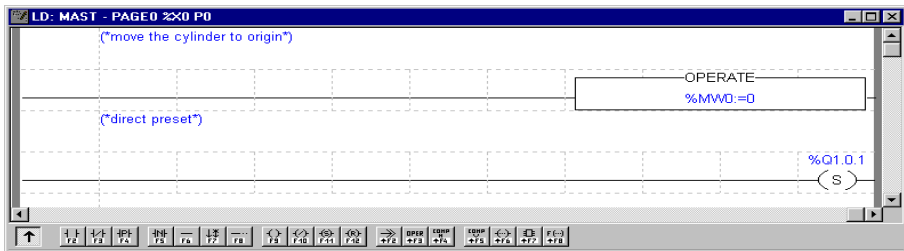
- Step 0 : (action on activation P1)



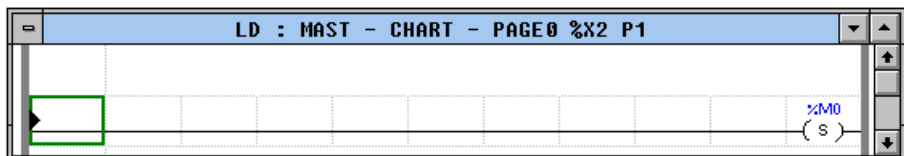
• Step 0 : (action on activation P1, continued)



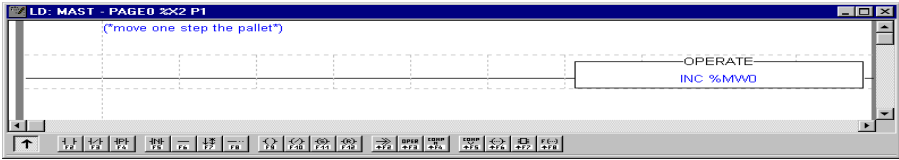
• Step 0 : (action on deactivation P0)



• Step 2 : (action on activation P1)

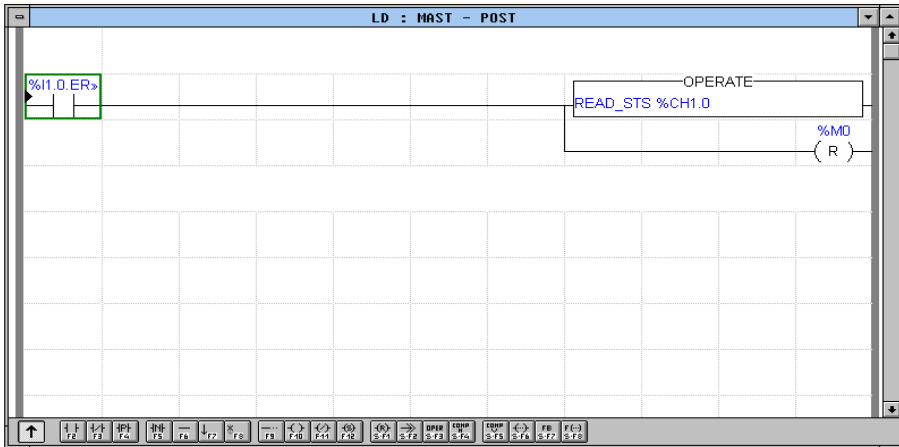


## • Step 2 : (action on activation P0)



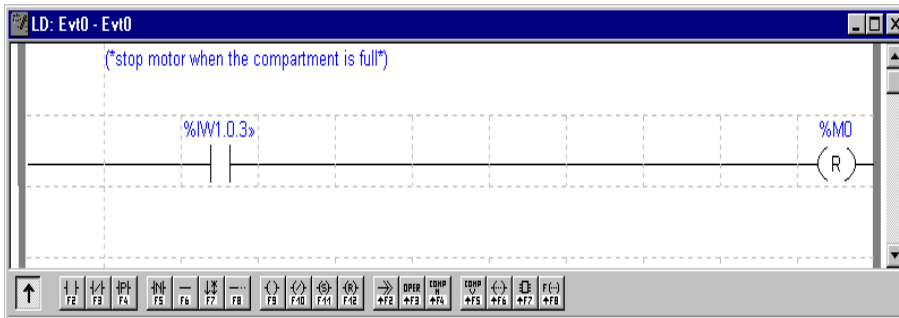
## 9.4 Post-processing

Error management and stopping the motor in the event of a problem.



## 9.5 Event processing

Signaling of the crossing of value 0.



## 10.1 Time performance levels

The table below shows the main time performance levels for TSX CTY 2A/4A/2C counter modules.

	TSX CTY 2A TSX CTY 4A	TSX CTY 2C
Frequency of counter inputs IA, IB :		
• counting pulses	40 kHz	1 MHz
• incremental encoder	40 kHz	500 kHz x1 250kHz x4
Immunity of counter inputs IA, IB or IZ in the case of mechanical contacts	1.6 ms	1.6 ms
Response time of event processing :		
• taking account of an event-triggered input	on DSY32T2K 3.2 ms on DSY8T22 2.1 ms	2.4 ms /
• setting an output.		
• Response time of physical output	1 ms	200µs
• Module cycle time	5 ms (CTY 2A) 10 ms (CTY 4A)	1 ms
• Impact of a channel on the CPU scan time	80µs	86µs
	Impact on scan time	Time period
READ_STS	370 ms	200 ms
WRITE_PARAM	810 ms	+ task
READ_PARAM	380 ms	period

The table below shows the permissible separation between two consecutive thresholds as a function of the counter input frequency.

Conforming with these conditions ensures that threshold crossing is taken into account.

FREQUENCY	TSX CTY 2A / 4A	TSX CTY 2C
500 Hz	1	1
1 kHz	2	1
10 kHz	13	1
40 kHz	50	1
250 kHz		1
500 kHz		1
1 MHz		1

### Consecutive edges of the read input (TSX CTY 2C)

The table below shows the permissible gap between two consecutive edges of the read input, used for example to measure the length of parts (configuration mode : capture on rising and falling edges of the physical read input), as a function of the frequency of the counter inputs. As long as these conditions are observed, the function for measuring the length of parts should operate correctly, ie. all edges present on the physical read input taken into account :  $\text{Separation} \geq 0.5 \text{ (ms)} \times \text{Frequency (kHz)}$

Frequency of the counter inputs	Minimum separation between two consecutive edges of the read input IRead
125 kHz	63
250 kHz	125
500 kHz	250
1 MHz	500

### Maximum counting frequency with multiplication by 1 and 4

When an incremental encoder is connected to a TSX CTY 2A/4A/2C module, it is possible to select operation with multiplication by 1 or 4 during configuration. This table shows the maximum frequencies on the counter inputs depending on the configuration :

Module	Maximum frequency with multiplication by 1	Maximum frequency with multiplication by 4
TSX CTY 2A / 4A	40 kHz	40 kHz
TSX CTY 2C	250 kHz	125 kHz
	500 kHz	250 kHz

## 10.2 Questions/Answers

Symptom	Diagnostics	Procedure to follow
<b>ADJUSTMENT</b>		
The counter channel does not appear to have taken account of the thresholds written by WRITE_PARAM.	Program a READ_PARAM in your application to determine the values actually used by the channel. A WRITE_PARAM triggered while another adjustment exchange is in progress is ignored.	Test bit %MWxy.i.0:X2 (adjustment in progress) before any adjustment exchange.
<b>ADJUSTMENT</b>		
My adjustments have been lost.	A cold start or a reconfiguration of the counter channel in online mode causes the loss of the current settings made via the screen or a WRITE_PARAM.	Save the current settings using the "Save Parameters" function or via the SAVE_PARAM instruction.
<b>EVENTS</b>		
Event processing associated with the counter channel is not executed.	Check the event feedback circuit is enabled. <ul style="list-style-type: none"> <li>• event number declared during configuration is identical to that of event processing,</li> <li>• source of the unmasked event (%QWxy.i.3),</li> <li>• events authorized at system level (%S38 = 1),</li> <li>• unmasked events at system level (UNMASKEVT()).</li> </ul>	See section 6, use of events.
<b>EXECUTION</b>		
My direct capture/preset is not taken into account.	<ul style="list-style-type: none"> <li>• immediately after a restart or a change to RUN : the first %Q are not taken into account by the counter channel.</li> </ul>	On a restart, wait for an additional cycle before performing the preset or the capture.
	<ul style="list-style-type: none"> <li>• in normal operation : the module cannot see the rising edge of the capture/preset</li> </ul>	Maintain the corresponding %Q for at least one module cycle in each state, so that the edge is visible.
<b>DIAGNOSTICS</b>		
The status words %MWxy.i.1 and 2 are not consistent with the state of my counter channel.	These words are only updated on explicit READ_STS request or if the debug screen is open.	Program a READ_STS in your application.

---

**Questions/Answers table (continued) :**

Symptom	Diagnostics	Procedure to follow
<b>DIAGNOSTICS</b>		
The "encoder power supply" fault persists even though my encoder is correctly supplied and the current value is changing.	The encoder power supply return signal (EPSR) is not wired correctly.	See setup manual for the different ways of wiring this signal.
<b>DEBUG</b>		
The commands in the debug screen have no effect.	The application or the task is in STOP.	Set the application or the task to RUN.
<b>DEBUG</b>		
Some commands in the counter debug screen cannot be modified.	%Qxy.i.r or %QWxy.i.r:Xj corresponding to these commands are written by the application.	Use bit forcing (for %Qxy.i.r type objects) or design an application which does not automatically write these command bits (modification on a transition and not on a state).

---



## 11 TSX CTY 2C special functions

## 11.1 Presentation

The TSX CTY 2C counter module is used to configure special functions. PL7 by program version V3.0 or later is required in order to use these special counting functions.

## 11.2 Description

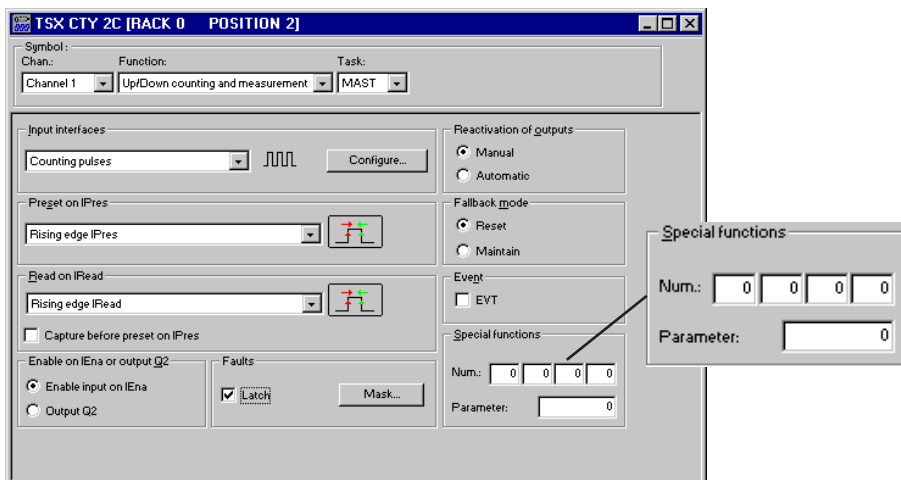
The special functions correspond to specific requirements in certain counting applications. They do not appear in the standard up/down counting and measurement application-specific functions.

The TSX CTY 2C counter module accepts the use of four simultaneous, non-exclusive special functions.

The parameters linked to the special functions are determined in the configuration of the counter module. The area of the screen entitled **Special functions** is specifically for entering the configuration parameters.

## 11.2-1 No special function

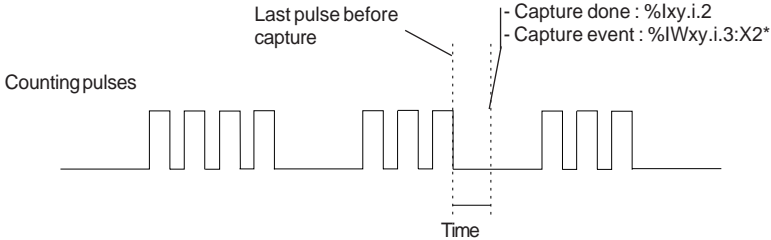
The fields for entering the special function values are initially configured at zero. These are four numeric fields identified by the **Num** item, and the **Parameter** field. If no special function has been configured, the TSX CTY 2C module performs the standard processing operations of the "up/down counting and measurement" function.



## 11.2-2 Special function number 1 : Time elapsed since last pulse

Special function no. 1 is used to perform a specific time calculation on the associated counter channel. This calculation represents the time elapsed between the last up/down counting pulse and the occurrence of a capture.

The time separating the last pulse from the capture is expressed in milliseconds with an accuracy of  $\pm 1$  ms.



\* only if the up/down counter channel is configured for event processing.

### Configuring special function no. 1

Special function no. 1 is declared by entering the value 1 in one of the four fields labeled **Num** in the **Special functions** function block in the counter module configuration screen.

The **Parameter** field is not used for this function and retains its default value (0).

### Debugging special function no. 1

Two language objects are associated with special function no. 1, containing time calculation information **%IDxy.i.11** and **%IDxy.i.6**.

This numeric information can only be accessed via PL7 animation tables.

### Special function no. 1 language objects

- **%IDxy.i.11** : Elapsed time (in Mast or Fast task) at the capture done corresponding to bit **%Ixy.i.2**
- **%IDxy.i.6** : Elapsed time (in event-triggered task) at a capture event corresponding to word **%IWxy.i.3:X2**

### 11.2-3 Special function number 2 : Internal capture and preset

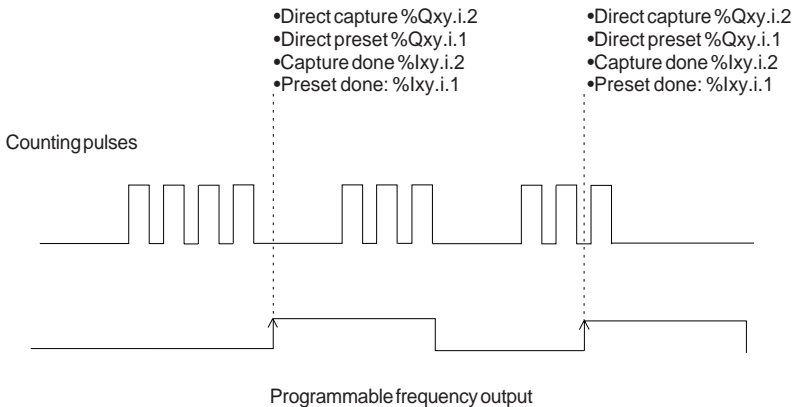
Special function number 2 triggers, on the counter channel, a direct capture (by program) and a direct preset (by program) of the up/down counter synchronized on the programmable frequency output (this output is not wired).

Each rising edge on the programmable frequency output on the counter channel triggers :

- A direct capture (by program) of the value of the counter followed by a direct preset (by program)

On the counter channel :

- The capture done bit `%Ixy.i.2` is then set to 1 (in the standard functions, a direct capture by program does not set the "capture done" bit to 1).
- The preset done bit `%Ixy.i.1` is then set to 1 (in the standard functions, a direct preset by program does not set the "preset done" bit to 1).



#### Configuring special function no. 2

Special function no. 2 is accessed by entering the value 2 in one of the four fields labeled **Num** in the **Special functions** function block in the counter module configuration screen.

The **Parameter** field is not used for this function and retains its default value (0).

#### Debugging special function no. 2

Special function number 2 has no specific language objects.

The capture done bit `%Ixy.i.2` and the preset done bit `%Ixy.i.1` can be accessed from the standard debugging screen.

---

### Special function no. 2 language objects:

Special function number 2 has no specific language objects. It simply makes the capture done %Ixy.i.2 and preset done %Ixy.i.1 bits operate in a specific way.

- %Ixy.i.2 : "Capture done"
  - **At state 1** : when the capture has been done; direct capture (by program) or capture via the physical enable input. In the standard functions, direct capture (by program) does not set the "preset done" bit to 1.
  - **At state 0** : on a rising or falling edge of the capture reset.
- %Ixy.i.1: "Preset done"
  - **At state 1** : when the preset has been done; direct preset (by program) or preset via the physical preset input. In the standard functions, direct preset (by program) does not set the preset done bit to 1.
  - **At state 0** : on a rising or falling edge of the preset reset.

---

### 11.2-4 Special function number 3 : Correct speed and moving part stationary

Special function number 3 performs the following for the counter channel :

- A "Correct speed" check which corresponds to the following Boolean function :

Correct speed := (Target speed - X% ≤ Speed measurement)  
AND (Speed measurement ≥ Target speed + X%)

- "Moving part stationary" detection (moving part immobile) which corresponds to the following Boolean function :

Moving part stationary := (Speed measurement ≤ Stationary speed)

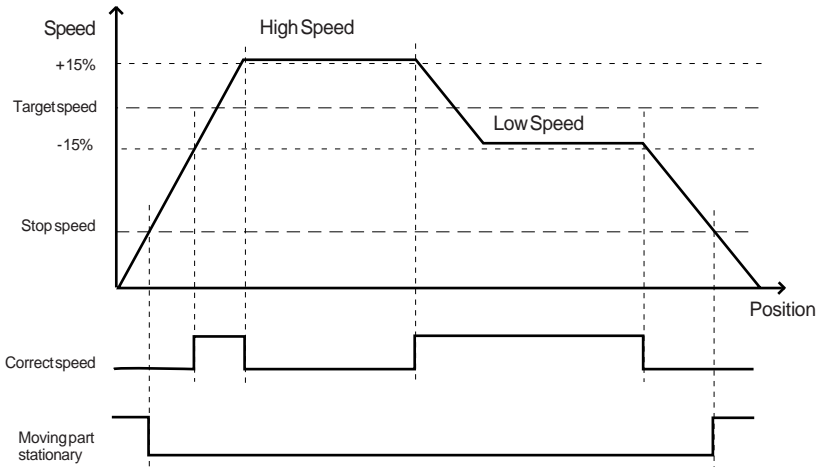
The tolerance on the "X%" speed is a configuration parameter entered by the user. The "Target speed" and the "Stationary speed" are adjustment parameters worked out by the user.

The speed measurement (%IDxy.i.8) is provided by the up/down counting and measurement standard function of the TSX CTY 2C module.

**Notes :**

The "Correct speed" data is significant when it is used on speed plateaux (constant or relatively stable speed, no acceleration or deceleration phases). When starting or stopping, the target speed is variable over a reduced time scale. Management of the relevance of the "Correct speed" data is thus left to the application program.

The "Moving part stationary" data is significant when it is used on speed plateaux (constant or relatively stable speed, no acceleration or deceleration phases). When starting or stopping, there are compulsory "Moving part stationary" phases, that is, phases when the speed is below the stop speed. Management of the relevance of the "Moving part stationary" data is thus left to the application program.

**Configuration of special function no. 3**

To access special function no. 3 :

- One of the four **Num** fields in the **Special functions** function block in the configuration screen must be configured with the value 3.
- The **Parameter** field is used for configuring the tolerance on the speed X%. The parameter values are between 1 and 100%. The default value is 0.

---

### Adjust mode

The numeric data of the "Correct speed" and "Moving part stationary" language objects associated with special function number 3 must be entered, so that the special function will provide significant data.

These adjustment language objects can be entered in the following way :

- By explicit writing in the application program

Example using Structured Text programming :

```
%MDxy.i.24 := Target speed (*pulses per second*);
```

```
%MDxy.i.26 := stop speed (*pulses per second*);
```

```
WRITE_PARAM %CHxy.i;
```

- Using a variables animation table in online mode (see PL7 Junior software Operating Modes manual)

### Debugging special function no. 3

In debug mode, the language objects of special function number 3, correct speed **%lxy.i.16** and Moving part stationary **%lxy.i.17** can only be accessed via an animation table.

### Special function no. 3 language objects:

- **%MDxy.i.24** : Target speed value - Word which can be read, written and tested.
- **%MWxy.i.26** : Stop speed value - Single word which can be read, written and tested. This word varies from 0 to 32000.
- **%lxy.i.16**: "Correct speed"
  - **At state 1** : the speed is correct - the speed is between the target speed - X% and the target speed + X% ( $\text{Target speed} - X\% \leq \text{correct speed} \leq \text{target speed} + X\%$ ).
  - **At state 0** : the speed is incorrect, strictly lower than the target speed - X%, strictly higher than the target speed + X%.
- **%lxy.i.17**: "Moving part stationary"
  - **At state 1** : the speed is lower than or equal to the stop speed.
  - **At state 0** : the speed is strictly higher than the stop speed.

### 11.3 Compatibility of the special functions

The version of the counter module is shown :

- on the product reference label on the right-hand side of the module
- in the debug screen in online mode in the module field

Specialfunctionnumber	Moduleversion
Function no. 1	V1.0
Function no. 2	V1.0
Function no. 3	V1.1

Any attempt to configure a counter module with a special function which is not supported causes an application fault.

### 11.4 Exclusivity of the special functions

Two special functions are exclusive as soon as there is an overlap between their specific language objects.

Non-exclusive special functions can be configured simultaneously up to a limit of 4 simultaneous special functions.

	Function no. 1	Function no. 2	Function no. 3
Function no. 1		Not exclusive (1)	Exclusive
Function no. 2	Not exclusive (1)		Exclusive
Function no. 3	Exclusive	Exclusive	

- (1) If special functions 1 and 2 are used simultaneously, the time calculation of special function number 1 %IDxy.i.11 can only be accessed in a Fast or Mast task, when capture done %lxy.i.2 = 1.





**A**

ABE-7CFA11 4/10, 5/13  
 Address of parallel output absolute encoder 5/21  
 Addressing objects 8/14  
 Adjustment 5/16  
 Automatic mode 2/15, 3/17, 4/24, 4/28

**C**

Capture 3/11, 4/19, 5/9, 5/15, 5/25, 5/27  
 Capture done 5/21  
 Capture value 5/21  
 Cold start 7/1, 7/2  
 Comparison 2/10, 3/12, 4/21, 5/25, 5/28  
 Configuration 5/4  
 Counter outputs 2/11, 3/13, 4/22, 5/18, 5/25  
 5/28  
 Counting direction 5/21  
 Counting pulses 5/11  
 Current value 5/20, 5/25

**D**

Debug 5/19, 5/22  
 Diagnostics 5/29, 10/3  
 Downcounting function 2/2, 5/7

**E**

Enable 2/5, 3/7, 4/13, 5/15, 5/24, 5/27  
 Enable active 5/21  
 Encoder data bits 4/9, 4/10  
 Event 5/10, 5/15  
 Event processing 2/18, 3/20, 4/31, 6/1  
 Explicit exchange 8/3

**F**

Fallback mode 5/10, 5/15  
 Fault 5/30  
 Fault acknowledgment 4/12, 5/1, 5/27, 5/33  
 Faults 4/28, 5/15  
 Filtering 5/11  
 Forcing 5/22  
 Frequency output 5/17

**H**

Header bits 4/9, 4/10  
 High setpoint 2/10

**I**

Implicit exchange 8/2  
 Incremental encoder 3/4, 4/7, 5/8, 5/11  
 Input interface 3/3, 4/6, 5/8, 5/11  
 Invalid 5/30  
 Invalid value 2/4, 3/6, 4/12, 5/21, 5/27, 5/33

**L**

Latching faults 4/15, 5/34  
 Levels of priority 2/12, 3/14, 4/23  
 Line check 3/5, 4/8, 4/9, 4/11  
 Long cam 3/10, 4/18

**M**

Maintain fallback mode 2/15, 3/17, 4/27  
 Manual mode 2/15, 3/17, 4/24, 4/28  
 Masked least significant bits 4/9, 4/10  
 Masked most significant bits 4/9, 4/10  
 Masking faults 5/34  
 Measurement 5/27, 5/30  
 Measurement inversion 4/8, 4/9, 4/11  
 Measurement period 4/14, 5/18  
 Mechanical contact 4/7  
 Modulo 4/9, 4/11, 4/21  
 Modulo mode 4/7  
 Multiplexed parallel output absolute encoders 4/25, 5/23  
 Multiplexing of parallel output absolute encoders 5/14  
 Multiplication by 4 3/5, 4/8

**O**

Offset 4/9, 4/11, 5/17, 5/21  
 Online mode 5/19, 7/2  
 Outputs 4/24, 5/26, 5/28  
 Overspeed fault 4/15  
 Overspeed threshold 4/1, 4/14, 5/18

**P**

Parallel output absolute encoder 4/10  
 Parallel output absolute sensor 5/13  
 Parity 4/9, 4/11  
 Performance 4/19, 10/1  
 Phase-shifted signals 3/4, 4/7

Physical outputs	2/14, 3/16	Special functions	5/15
Preset	2/4, 2/7, 3/6, 3/8, 4/12, 4/16, 5/1	Specific encoder fault	4/9, 4/11
	5/9, 5/15, 5/24, 5/27	Speed	5/21, 5/27
Preset done	5/21	Speed monitoring	4/1, 4/14, 5/18
Preset value	5/17, 5/21	SSI absolute encoder	5/12
Presymbolization	5/2	SSI serial output absolute encoder	4/9
Programmable frequency	4/24	SSI transmission frequency	4/9, 4/10
Protection against overloads	2/16, 3/18, 4/29	Status bits	4/9, 4/11
Protection against short-circuits	2/16, 3/18, 4/29	STOP mode	7/2

## R

Reactivation of outputs	5/10, 5/15
Reactivation of the physical outputs	3/18, 4/29
READ_PARAM	8/17
READ_STS	8/17
Reduction of the resolution	4/9
Reset	2/7, 5/1, 5/9, 5/24
RESET conditions	2/11, 3/13, 4/22
Reset fallback mode	2/15, 3/17, 4/27
Resetting outputs to 0	4/27
RESTORE_PARAM	8/18

## S

Safety	2/16, 3/18, 4/27, 4/29
SAVE_PARAM	8/18
Separate signals	4/6
SET conditions	2/11, 3/13, 4/22
Setpoints	3/12, 5/17
Short cam	3/9, 4/17
Solid state contact	4/7, 4/8

## T

Thresholds	2/10, 3/12, 4/21, 5/17
------------	------------------------

## U

Unmasking events	2/18, 5/28, 6/3
Up/down counting	3/1
Up/down counting and measurement	4/2
Up/down counting and measurement function	5/10
Up/down counting function	5/8
Upcounting function	2/3, 5/7

## W

Warm restart	7/1
WRITE_PARAM	8/18

## Z

Zero marker	3/4, 3/9, 3/10, 4/7, 4/17, 4/18
-------------	---------------------------------

---

**Preface**

---

**General introduction**

**B0**

---

**Independent axes**

**B1**

---

**Interpolation**

**B2**

---

**Index**

---



---

## Introduction to part B

Part B, which describes servomotor axis control, has 3 subsections :

---

### Part

**B0** is common to all 5 axis control modules.

**B1** presents the independent axis function for TSX CAY modules.

The differences between TSX CAY modules are shown by an icon. If there is no icon, the page or section applies to all 5 modules.

Functions specific to TSX CAY 21/41 modules are identified by the following icon.



Functions specific to TSX CAY 22/42 modules are identified by the following icon.



Functions specific to TSX CAY 22/42 and 33 modules are identified by the following icon.




---

**B2** describes the TSX CAY 33 module used to control interpolated axes. The presentation icon only appears on the first page.



The names of the various parts are :

- **B0** : General introduction
- **B1** : Independent axes
- **B2** : Interpolation

---

## Prerequisites

Part B describes axis control and assumes that the user is familiar with the functions common to all TSX Premium modules, described in the section on "Common application-specific functions", in Volume 1 :

- Exchange of data between the processor and the modules
- Presymbolization of language objects

## Terms used

In the rest of the document, **TSX CAY** signifies all the axis control modules :

- **TSX CAY • 1** signifies TSX CAY 21 and 41 modules
- **TSX CAY • 2** signifies TSX CAY 22 and 42 modules
- **TSX CAY 3 •** signifies TSX CAY 33 modules
- **TSX CAY 4 •** signifies TSX CAY 41 and 42 modules

The term **•move** is a generic term used to signify SMOVE instructions for an independent axis and XMOVE instructions for interpolated axes.

The term **TSX Premium** is a generic term used to signify all PLCs (TSX 57, PMX 57, PCX 57) or processors (TSX P 57, TPMX P57, TPCX 57) in the Premium range.

## Compatibility between TSX CAY modules, processors and PL7

A PLC (and therefore a processor), one or more TSX CAY modules and PL7 Pro or PL7 Junior programming software are required in order to set up a motion control application.

All of these devices are available in various versions (sold individually or as part of the installed base), corresponding to developments linked to the addition of new functions. Not all combinations of hardware and software are possible as there are **incompatibilities** or **restrictions** linked to certain configurations (see the tables below):

### Compatibility between processors and TSX CAY modules

Processors	TSX P57 10 / 20 (V 1.5 / V 1.6)	TSX P 57 ••2 TPMX P57 ••2 TPCX 57 ••12 (V 3.0)	TSX P 57 ••2 TPMX P57 ••2 TPCX 57 ••12 (V3.3)
<b>TSX CAY</b>			
TSX CAY 21 / 41 (V1.0)	Yes	Yes	Yes
TSX CAY 21 / 41 (V1.5)	Yes	Yes	Yes
TSX CAY 21 / 41 (V1.9)	Yes	Yes	Yes
TSX CAY 22 / 42 / 33	No	Yes	Yes

---

**Compatibility between processors and PL7 software**

<b>Processors</b>	TSX P57 10 / 20 (V 1.5 / V 1.6)	TSX P 57 **2 TPMX P57 **2 TPCX 57 **12 (V 3.0)	TSX P 57 **2 TPMX P57 **2 TPCX 57 **12 (V3.3)
<b>PL7</b>			
PL7 Junior V1.5	Yes	No	No
PL7 Junior V1.7	Yes	No	No
PL7 Junior / Pro V3.0	Yes	Yes	Yes (*)
PL7 Junior / Pro V3.1	Yes	Yes	Yes (*)
PL7 Junior / Pro V3.3	Yes	Yes	Yes

(\*) Compatibility with certain restrictions (described in the rest of the document)

**Compatibility between TSX CAY modules and PL7 software**

<b>TSX CAY modules</b>	TSX CAY 21 TSX CAY 41 (V 1.0)	TSX CAY 21 TSX CAY 41 (V 1.5)	TSX CAY 21 TSX CAY 41 (V 1.9)	TSX CAY 22 TSX CAY 42 TSX CAY 33
PL7 Junior V1.5	Yes	Yes (*)	No	No
PL7 Junior V1.7	Yes	Yes	Yes (*)	No
PL7 Junior / Pro V3.0	Yes	Yes	Yes	No
PL7 Junior / Pro V3.1	Yes	Yes	Yes	No
PL7 Junior / Pro V3.3	Yes	Yes	Yes	Yes

(\*) Compatibility with certain restrictions (described in the rest of the document)

---

## Glossary

**Absolute encoder** : Encoder whose range of measurement is divided into a finite number of equal elementary spaces which are each given a single coded marker.

**Axis** : Motor/speed drive/mechanism combination which controls the movement of a moving part in a given direction (axis, linear movement) or around a fixed rotation axis (rotating axis, circular movement).

**Axis referenced** : Module status once the reference point has been set. Position measurements are only significant and movements are only permitted in this state.

**Deviation** : Difference between the position reference and the measurement during a movement.

**Direction discriminator** : A microprogrammed system which determines the direction of movement of the moving part (fixed by the adjustment parameter SL\_MIN).

**Emergency stop** : Movement stop with maximum deceleration.

**Event** : A change of state of the event input or the EXT\_EVT bit which is accessed by the program.

**Feedforward gain (KV)** : A coefficient used to adjust the anticipation of the position control loop speed (a compromise between the deviation and the stop point overshoot).

**Forced reference point** : Procedure for loading the current position measurement at a preset value. This operation references the axis.

**Gray code** : Binary code which is said to be reflected, in which term  $n$  is changed to term  $n+1$ , changing only one digit. The code can thus be read without any ambiguity.

**Incremental encoder** : Pulse generator with two 90° phase-shifted signals.

**Independent multi-axis** : The movement rule is applied independently to each axis. The axes start at the same time, and move at a reference speed, the duration of the movement depends on the distance to be travelled. The "axes" do not arrive at the same time. The movement in space can be of any type. The objective is to reach the target marker without the restriction of a trajectory.

**Infinite machine** : the moving part moves continuously between the value 0 and the modulo limit (for example, a conveyor belt).

**Interpolation** : used to link 2 or 3 axes together, to perform movements in a plane or in space.

**ISO** : International Standards Organization. The ISO code is widely used. Formats, symbols and transmission rules are the subject of the ISO standards. AFNOR is a member of the ISO.

**Limited machine** : the moving part moves between two upper and lower limits, in two directions.



---

**Lower soft stop** : Lower position measurement which the moving part must not pass below (set by the command parameter SL\_MIN).

**Machine reference point** : Machine axis measurement reference point.

**Mechanical cam** : Mechanical projection on an axis which actuates a limit switch when the moving part passes over it.

**Modulo** : measurement evolution range for an infinite axis.

**Movement profile** : This is the profile of variation in position, speed and acceleration references. It is often illustrated by the curve : speed = F (time).  
In an increasing order of complexity there are : rectangular, triangular, trapezoid, parabolic and sine squared profiles.

**Parametered indexed position (PREF)** : Index value for calculating indexed positions.  
Absolute position = index (PREF) + indexed position

**Reference point setting** : Procedure for loading the current position measurement by moving the moving part and detection of the external event (reference point input and/or cam input). This operation references the axis without causing any movement.

**Resolution** : Smallest variation of input data which provides measurable information on the output data.

**Servo control** : Control system function which consists of making a physical measurement conform to a fixed or variable reference (position control, speed control, etc).

**Speed correction coefficient (CMV)** : A coefficient which multiplies all speeds by a value of 0 to 2 in steps of 1/1000.

**Speed reference** : Theoretical speed of the moving part calculated by the module using the maximum acceleration profile and the programmed speed.

**Target window** : Positioning tolerance around the stop point.

**Trajectory** : Series of elementary movements which pass through intermediate markers between a start marker and a target marker. The movement between the two markers is executed at an appropriate speed or in an appropriate time.

**Upper soft stop** : Upper position measurement which the moving part must not exceed (set by the command parameter SL\_MAX).

**Valid measurement area** : Set of measurement points between the two soft stops.

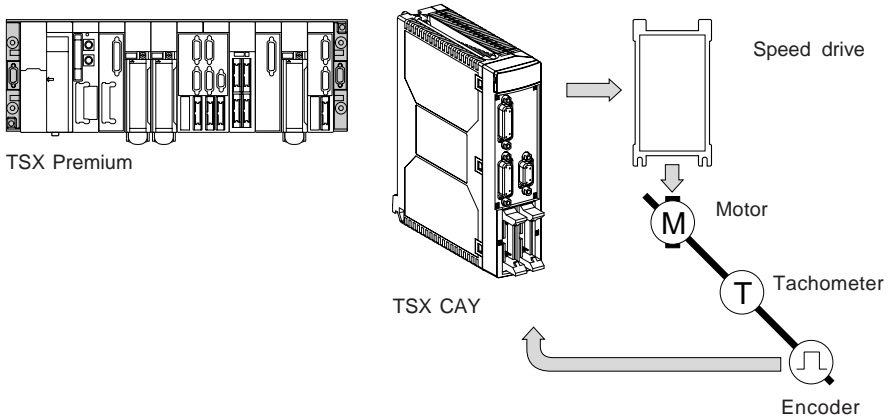
**Zero marker** : Pulse supplied by a rotary incremental encoder. It is detected on each complete axis revolution.



<b>Section</b>	<b>Page</b>
<b>1 General introduction</b>	<b>1/1</b>
1.1 Introduction	1/1
1.2 Functions	1/4
1.2-1 General	1/4
1.2-2 TSX CAY module functions	1/5
1.3 Software setup	1/7
1.3-1 Programming movements	1/7
1.3-2 Configuring the axes	1/10
1.3-3 Adjusting the axes	1/11
1.3-4 Presymbolization	1/12
1.3-5 Debugging	1/13
1.3-6 Man-machine interface and control	1/14
1.4 TSX CAY module functions	1/14
<b>2 Performance and limitations</b>	<b>2/1</b>
2.1 Characteristics of the axis control functions	2/1
2.2 TSX CAY •1 module limitations	2/2
2.2-1 Functional limitations	2/2
2.3 Compatible absolute encoders	2/2

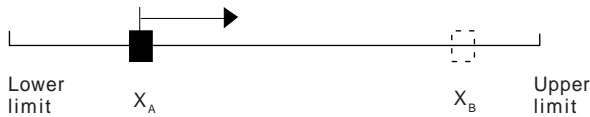
**Section****Page**

1.1 Introduction

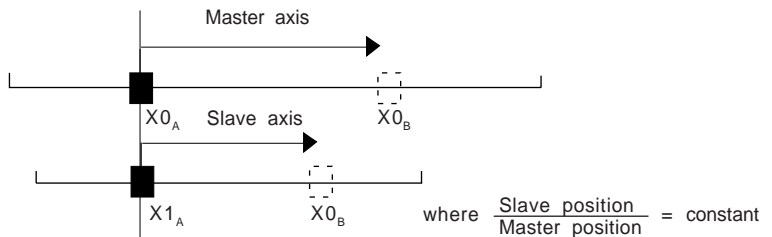


The servomotor axis control range for TSX Premium PLCs comprises 5 modules : **TSX CAY 21 / 41 / 22 / 42 / 33**.

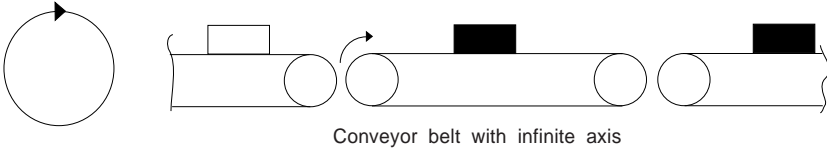
**TSX CAY 21 / 41 modules** (2 and 4 axes respectively) are used to control the movement of axes on limited travel machines.



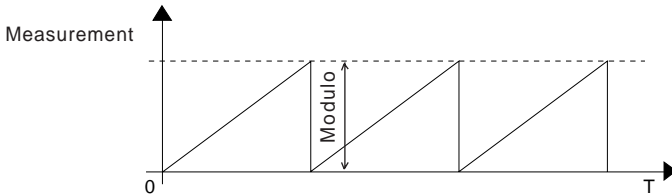
Master/slave applications, such as the one illustrated below, can also be implemented.



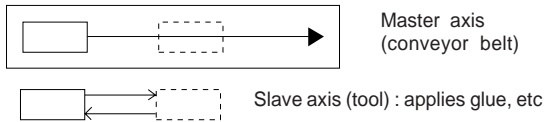
**TSX CAY 22 and TSX CAY 42 modules** (2 and 4 axes respectively) are used to control the movement of axes on infinite travel machines (typically rotary or machinable axes).



This type of application results in the existence of an area of measurement variation known as "MODULO", which can be illustrated as follows :



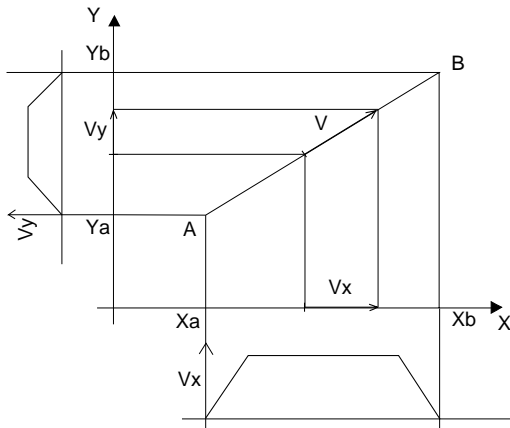
Object tracking type master/slave applications can also be created :



**The TSX CAY 33 module** (3 axes) is used for linear control of the movement of interpolated axes on Cartesian limited travel machines, enabling trajectory tracking either in one dimension (2 axes) or in three dimensions (3 axes).

This module can be used in the following configurations :

- 3 interpolated axes
- 2 interpolated axes and one independent axis
- 3 independent axes (use without interpolation)



This module does not offer circular interpolation which is required for processing contour applications.

---

PL7 software integrates motion control functions as standard for programming these axis control modules.

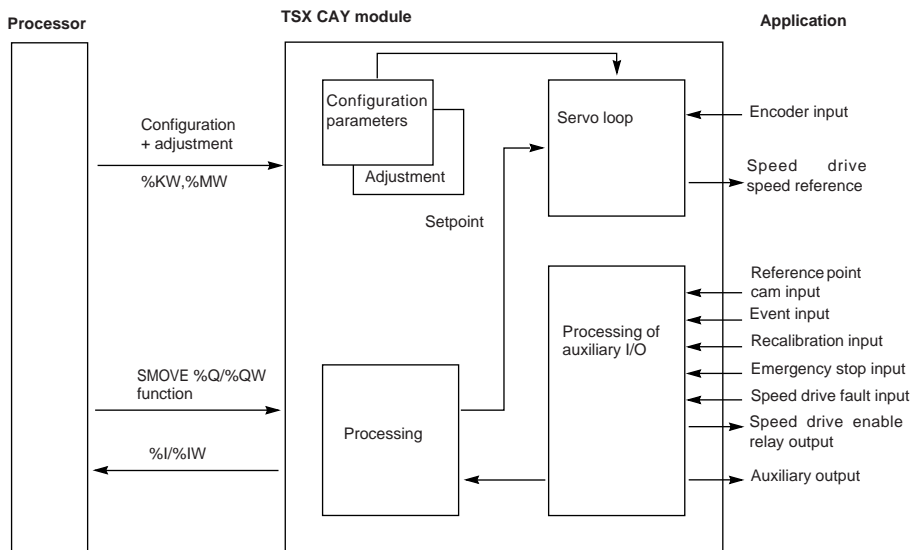
Elementary movements are managed from the main sequential control program of the machine, but performed and controlled by the TSX CAY modules. The TSX CAY axis control modules control the position of the moving part.

On each channel the position is measured by an incremental or absolute encoder, and the analog output controls a speed drive.

## 1.2 Functions

### 1.2-1 General

#### Block diagram of a channel



#### Application I/O

For each of the axes, axis control modules provide :

Inputs :

- An input for the acquisition of position measurements :
  - either via an RS 485 type incremental encoder (maximum frequency 500 kHz without multiplication, 1 MHz with multiplication by 4). The module will perform the multiplication by one or by four as required
  - or an absolute encoder, up to 25 data bits, with serial link and transmission according to SSI protocol (clock frequency 200 kHz)
- An input serving as a reference point cam (if an incremental encoder is selected)
- An event input
- An emergency stop input
- A recalibration on the fly input
- A speed drive fault input

Outputs :

- A  $\pm 10$  V analog output isolated from the logic part of the module, with a resolution of 13 bits + sign for controlling a speed drive connected to a self-starting synchronous DC motor, or to a self-controlled asynchronous DC motor
- A relay output to enable the speed drive
- A solid state auxiliary output



---

**Processing commands controlled from the PLC sequential program :**

Each movement is defined by an SMOVE motion control function in PL7 language or an XMOVE function for the linear interpolation of the TSX CAY 33 module. From this SMOVE or XMOVE command and the position of the moving part, the TSX CAY module generates the position/speed setpoint.

**Configuration and adjustment parameters :**

These parameters define the usage characteristics, limits, resolution, servo values, etc.

**Position control servo loop :**

The loop controller is of the proportional type with feedforward, in order to reduce deviation.

The user can choose from 3 types of profile for each axis : rectangular, trapezoid or triangular acceleration profile.

---

**1.2-2 TSX CAY module functions**

The following functions are offered by axis control modules :

- **Movement following the position of another axis** : one or more axes can be governed by a master axis. The movements of the slave axis follow all the movements of the master axis.
- **Movement following a periodic setpoint** : the position setpoint can be periodically transmitted directly by the PLC processor.
- **Recalibration on the fly** : this function (used with an incremental encoder) is used to control the position of the moving part and recalibrate the measurement when the recalibration input is activated. It can be used for movements with slip, in order to periodically recalibrate the position measurement.
- **Event-triggered processing** : events detected by the module can be used to activate an event-triggered task in the sequential program.
- **Immediate pause** : this function is used to momentarily stop a movement which is in progress (for example to synchronize axes with one another).
- **Deferred pause** : this function is used to momentarily stop a machine cycle without disturbing it.
- **Step by step mode** : this mode is used to execute a sequence of movements, stopping after each elementary instruction.
- **Motion control** : this function is used to detect any abnormal operations in the process (such as a faulty encoder), which mean that the position value does not change even when the moving part moves. This function is in addition to the checks usually performed on the TSX CAY 21/41 module. It is available from module version 1.9 onwards and requires version 3.0 PL7 software for its setup.

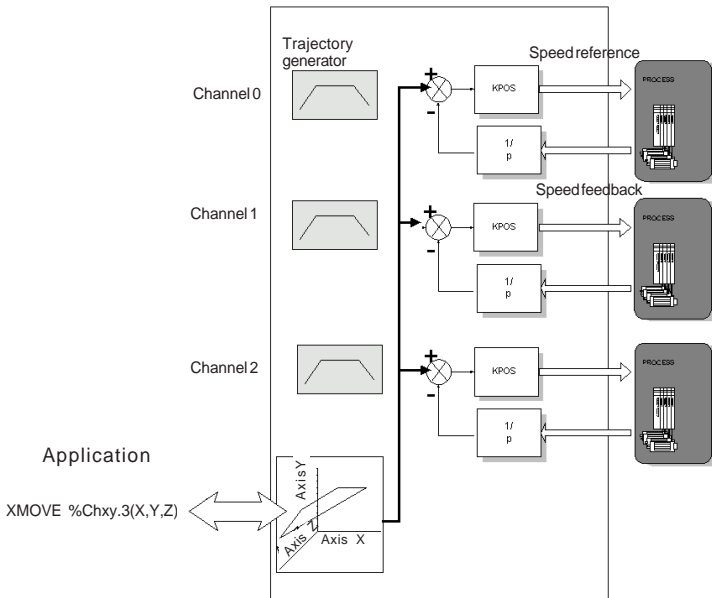
- **Infinite axis function** : (only available on TSX CAY •2 modules) this function is used to process **unlimited** axes ("conveyor belt" applications). The axis describes a motion which is always executed in the same direction.

It can also process two other types of application :

- spindle function
  - synchronized movement of infinite axes
- **Linear interpolation function** (only available on the TSX CAY 33 module)  
This function is used to associate 2 or 3 axes with the **Interpolation** function. It provides access to certain operating characteristics of numerically controlled machines.

### Important :

The 3 physical axes (X, Y, Z) use channels 0, 1 and 2.  
Channel 3 is dedicated to the linear interpolation function.



### 1.3 Software setup

The PL7 setup software provides :

- for programming movements, a movement control function, which can be used in Ladder language, Instruction list language or Structured Text language :
  - SMOVE for independent axes
  - XMOVE for interpolated axes
- for configuration, adjustment and moving axes, screens which are available in the configuration editor.

#### 1.3-1 Programming movements

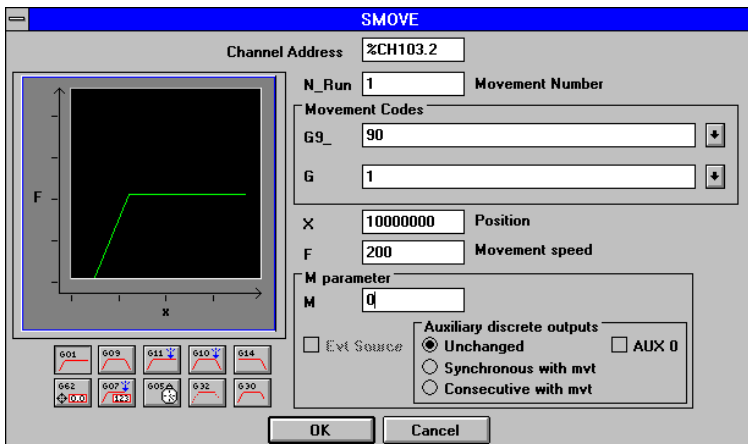
A movement is initiated by executing a •MOVE control function in the PL7 program.

##### Example 1 :

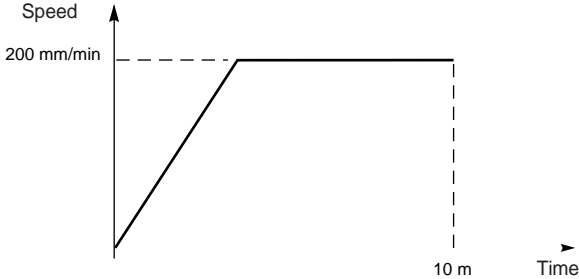
Go to the absolute position 10 000 000  $\mu\text{m}$ , at a speed of 200 mm/min, without stopping.



A screen can be used for assisted entry of the parameters of the SMOVE function in the operation block.



```
○ SMOVE%CH103.2 (1, 90, 01, 10000000, 200, 0) ○
```



Significance of each parameter (see complete description in part B1) :

- SMOVE** : motion control function
- %CH103.2** : address of CAY module on the rack (channel no. 2, position 03 in rack 1)
- 01** : movement number 1
- 90** : move to an absolute position
- 01** : instruction code corresponding to movement to a position without stopping
- 10 000 000** : position to be reached by the moving part in µm
- 200** : speed of the moving part in mm/min
- 0** : auxiliary output not assigned and events not enabled

**Example 2** : Go to the position defined by word %MD50 and to the speed defined by word %MD60.

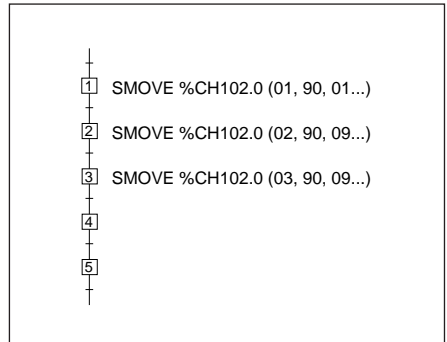
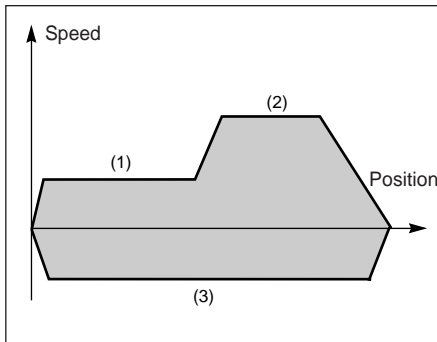
The screenshot shows the SMOVE configuration window. The title bar is blue and says 'SMOVE'. The 'Channel Address' is set to '%CH7.1'. 'N\_Run' is 2 and 'Movement Number' is also 2. Under 'Movement Codes', 'G9\_' is 'Absolute.' and 'G' is 'Movement to position with stop.'. 'X' is '%MD50' (Position) and 'F' is '%MD60' (Movement speed). 'M parameter' 'M' is '16#100'. There are radio buttons for 'Auxiliary discrete outputs': 'Unchanged', 'Synchronous with mvt' (selected), and 'Consecutive with mvt'. An 'Ext Source' checkbox is unchecked and 'AUX 0' is also unchecked. At the bottom are 'OK' and 'Cancel' buttons. A small graph on the left shows a trapezoidal speed profile.

(\*) Syntax similar to that for a numerical control program block written in ISO language.

## Programming a trajectory

A complete trajectory can be programmed by means of a series of SMOVE elementary motion control functions.

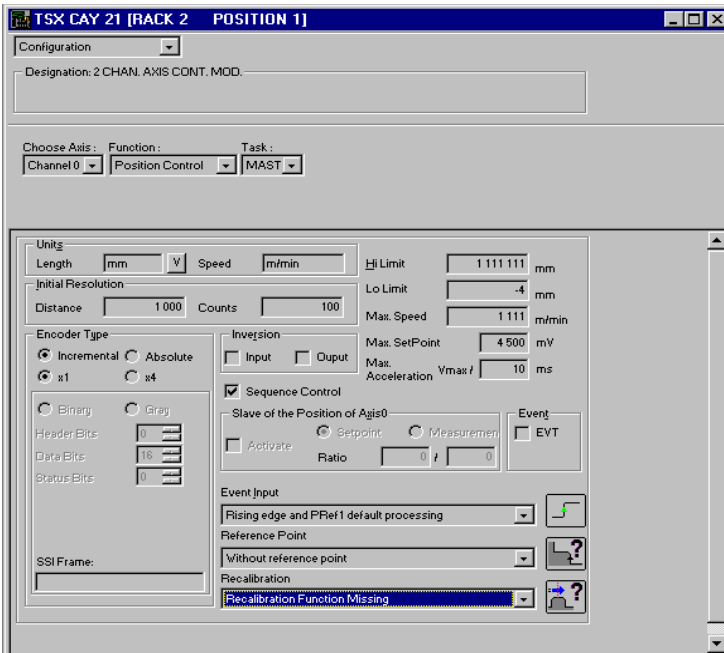
Grafset language is ideal for this type of programming. An elementary movement is associated with a step.



### 1.3-2 Configuring the axes

The configuration editor provides assistance with entering and modifying the values of the various axis configuration parameters. These parameters enable the operation of the axis control module to be adapted to the machine which is to be controlled.

Axis configuration parameters : units of measurement, type of encoder, encoder resolution, maximum and minimum position limits, maximum speed, etc, are linked to the machine and cannot be modified by the program.



Configuration parameters must be entered (no default configuration).

#### Note :

The configuration screens are adapted for the various types of TSX CAY modules.

### 1.3-3 Adjusting the axes

These parameters are linked to operation of the axes. They generally require the operations on and movements of the moving part to be known. These parameters are adjusted in online mode (they are initialized in offline mode).

Axis operating parameters :

- Resolution correction, etc
- Motion control : deviation, recalibration, overspeed, etc
- Stop control : delay, speed, target window, etc
- Servo (position loop) : position gain, feedforward coefficient, etc
- Commands : soft limits, acceleration, acceleration profile, etc
- Manual mode : Speed, reference point value, etc

These parameters can be modified by the program.

#### Note :

The adjustment screens are adapted for the various types of TSX CAY modules.

### 1.3-4 Presymbolization

Application-specific modules provide a way of allocating symbols automatically to objects which are associated with them. The user gives the generic symbol for channel %CHxy.i and all the symbols for the objects associated with this channel can then be generated automatically on request.

This presymbolization operation, performed in the variables editor, makes programming easier by using mnemonics rather than identifiers which are more difficult to handle.

These objects are symbolized using the following syntax :

#### User\_prefix\_Manufacturer\_suffix

where

The **User\_prefix** is the generic symbol given by the user to channel %CHxy.i (12 characters maximum),

The **Manufacturer\_suffix** is the part of the symbol which corresponds to the channel bit or word (20 characters maximum) given by the system.

In addition to the symbol, a manufacturer comment is generated automatically which gives a brief description of the role of the object.

**Example :** Auxil\_motor2 is the user prefix for channel 0.

Address	Type	Symbol	Comment
%CH0.0	CH	Auxil_motor2	
%I0.0.6	WWORD	Auxil_motor2_sync_nrun	Block SMOVE number in progress
%ID0.0	DWORD	Auxil_motor2_s_pos	Measured position of the axis \$<VOIE>
%ID0.0.2	DWORD	Auxil_motor2_speed	Measured speed of the axis \$<VOIE>
%ID0.0.4	DWORD	Auxil_motor2_remain	---
%ID0.0.7	DWORD	Auxil_motor2_pref	Axis captured position PRFIF
%Q0.9.0	WWORD	Auxil_motor2_mode_sel	Mode selection
%Q0.9.0.1	WWORD	Auxil_motor2_emv	Speed modulation coefficient (Feedrate override percentage)
%Q0.9.0.2	DWORD	Auxil_motor2_param	DIPDRV Parameter or INC, RP, HERE Parameter
%K0.9.0	WWORD		
%K0.9.0.1	WWORD		
%K0.9.0.2	WWORD		
%K0.9.0.3	WWORD		
%K0.9.0.8	WWORD		
%KD0.0.4	DWORD	Auxil_motor2_fmax	---
%KD0.0.6	DWORD		
%I0.0	EBOOL	Auxil_motor2_next	Ready for next command block
%I0.0.1	EBOOL	Auxil_motor2_done	All instructions have been completed
%I0.0.2	EBOOL	Auxil_motor2_as_fr	Fault on the axis \$<VOIE>
%I0.0.3	EBOOL	Auxil_motor2_as_ok	No fault on the axis \$<VOIE>
%I0.0.4	EBOOL	Auxil_motor2_hrd_err	Hardware error on the axis \$<VOIE>
%I0.0.5	EBOOL	Auxil_motor2_as_err	Presence of an error on the axis \$<VOIE>
%I0.0.6	EBOOL	Auxil_motor2_cmd_nok	Command refused
%I0.0.7	EBOOL	Auxil_motor2_noemotion	No motion on the axis \$<VOIE>
%I0.0.8	EBOOL	Auxil_motor2_at_pnt	Axis \$<VOIE> is in position
%I0.0.9	EBOOL		
%I0.0.10	EBOOL	Auxil_motor2_sys_err	System error on the axis \$<VOIE>
%I0.0.11	EBOOL	Auxil_motor2_conf_ok	Axis \$<VOIE> has been configured
%I0.0.12	EBOOL	Auxil_motor2_ref_ok	Axis \$<VOIE> has been calibrated
%I0.0.13	EBOOL	Auxil_motor2_as_ext	Image of the physical event input
%I0.0.14	EBOOL	Auxil_motor2_home	Image of the physical home switch input
%I0.0.15	EBOOL	Auxil_motor2_direct	Displacement in plus (+ 1), in minus (- 0) direction
%I0.0.16	EBOOL	Auxil_motor2_in_off	In Drive Off mode
%I0.0.17	EBOOL	Auxil_motor2_in_dir	In Direct Drive mode
%I0.0.18	EBOOL	Auxil_motor2_in_manu	In Manual mode
%I0.0.19	EBOOL	Auxil_motor2_in_auto	In Automatic mode



### 1.3-5 Debugging

In online mode, the configuration editor also provides the user with a control panel screen, giving him a quick visual display which he can use to control and observe the behavior of the axis.

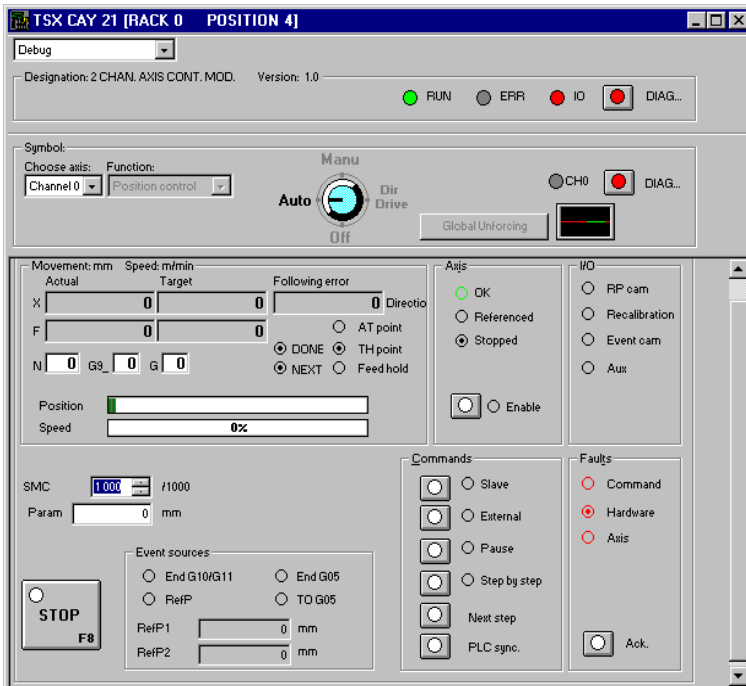
The control panel provides different information and commands according to the selected operating mode :

- automatic mode (Auto)
- manual mode (Man)
- direct drive (Dir\_Drive) mode
- measurement (Off) mode

The upper part of the debug screen is identical in all modes. It gives information on the operating state of the module and diagnostic information, and is used to choose the module channel and select the mode.

The lower part gives information and commands specific to the selected operating mode :

- information on the movement
- information on the state of the axis and the I/O
- manual movement commands (when this mode is selected)
- information on faults



The debug screens are adapted for the various types of TSX CAY modules.

### 1.3-6 Man-machine interface and control

The user can make use of all the commands and all the axis parameters and measurements in the processor in the form of language objects. He can thus design the control interface for his machine and include in it all or part of the axis control data.

This man-machine interface can be supported by CCX17 terminals.

### 1.4 TSX CAY module functions

TSX CAY axis control modules provide the following functions :

TSX CAY modules	21	41	22	42	33
<b>2/3 axis interpolation</b>	-	-	-	-	Yes
<b>Limited axes</b>	Yes	Yes	Yes	Yes	Yes
<b>Infinite axes</b>	-	-	Yes	Yes	Yes
<b>Follower axes</b>					
Static ratio	Yes	Yes	-	-	-
Dynamic ratio	-	-	Yes	Yes	-
<b>Speed drive offset correction</b>	-	-	Yes	Yes	Yes

## 2.1 Characteristics of the axis control functions

### Memory consumption of the axis control module

	Bit memory	Data zone	Program zone
TSX CAY 1	78	520	140
TSX CAY 2	78	376	232
TSX CAY 33 channel 3	78	264	170
Plus for 1st channel configured on 1			2130
Plus for 1st channel configured on 2/33			3600
Plus for 1st channel 3 configured on CAY 33			3600

Approximate sizes are given in words.

### Execution time

Reading of TSX CAY I/O	107 $\mu$ s
SMOVE function	80 $\mu$ s
XMOVE function	80 $\mu$ s
READ_STATUS	380 $\mu$ s   680 $\mu$ s ch. 3 TSX CAY 33
READ_PARAM	450 $\mu$ s
WRITE_PARAM	560 $\mu$ s
SAVE_PARAM	500 $\mu$ s
RESTORE_PARAM	590 $\mu$ s
Taking an adjustment into account (after a WRITE_PARAM instruction)	4 ms (CAY 2)   8 ms (CAY 4/33)
Taking the reconfiguration of a channel into account	2.5 s

### Module-specific cycle time

TSX CAY 2	2 ms
-----------	------

TSX CAY 4 and 33	4 ms
------------------	------

The cycle time also corresponds to the sampling periods for the servo loops.

### Processing time for TRL events

This corresponds to the time between the appearance of an event (eg : rising edge on the event input of the TSX CAY module) and the physical assignment of a discrete output (eg : TSX DSY 64 module).

TRL maximum = 5ms for a TSX CAY 2

TRL maximum = 7ms for a TSX CAY 4/33

---

## 2.2 TSX CAY •1 module limitations

---

### 2.2-1 Functional limitations

There are several TSX CAY •1 module versions. Some functions are only offered for version V1.5 or later and can only be installed with PL7 Pro or PL7 Junior software versions V1.7 or later :

- codes G32 and G30
- sequence check
- step by step mode
- referencing and clear reference with an absolute encoder in direct offset
- absolute encoder in assisted offset

The following function is only supported by TSX CAY •1 modules with software versions V1.9 or later and can only be installed with PL7 Pro/Junior software versions V3.0 or later :

- motion control

The module version is displayed :

- on the label on the side of the module
- in the module zone in the debug screens in online mode

---

## 2.3 Compatible absolute encoders

---

The following brands of encoder have been tested by Schneider Automation :

### **Codechamp :**

- COM 2G7 M 1212A  
11-30 volts, Gray code, 0 header bits, 24 data bits, 0 status bits, even parity.

### **IVO :**

- GM400 0 10 11 01  
24 volts, Gray code, 0 header bits, 25 data bits, 0 status bits, no parity.
- GM401 1 30 R20 20 00  
24 volts, Gray code, 0 header bits, 25 data bits, 1 status bit, even parity.

### **Hengstler :**

- RA58-M/1212 ES.41TBSG  
11-30 volts, Gray code, 0 header bits, 24 data bits, 1 status bit, no parity.

### **Stegmann :**

- AG 661 01  
10-30 volts, Gray code, 0 header bits, 24 data bits, 0 status bits, no parity.

### **IDEACOD :**

- SHM506S 428R / 4096 / 8192 / 26  
11-30 volts, Gray code, 25 data bits, 0 status bits, no parity.

<b>Section</b>	<b>Page</b>
<b>1 Tutorial</b>	<b>1/1</b>
1.1 Description of the example	1/1
1.2 Prerequisites	1/3
1.3 Application design	1/4
1.3-1 Software declaration of the PLC configuration used	1/4
1.3-2 Entering the configuration parameters for each axis	1/4
1.3-3 Entering the symbols for the application	1/7
1.3-4 Programming	1/9
1.3-5 Program transfer	1/13
1.4 Adjustment and debugging	1/14
1.4-1 Parameter adjustment	1/14
1.4-2 Using manual mode	1/16
1.4-3 Debugging	1/17
1.4-4 Archiving	1/17
<b>2 Setup methodology</b>	<b>2/1</b>
2.1 Setup methodology	2/1
<b>3 Configuration</b>	<b>3/1</b>
3.1 Configuring axis control modules	3/1
3.1-1 Introduction	3/1
3.1-2 Brief description of the configuration editor	3/1
3.2 Declaring the axis control modules	3/2

Section	Page
3.3 Entering the configuration parameters	3/4
3.3-1 Access to the parameter configuration screen	3/4
3.3-2 Type of axis (machine)	3/5
3.3-3 Type of encoder	3/5
3.3-4 Initial resolution	3/8
3.3-5 Measurement units	3/8
3.3-6 Upper and lower limits	3/9
3.3-7 Modulo	3/9
3.3-8 Maximum speed	3/10
3.3-9 Maximum setpoint	3/10
3.3-10 Event	3/11
3.3-11 Inversion	3/11
3.3-12 Sequence check	3/12
3.3-13 Maximum acceleration (and deceleration)	3/12
3.3-14 Position follower of axis 0	3/13
3.3-15 Reflex inputs	3/14
3.3-16 Reference point	3/15
3.3-17 Recalibration	3/17
3.3-18 Fault masking	3/17
3.3-19 Special functions	3/17
3.4 Confirming the configuration parameters	3/18
<b>4 Programming</b>	<b>4/1</b>
4.1 Programming principle	4/1
4.2 Operating modes	4/1
4.3 Programming in automatic mode : SMOVE function	4/2
4.3-1 Programming an SMOVE function	4/2
4.3-2 Entering the parameters of the SMOVE function	4/3
4.3-3 Description of elementary movements	4/7
4.3-4 Description of instructions	4/10
4.3-5 Sequence of movement commands	4/18

<b>Section</b>	<b>Page</b>
4.4 Programming : other functions	4/21
4.4-1 Recalibration on the fly function	4/21
4.4-2 Follower movement of another axis TSX CAY •1	4/22
4.4-3 Follower movement of another axis TSX CAY •2	4/23
4.4-4 Follower movement of an external periodic setpoint	4/26
4.4-5 Deferred "PAUSE" function	4/27
4.4-6 Step by step mode	4/28
4.4-7 Immediate "PAUSE" function	4/30
4.4-8 Event processing	4/31
4.5 Managing the operating modes	4/32
4.6 Fault management	4/33
4.6-1 Role	4/33
4.6-2 Principle	4/33
4.6-3 Programming	4/34
4.6-4 Summary table	4/35
4.6-5 Description of channel faults	4/35
4.6-6 Description of external hardware faults	4/36
4.6-7 Description of application faults	4/38
4.6-8 Description of command failure faults	4/41
4.6-9 Fault Masking	4/42
4.7 Management of manual mode	4/43
4.7-1 Selecting manual mode	4/43
4.7-2 Execution of manual commands	4/43
4.7-3 Detailed description of manual commands	4/44
4.8 Managing direct drive mode (DIRDRIVE)	4/48
4.8-1 Selecting direct drive	4/48
4.8-2 Executing commands in direct drive mode	4/48
4.9 Management of measurement mode (OFF)	4/49

Section	Page
<b>5 Adjusting the axes</b>	<b>5/1</b>
5.1 Operations prior to adjustment	5/1
5.1-1 Preliminary conditions	5/1
5.1-2 Preliminary checks	5/1
5.1-3 Adjusting the speed drive	5/1
5.2 Adjusting the configuration parameters	5/2
5.2-1 Access to the configuration parameters	5/2
5.2-2 Inversion parameters	5/2
5.3 Adjusting the parameters	5/4
5.3-1 Access to the adjustment parameters	5/4
5.3-2 ncoder offset	5/5
5.3-3 Adjusting the resolution	5/6
5.3-4 Adjusting the servo parameters	5/7
5.3-5 Adjusting the error control parameters	5/12
5.3-6 Manual mode parameters	5/16
5.3-7 Parameters associated with master/slave axes	5/17
5.4 Confirming and saving adjustment parameters	5/18
5.4-1 Confirming	5/18
5.4-2 Save	5/19
5.4-3 Restore	5/19
5.5 Reconfiguration in online mode	5/20



<b>Section</b>	<b>Page</b>
<b>6 Debugging an axis control program</b>	<b>6/1</b>
6.1 Principle of debugging an axis control program	6/1
6.2 Debug screens	6/2
6.2-1 Accessing the debug screens	6/2
6.2-2 User interface	6/2
6.2-3 Description of the debug screens	6/3
6.2-4 Measurement mode (Off)	6/5
6.2-5 Direct drive mode (Dir Drive)	6/6
6.2-6 Manual mode (Man)	6/7
6.2-7 Automatic mode (Auto)	6/10
6.3 Diagnostics	6/13
6.4 Archiving	6/14
6.5 Documentation	6/14
<b>7 Operation</b>	<b>7/1</b>
7.1 Designing a man-machine interface	7/1
7.1-1 Control station	7/1
7.1-2 Man-machine interface on CCX 17	7/1
<b>8 Diagnostics and maintenance</b>	<b>8/1</b>
8.1 Fault monitoring	8/1
8.2 Conditions for executing commands	8/1
8.3 Diagnostic help	8/2

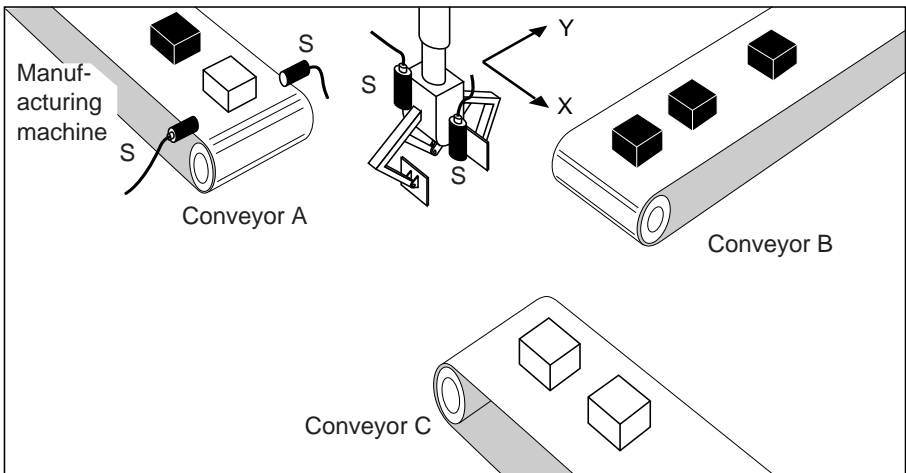
<b>Section</b>	<b>Page</b>
<b>9 Additional functions</b>	<b>9/1</b>
9.1 Teaching the positions	9/1
9.2 Example of use for a TSX CAY module for cutting a metal sheet which arrives continuously	9/3
9.2-1 Description of the application	9/3
9.2-2 Configuring the application	9/6
9.2-3 Programming the application	9/7
<b>10 Quick reference guide</b>	<b>10/1</b>
10.1 SMOVE programming function	10/1
10.2 General module data	10/3
10.3 Internal command data (implicit exchanges)	10/3
10.4 Internal status data (implicit exchanges)	10/4
10.5 Internal status data (explicit exchanges)	10/5
10.6 Adjustment parameters (explicit exchanges)	10/6
10.7 Block diagram of data exchanges	10/7
10.8 Block diagram of the TSX CAY module	10/8
<b>11 List of CMD_FLT error codes</b>	<b>11/1</b>
11.1 List of CMD_FLT error codes	11/1

## 1.1 Description of the example

This example is given for information and learning purposes. It will enable you to follow all the stages involved in setting up a TSX CAY axis control system without having to read all the documentation.

A transfer device removes all the items as they leave the manufacturing process. This device consists of a clamp which can move spatially on a plane (X and Y axes) parallel to the ground.

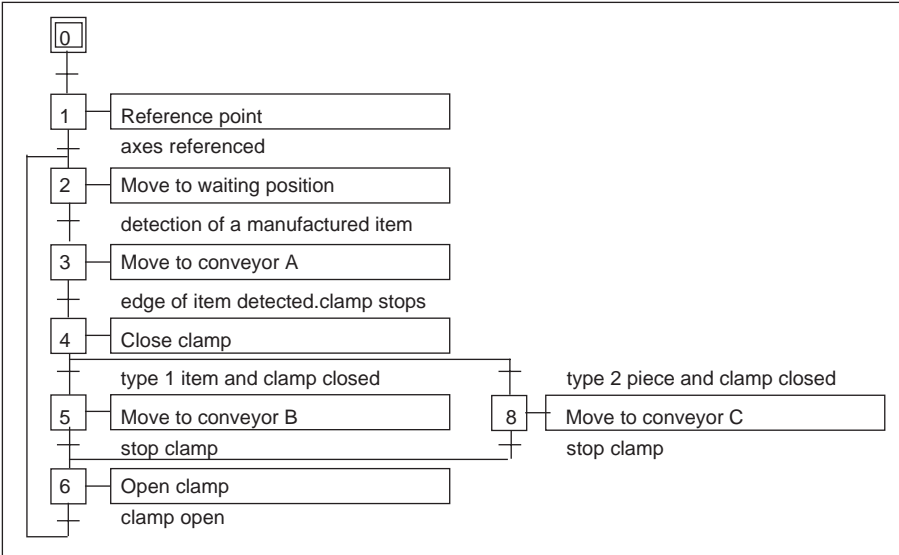
As soon as an item appears on exit conveyor A, the clamp will automatically pick it up and transfer it to conveyor B or conveyor C, depending on the type of item. The clamp then returns to waiting position ready to pick up another manufactured item as soon as one is detected.



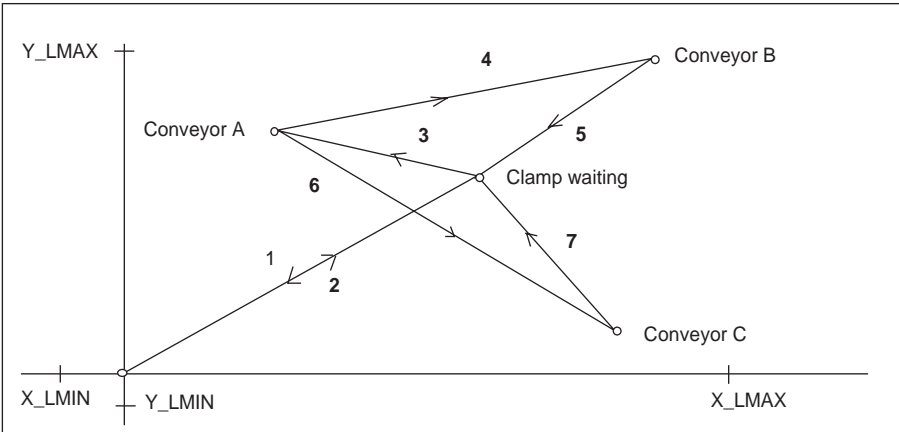
### I/O :

- S1 : cell for detecting the presence of a manufactured item,
- S2 : sensor for identifying the type of item,
- S3 : sensor for detecting clamp open/clamp closed,
- S4 : sensor for detecting the edge of an item (in the clamp), connected to module event input,
- ENC0 : incremental encoder for position on axis X,
- ENC1 : absolute encoder for position on axis Y,
- O/F clamp : open/close clamp command.

## Grafcet chart for the application



## Description of the trajectory



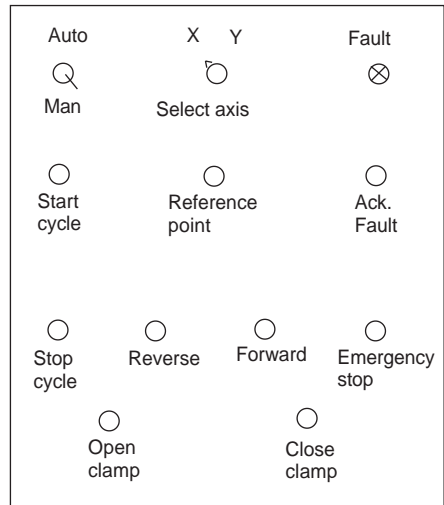
- 1 Reference point at speed  $V_{rp}$
- 2 Move at speed  $V_{ret}$  to waiting position  $(X_{wait}, Y_{wait})$  and stop
- 3 Move to conveyor A  $(X_A, Y_A)$  at speed  $V_A$  until the manufactured item is detected
- 4 Move at speed  $V_B$  to conveyor B  $(X_B, Y_B)$  and stop
- 6 Move at speed  $V_C$  to conveyor C  $(X_C, Y_C)$  and stop
- 5 and 7 Move at speed  $V_{ret}$  to waiting position  $(X_{wait}, Y_{wait})$  and stop

## Man-machine interface

The following commands are all on the front panel, and are used to control the moving part manually when there is a fault in the installation. The commands and the indicator lamps are controlled by an input module and a discrete output module.

Description of the commands :

- **Auto/Man** : switch for selecting the operating mode,
- **Start Cycle** : automatic execution of the cycle,
- **Stop Cycle** : automatic cycle stop,
- **Select axis X/Y** : selects the axis to be controlled in manual mode,
- **Reference point** : manual reference point for the selected axis,
- **Forward/Reverse** : manual move command in positive or negative direction, for the selected axis.
- **Fault** : indicator lamp signaling any hardware or application fault,
- **Ack. Fault** : fault acknowledgment command,
- **Emergency stop** : immediate stop of the moving part whatever mode is selected.
- **Open clamp** : open clamp command,
- **Close clamp** : close clamp command.



## 1.2 Prerequisites

Only functions which are specific to axis control will be described here. It is therefore assumed that the following operations have been performed :

- PL7 software has been installed,
- the hardware has been installed : module, speed drives and encoders controlling the 2 axes have been wired.

## 1.3 Application design

### 1.3-1 Software declaration of the PLC configuration used

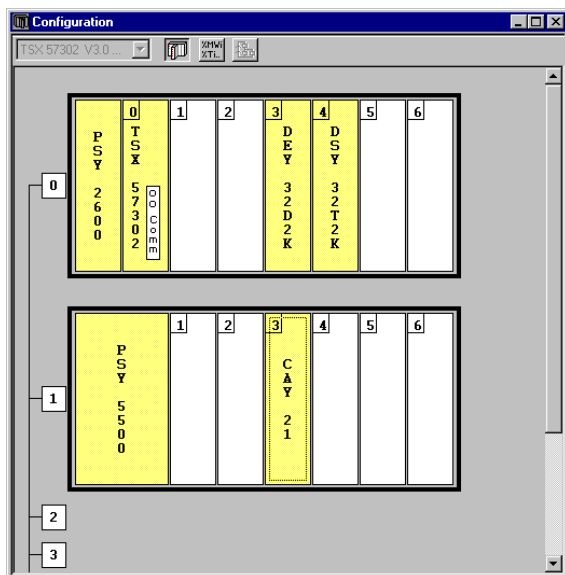
Launch PL7 software, select the **File/New** command, select a TSX 57 2• or TSX 57 3• processor and check the Grafcet box.

Access the configuration editor from the Application Browser. To do this :

1. Open the Station folder (double-click on the icon or click on the plus sign),
2. Open the Configuration folder (double-click on the icon or click on the plus sign),
3. Double-click on the Hardware configuration icon.

Then select each element of the PLC configuration. The following selections have been made in this application :

- rack 0 and rack 1 : TSX RKY 8E
- 32-input module : TSX DEY 32D2K in position no.3 of rack 0
- 32-output module : TSX DSY 32T2K in position no.4 of rack 0
- 2-axis control module : TSX CAY 21 in position no.3 of rack 1



### 1.3-2 Entering the configuration parameters for each axis

Select position no.3 of rack 1 and execute the **Edit/Open Module** command (or double-click on the selected module).

#### Configuration of channel 0

For channel 0, select the position control function and the MAST task.

Enter the parameter values in the screen below as shown in the table.

**Symbol:**  
**Choose Axis :** Channel 0 **Function :** Position Control **Task :** MAST

---

**Units**  
 Length:  Speed:

**Initial Resolution**  
 Distance:  Counts:

**Encoder Type**  
 Incremental  Absolute  
 x1  x4

Binary  Gray  
 Header Bits:   
 Data Bits:   
 Status Bits:

**SSI Frame:**

**Inversion**  
 Input  Output

Sequence Control

**Slave of the Position of Axis0**  
 Setpoint  Measurement  
 Activate Ratio:  /

**Event**  
 EVT

**Event Input**  
 Rising edge and PRef1 default processing

**Reference Point**  
 Long Cam / Zero Marker / - Direction

**Recalibration**  
 Recalibration Function Missing

Hi Limit:   $\mu\text{m}$   
 Lo Limit:   $\mu\text{m}$   
 Max. Speed:  mm/min  
 Max. SetPoint:  mV  
 Max. Acceleration Ymax f:  ms

Parameter	Description	Value	Comment
Units	Physical length unit	$\mu\text{m}$	-
	Physical speed unit	mm/min	calculated automatically
Initial resolution	Distance	2000	-
	No. of points	500	-
Encoder type		Incremental	Default selection
		x 1	Default selection
Max setpoint		9000 mV	-
Max speed	Max speed of the moving part	5400	-
Max acceleration		200 ms	-
Upper limit	Upper limit for the axis	500000	-
Lower limit	Lower limit for the axis	-5000	-
Event		Rising edge and PRef1	
Reference point		Long cam / Zero marker / Negative direction	

Confirm the entry using the **Edit/Confirm** command or the  icon.

## Configuration of channel 1

For channel 1 select the position control function and the MAST task.  
Enter the parameter values in the screen below as shown in the table.

Symbol:	
Choose Axis :	Function :
Channel 1	Position Control
Task :	MAST


  

Units	Length	um	Speed	mm/min	Hi Limit	900 000	um
Initial Resolution	Distance	4 000	Counts	4 000	Lo Limit	0	um
Encoder Type	Incremental	<input type="radio"/>	Absolute	<input checked="" type="radio"/>	Max. Speed	5 400	mm/min
	Direct Off.	<input checked="" type="radio"/>	Assisted Off.	<input type="radio"/>	Max. SetPoint	9 000	mV
	Binary	<input checked="" type="radio"/>	Gray	<input type="radio"/>	Max. Acceleration	Vmax /	300
	Header Bits	0	Data Bits	24	Status Bits	0	
	Parity	<input checked="" type="checkbox"/>	Even	<input type="radio"/>	Odd	<input checked="" type="radio"/>	
	SSI Frame:	x24x l					
	Inversion	Input	<input type="checkbox"/>	Output	<input type="checkbox"/>		
	Sequence Control	<input checked="" type="checkbox"/>					
	Slave of the Position of Axis0	Setpoint	<input checked="" type="radio"/>	Measurement	<input type="radio"/>	Event	<input type="checkbox"/>
	Activate	Ratio				EVT	<input type="checkbox"/>
	Event Input	Rising edge and PRef1 default processing					
	Reference Point	Without reference point					
	Recalibration	Recalibration Function Missing					

Parameter	Description	Value	Comment
Units	Physical length unit	µm	-
	Physical speed unit	mm/min	calculated automatically
Initial resolution	Distance	4000	-
	No. of points	4000	-
Type of encoder		Absolute	-
	Offset	Direct	-
	Code	Gray	-
	No. of header bits	0	-
	No. of data bits	24	-
	No. of status bits	0	-
	Parity	Odd	-
Max setpoint		9000 mV	-
Speed	Max speed of the moving part	5400	-
Max acceleration		300 ms	-
Upper limit	Upper limit for the axis	900 000	-
Lower limit	Lower limit for the axis	0	-
Event		Rising edge and PRef1	
Reference point		Short cam / Negative direction	



Confirm the entry using the **Edit/Confirm** command or the  icon.

At the level of the basic configuration editor screen, confirm the configuration using the **Edit/Confirm** command or the  icon.

### 1.3-3 Entering the symbols for the application

This is performed by double-clicking on the **Variables** icon and then the **I/O** icon in the **Application Browser**.

Variables				
<input checked="" type="checkbox"/> Parameters	I/O	Module Address	3	<input type="checkbox"/> Entry Field
Address	Type	Symbol	Comment	
%I3.3	EBOOL	Auto_man	SWITCH for selecting AUTOMATIC (=0) or MANUEL (=1) mode	
%I3.3.2	VORD			
%I3.4	EBOOL	Start_cycle	pushbutton to START automatic cycle	
%I3.4.2	VORD			
%I3.5	EBOOL	Stop_cycle	pushbutton to STOP automatic cycle	
%I3.5	EBOOL			
%I3.6	EBOOL	Select_x_y	selection of axis to be controlled manually (1=X, 0=Y)	
%I3.6.2	VORD			
%I3.7	EBOOL	Rf_man	manual reference point	
%I3.7.2	VORD			
%I3.8	CH			
%I3.8	EBOOL	Forward	move moving part in positive direction	
%I3.8	EBOOL			
%I3.8	VORD			

Symbol	Object	Role
Sensor_1	%I3.0	Cell for detecting the presence of a manufactured item
Sensor_2	%I3.1	Sensor for identifying the type of item (0=type 2, 1=type 1)
Sensor_3	%I3.2	Sensor for detecting clamp open/clamp closed
Auto_man mode	%I3.3	Switch for selecting AUTOMATIC (=0) or MANUAL (=1)
Start_cycle	%I3.4	Pushbutton to start automatic cycle
Stop_cycle	%I3.5	Pushbutton to stop automatic cycle
Select_x_y	%I3.6	Selection of axis to be controlled manually (1=X, 0=Y)
Rf_man	%I3.7	Manual reference point
Forward	%I3.8	Move moving part in positive direction
Reverse	%I3.9	Move moving part in negative direction
Ack_flt	%I3.10	Fault acknowledgment
Emg_stop	%I3.12	Emergency stop
O_clamp	%I3.13	Pushbutton to open the clamp
C_clamp	%I3.14	Pushbutton to close the clamp
Clamp	%Q4.0	Open/close clamp actuating command (0=open, 1=close)
Fault	%Q4.1	Fault indication
X_wait	%MD50	Waiting position (X axis)
Y_wait	%MD52	Waiting position (Y axis)
X_b	%MD54	Position of conveyor B (X axis)
Y_b	%MD56	Position of conveyor B (Y axis)
X_c	%MD58	Position of conveyor C (X axis)
Y_c	%MD60	Position of conveyor C (Y axis)

Symbol	Object	Value	Role
Cycle	%M0		Condition of the machine in work mode
Speed_r_p_x	%KD0	1000	Reference point speed on X axis
Speed_x_wait	%KD4	1200	Speed towards waiting position, X axis
Speed_y_wait	%KD6	1200	Speed towards waiting position, Y axis
Speed_pos_a_x	%KD8	1500	Speed towards conveyor position A, X axis
Speed_pos_a_y	%KD10	1500	Speed towards conveyor position A, Y axis
Speed_pos_b_x	%KD12	1200	Speed towards conveyor position B, X axis
Speed_pos_b_y	%KD14	1200	Speed towards conveyor position B, Y axis
Speed_pos_c_x	%KD16	1800	Speed towards conveyor position C, X axis
Speed_pos_c_y	%KD18	1800	Speed towards conveyor position C, Y axis

## Entering symbols for the axis control module

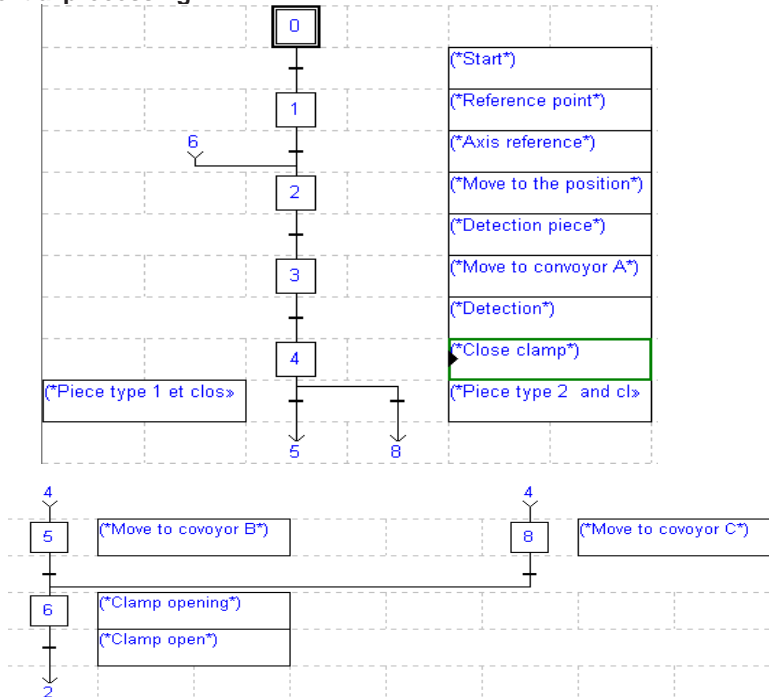
Symbol	Object	Symbol	Object
Axis_x	%CH103.0	Axis_y	%CH103.1
Next	%I103.0	Next_y	%I103.1
Done	%I103.0.1	Done_y	%I103.1.1
Error	%I103.0.2	Error_y	%I103.1.2
Ok	%I103.0.3	Ok_y	%I103.1.3
Hard_err_x	%I103.0.4	Hard_err_y	%I103.1.4
Axis_error_x	%I103.0.5	Axis_error_y	%I103.1.5
Ref_cmd_x	%I103.0.6	Ref_cmd_y	%I103.1.6
At_point	%I103.0.9	At_point_y	%I103.1.9
Calib	%I103.0.14	Calib_y	%I103.1.14
Mode_drive_off	%I103.0.20	Mode_drive_off_y	%I103.1.20
Mode_dir_drive	%I103.0.21	Mode_dir_drive_y	%I103.1.21
Mode_manual	%I103.0.22	Mode_manual_y	%I103.1.22
Mode_auto	%I103.0.23	Mode_auto_y	%I103.1.23
Varvalid_x	%I103.0.40	Varvalid_y	%I103.1.40
Dirdrive	%Q103.0	Dirdrive_y	%Q103.1
Jog_p	%Q103.0.1	Jog_p_y	%Q103.1.1
Jog_m	%Q103.0.2	Jog_m_y	%Q103.1.2
Inc_p	%Q103.0.3	Inc_p_y	%Q103.1.3
Inc_m	%Q103.0.4	Inc_m_y	%Q103.1.4
Setrp	%Q103.0.5	Setrp_y	%Q103.1.5
Rp_here	%Q103.0.6	Rp_here_y	%Q103.1.6
Acq_def	%Q103.0.8	Acq_def_y	%Q103.1.8
Enable	%Q103.0.9	Enable_y	%Q103.1.9
Event_uc	%Q103.0.10	Event_uc_y	%Q103.1.10
Posrp	%MD103.0.41	Posrp_y	%MD103.1.41

### 1.3-4 Programming

The programming in this example uses Grafcet structure :

- the sequential processing for the sequential description of the application : processing of the automatic cycle,
- the preprocessing for managing the operating modes,
- the post-processing for the execution of manual mode.

#### Sequential processing



Step 0 :

Transition X0 > X1

!(\*Channel X not faulty, clamp open, switch Auto\_man to Auto, start cycle, channel Y not faulty and automatic mode active\*)

NOT Error AND NOT Sensor\_3 AND NOT Auto\_man AND Cycle  
AND NOT Error\_y AND Mode\_auto

Step 1 : Action on activation

!(\*Reference point on X axis\*)

S MOVE Axis\_x(1,90,14,0,Speed\_r\_p\_x,16#0000);

Transition X1 > X2

!(\*Test : X axis ready and referenced\*)

Done AND Calib

---

```
Step 2 : Action on activation
!(*Move to waiting position (Xwait, Ywait)*)
SMOVE Axis_x(2,90,9,X_wait,Speed_x_wait,16#0000);
SMOVE Axis_y(2,90,9,y_wait,Speed_y_wait,16#0000);

Transition X2 > X3
!(*Mobile in waiting position and item detected on conveyor A*)
Sensor_1 AND Next AND Cycle AND Next_y

Step 3 : Action on activation
!(*Move to conveyor A*)
SMOVE Axis_x(3,90,10,150000,Speed_pos_a_x,16#0000);
SMOVE Axis_y(3,90,10,280000,Speed_pos_a_y,16#0000);

Transition X3 > X4
!(*Moving part in position to pick up item detected on conveyor A*)
At_point AND Next AND Next_y AND At_point_y

Step 4 : Continuous action
!(*Close clamp*)
SET Clamp;

Transition X4 > X5
!(*Type 1 item and clamp closed*)
Sensor_2 AND Sensor_3

Step 5 : Action on activation
!(*Move to conveyor B*)
SMOVE Axis_x(4,90,9,X_b,Speed_pos_b_x,16#0000);
SMOVE Axis_y(4,90,9,Y_b,Speed_pos_b_y,16#0000);

Transition X4 > X8
!(*Type 2 item and clamp closed*)
Not Sensor_2 AND Sensor_3

Step 8 : Action on activation
!(*Move to conveyor C*)
SMOVE Axis_x(5,90,9,X_c,Speed_pos_c_x,16#0000);
SMOVE Axis_y(5,90,9,Y_c,Speed_pos_c_y,16#0000);

Transition X5 > X6
!(*Moving part in position on conveyor B*)
At_point AND Next AND Next_y AND At_point_y

Transition X8 > X6
!(*Moving part in position on conveyor C*)
At_point AND Next AND Next_y AND At_point_y

Step 6 : Continuous action
!(*Clamp opening*)
RESET Clamp;

Transition X6 > X2
!(*Clamp open*)
NOT Sensor_3 AND Cycle
```

---

## Preprocessing

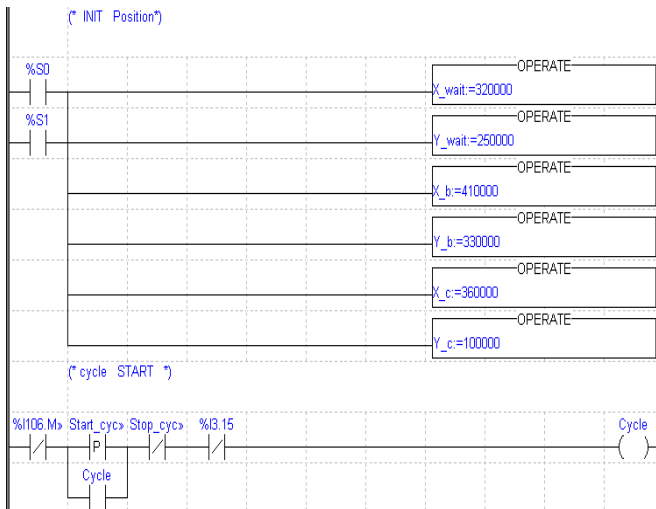
Preprocessing includes the management of the operating modes.

On a blocking fault :

- the chart freezes,
- the operator can then control the moving part in manual mode and correct and acknowledge the fault from the front panel.
- the chart is reinitialized when the fault has disappeared and has been acknowledged.

When changing to manual mode :

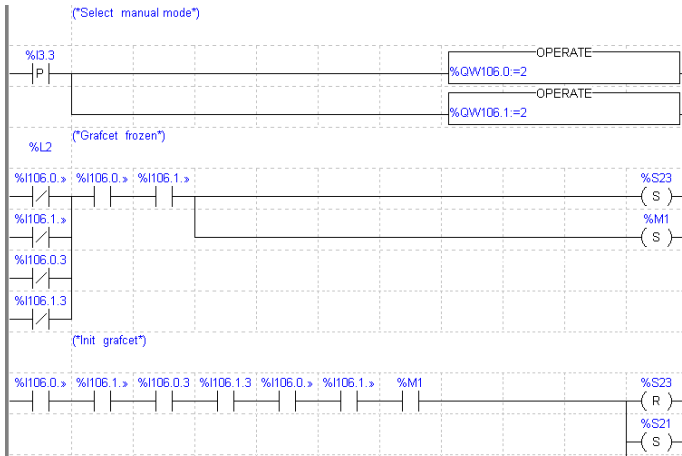
- the chart freezes,
- the chart is reinitialized when AUTOMATIC mode is reselected.



%I103.M>> = %I103.MOD.ERR

Start\_>> = Start\_cycle

Stop\_>> = Stop\_cycle



Auto\_m>> = Auto\_man  
 Varvalid>>= Varvalid\_x  
 Varvalid>>= Varvalid\_y  
 Mode\_>> = Mode\_auto  
 Mode\_>> = Mode\_auto\_y  
 %M1 = Grafcet frozen

## Postprocessing

Manual mode is managed in post-processing.

```
! (*Testing the selected mode*)
  IF Mode_auto AND Mode_auto_y AND Config AND Config_y
  THEN JUMP %L200;
  END_IF;
! (*Selecting the axis to drive*)
  %L100: IF NOT Selection_x_y
  THEN JUMP %L200;
  END_IF;
! (*Manual setpoint command of X axis*)
  IF RE Po_man
  THEN Posrp:=0; SET setrp; Fmanu_x:=1000; WRITE_PARAM Axis_x;
  END_IF;
  IF NOT Po_man
  THEN RESET Setrp;
  END_IF;
! (*Moving part in + direction of X axis*)
  Jog_p:=front;
! (*Moving part in - direction of X axis*)
  Jog_m:=rear;
! %L200: IF Selection_x_y
  THEN JUMP %L300;
  END_IF;
! (* Moving part in + direction of Y axis*)
  Jop_p_y:=front;
! (* Moving part in - direction of Y axis*)
  Jog_m_y:=rear;
! (*Opening the clamp*)
  %L300: IF Auto_man AND op_clamp
  THEN RESET Clamp;
  END_IF;
  (*Closing the clamp*)
  IF Auto_man AND cl_clamp
  THEN SET Clamp;
  END_IF;
! (*Defaults Acknowledgement*)
  Ack_def:=Ack_def_y:=Ack_defaults;
! %L999;
```

### 1.3-5 Program transfer


Once the program has been entered, this operation consists of transferring the configuration and the program to the PLC processor memory :

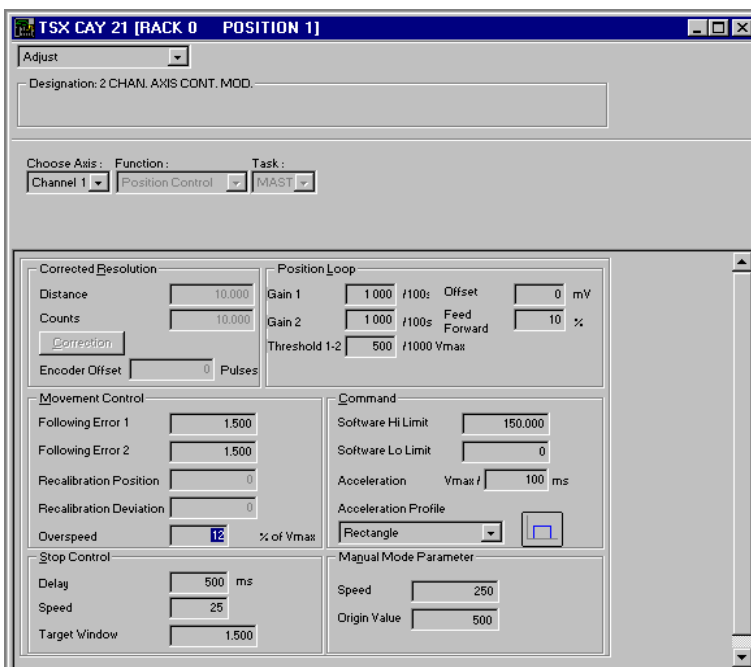
- connect the terminal to the PLC using the **PLC/Connect** command,
- launch the **PLC/Transfer** command, select the "Terminal -> PLC" option, then confirm.

## 1.4 Adjustment and debugging

### 1.4-1 Parameter adjustment

As a safety measure, first perform the preliminary operations described in section 6.2. Then perform the following operations :

- set the PLC to RUN mode,
- select the **Application/Configuration** command or click on the  icon.
- select position no.3 of rack 1 and execute the **Edit/Open Module** command (or double-click on the selected module).
- select the "**View/Adjust**" command.




The screenshot shows the 'Adjust' window for 'TSX CAY 21 [RACK 0 POSITION 1]'. The window title is 'Adjust' and the designation is '2 CHAN. AXIS CONT. MOD.'. The 'Choose Axis' is set to 'Channel 1', 'Function' is 'Position Control', and 'Task' is 'MAST'. The 'Corrected Resolution' section shows 'Distance' and 'Counts' both set to 10,000. The 'Position Loop' section shows 'Gain 1' and 'Gain 2' both set to 1,000, 'Offset' set to 0 mV, 'Feed Forward' set to 10 %, and 'Threshold 1-2' set to 500 /1000 Vmax. The 'Movement Control' section shows 'Following Error 1' and 'Following Error 2' both set to 1,500, 'Recalibration Position' and 'Recalibration Deviation' both set to 0, 'Overspeed' set to 12 % of Vmax, and 'Stop Control' section shows 'Delay' set to 500 ms, 'Speed' set to 25, and 'Target Window' set to 1,500. The 'Command' section shows 'Software Hi Limit' set to 150,000, 'Software Lo Limit' set to 0, 'Acceleration Vmax t' set to 100 ms, and 'Acceleration Profile' set to 'Rectangle'. The 'Manual Mode Parameter' section shows 'Speed' set to 250 and 'Origin Value' set to 500.

The following table shows the parameters modified in this example. The other parameters have kept their default values.

Parameter	Value
Target window	320 $\mu\text{m}$
Speed (manual mode)	5400 mm/min
RP value	0 $\mu\text{m}$

- confirm the values entered using the **Edit/Confirm** command,


or click on the  icon



- select channel 1 in the channel zone

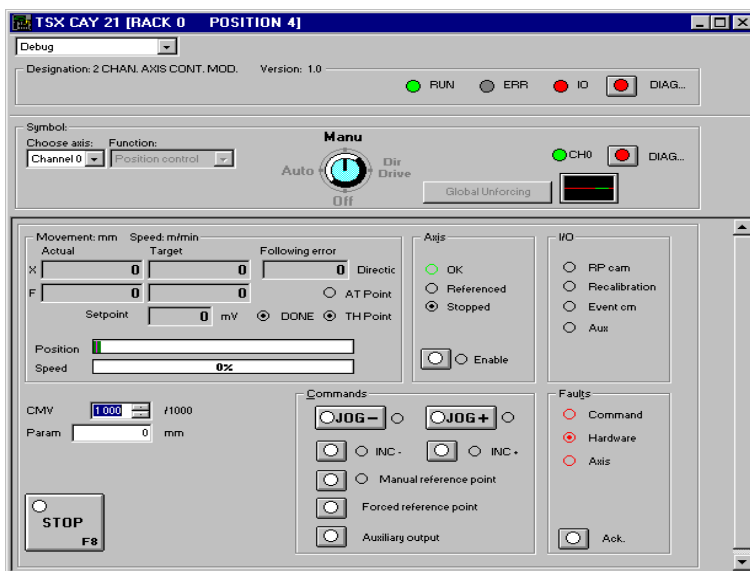
The following table shows the parameters modified in this example. The other parameters have kept their default values.

Parameter	Value
Encoder offset	8 388 607
Deviation 1 and 2	8000 $\mu\text{m}$
Target window	8000 $\mu\text{m}$
Speed (manual mode)	5400 mm/min


- confirm the values entered using the **Edit/Confirm** command or click on the 
- save these values in the PLC processor by selecting the **Utilities/Save Parameters** command,

## 1.4-2 Using manual mode

If a user wishes to move a moving part without performing the programming phase, select Manual mode. To do this, access the debug screen in online mode and activate the **Tool/Configuration** command then select the TSX CAY module to be opened and execute the **Service/Open the Module** command (or double-click on the module to be opened). The debug screen is selected by default.



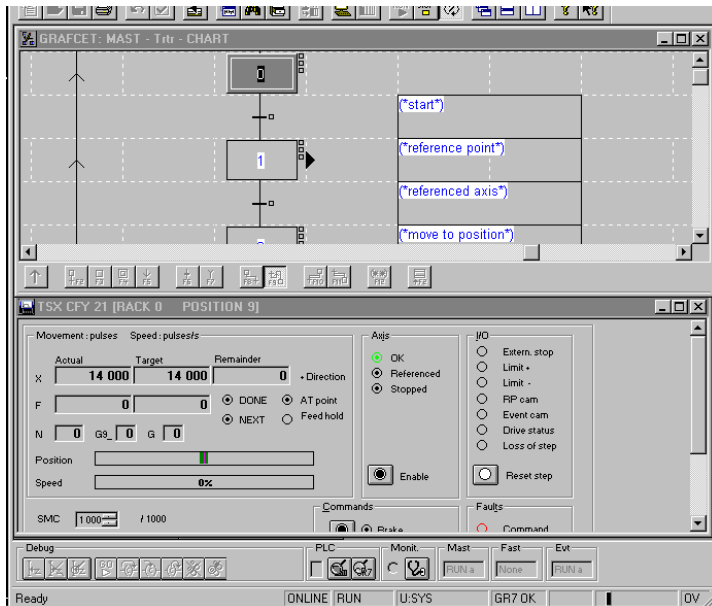
Perform the following operations using the debug screen :

- set the PLC to RUN (**PLC/Run** command or click on the  icon),
- select the axis to be controlled : channel 0 (X axis) or channel 1 (Y axis),
- select manual mode with the mode selector in the **Man** position,
- click on the **Enable** button in the **Axis** zone (enabling the speed drive safety relay)
- acknowledge any faults by clicking on the **Ack** button in the **Faults** field,
- set a reference point :
  - either by selecting the **Manual Reference Point** command,
  - or by selecting the **Forced Reference Point** command. In this case, first enter the value of the position of the moving part in relation to the reference point in the **Param** field,
- perform the positive direction movements using the **JOG+** command or the negative direction movements using the **JOG-** command. The position of the moving part is displayed in the X field and the speed in the F field in the **Movement/Speed** zone.

### 1.4-3 Debugging

To debug the program :

- set the PLC to RUN mode,
- display the TSX CAY module debug screen,
- at the same time display the Grafcet chart screen to follow the progress of the sequential processing,
- start the program by pressing the "Start\_cycle" button on the front panel.



### 1.4-4 Archiving

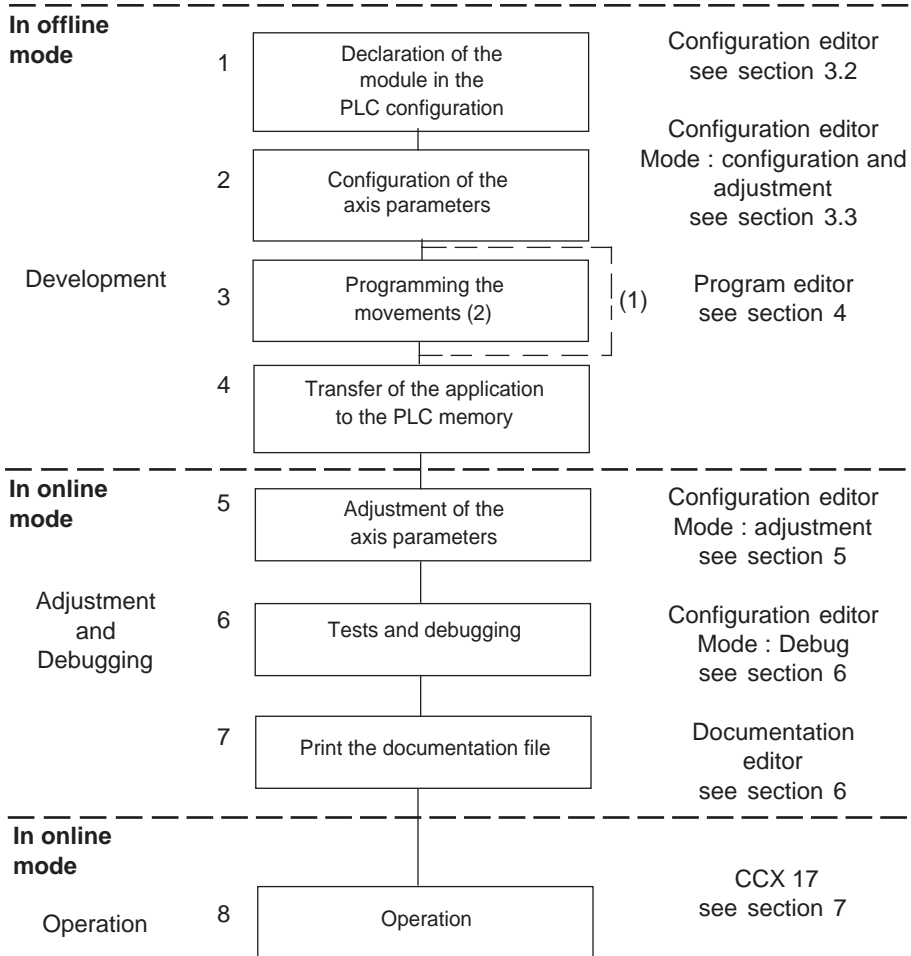
When debugging of the program is completed :

- save the parameters again if they have been modified during debugging by selecting the **Utilities/Save Parameters** command,
- transfer the application from the PLC processor to the hard disk to archive it, using the **PLC/Transfer** command, "PLC--> Terminal" option. Then execute the **File/Save As** command, give the application a name and confirm.



## 2.1 Setup methodology

The tutorial has shown the various phases in setting up an axis control application. The flowchart below summarizes these phases.



(1) If the user wishes, before programming, to move the moving part on the various axes in Manual mode, he can leave out operation 3. However, operations 1, 2, 4, 5 and 6 are compulsory.

(2) The programming operation may be preceded by symbolization of the variables which may be performed with the help of the variables editor.

The variables editor offers the Presymbolization function which automatically generates symbols for the axis control module (refer to sections 1.3-4 and 5.10).



## 3.1 Configuring axis control modules

### 3.1-1 Introduction

Before creating an application program, the physical and software operating context in which it will be executed must be defined : type of TSX Premium processor, I/O modules used.

Programming axis control modules also requires the configuration parameters of the axes used to be defined.

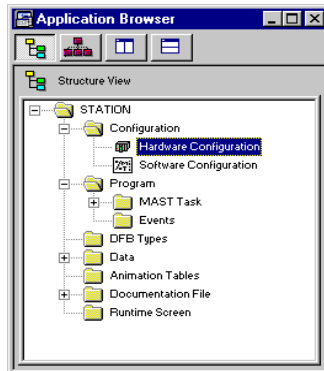
PL7 software provides the configuration editor to perform these operations easily.

This editor also provides access to the adjustment parameters of the axes, and during operation online to the application, it is used to access the debug functions.

### 3.1-2 Brief description of the configuration editor

The Application Browser accesses the configuration editor. To do this :

- 1 Open the **Station** folder (double-click on the icon or click on the plus sign),
- 2 Open the **Configuration** folder (double-click on the icon or click on the plus sign),
- 3 Double-click on the **Hardware configuration** icon.



If the **Application Browser** window is not open on the screen :

- pull down the **Tools** menu and activate the **Application Browser** command,

or

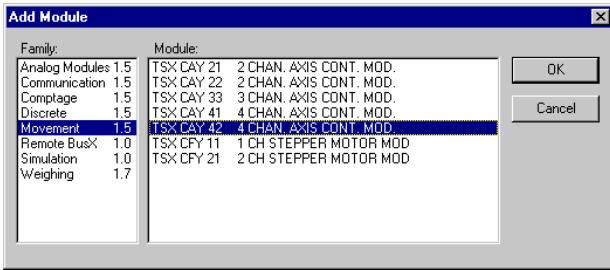
- in the toolbar, click on the Application Browser icon :



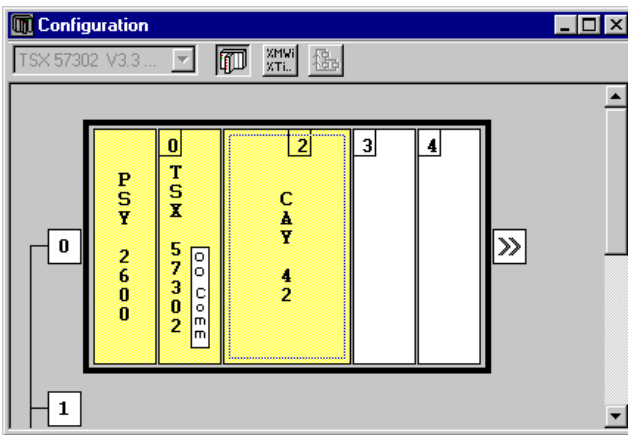
### 3.2 Declaring the axis control modules

This operation consists of declaring the positions which the TSX CAY axis control modules occupy in the PLC I/O configuration and determining their use in the master or fast task.

- Access the configuration editor.
- Select and confirm the rack where the TSX CAY axis control module is to be installed.
- Select the type of rack and confirm with **OK**.
- Select and confirm the position in the rack where the TSX CAY axis control module is to be installed.
- Select the Motion Control family, then in this family select the TSX CAY axis control module and confirm with **OK**.



- After confirming, the module is declared in its position (the position contains the module reference).



**Notes :**

The TSX CAY 41/42/33 modules occupy 2 positions. To install a double module in position n, the preceding slot must be empty.



**Example :**

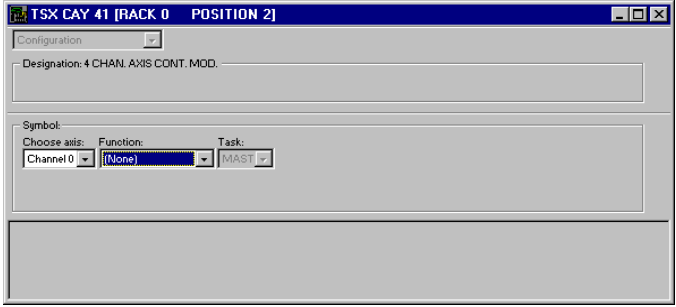
To install a TSX CAY 42 module in the position 2 slot, the preceding slot (position 1) must be free (see diagram opposite).

To move a module from one position to another, select the module and activate the command to **Edit/Move** a module, then set the target position or simply select the module using the mouse and move it, holding the left mouse button down, to the target position (Drag and drop).

### 3.3 Entering the configuration parameters

#### 3.3-1 Access to the parameter configuration screen

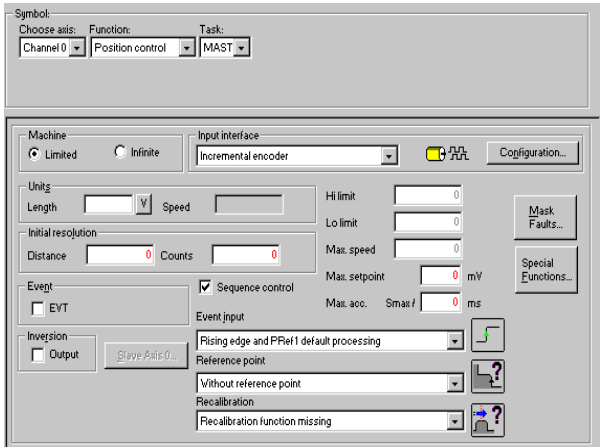
Select and confirm (or double-click) the position containing the declared axis control module.



Using the configuration fields, select :

- the channel to be controlled,
- the position control function,
- the MAST or FAST task in which the channel is used.

The lower part of the screen then displays the configuration parameters.



To display the whole configuration parameters zone select the **View/Module Zone** and **View/Channel Zone** commands (to restore these zones, use the same commands).

**Note :** The limits for each parameter are shown in the status bar.

### 3.3-2 Type of axis (machine)

Enables the configuration of the type of axis managing the channel :

- **limited machine** : the position measurement evolves between two values defined by the soft stops
  - **infinite machine**: the position measurement evolves between 0 and the Modulo
- This selection should be confirmed **before** the type of encoder is selected.

### 3.3-3 Type of encoder

These fields relate to the physical input interface for up/down counting.

#### For TSXCAY•1

The **type of encoder** zone is used to access the parameters given below.

#### For TSXCAY•2 and 33

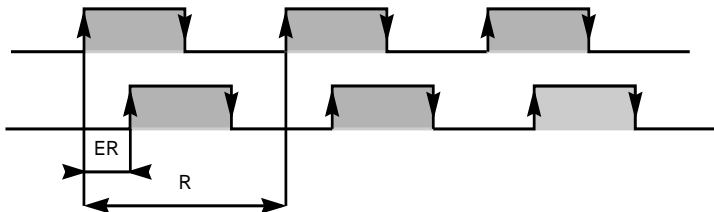
The **Configuration** button is used to access the screens for selecting the type of encoder. This screen is called **Input interface details**. The input interface field is used to determine the type of encoder : incremental or absolute.

#### Incremental encoder parameters :

- **Incremental encoder**
- **Measurement inversion** with or without (by default)
- **Multiplication by 1 or by 4** : (x1 by default)
  - **x4** with multiplication by 4 of the encoder signals
  - **x1** without multiplication by 4 (default selection).

Multiplying by 4 enables :

- 4 times greater precision to be achieved when using a given encoder
- or an encoder with a resolution which is 4 times lower to be used to obtain a given resolution



ER, which corresponds to the resolution obtained multiplied by 4, is also known as the equivalent resolution.

- SSI transmission multi-turn **absolute encoder** : for TSX CAY •1 modules  
When this encoder is selected the zone shown below is enabled, and is used to enter the characteristics of the SSI frame used by the encoder. The "SSI frame:" field displays the composition of the frame schematically.

**Encoder Type**

Incremental  **Absolute**

**Direct Off.**  Assisted Off.

**Binary**  Gray

**Header Bits** [ 4 ]

**Data Bits** [ 16 ]

**Status Bits** [ 3 ]

**Error** Pos. [ 1 ]

**Parity**  **Even**  Odd

**SSI Frame:**  
xxxx x16x ExxxP

① ② ③ ④

- Possible choices :
- Direct offset or assisted offset (default is direct offset)
  - Gray code or binary code (default is binary code)
  - no. of non-significant frame header bits ① : N min = 0, N max = 4 (default = 0) (1)
  - no. of data bits ② : N min =16, N max =25, (default = 16) (1)
  - no. of status bits ③ : N min = 0, N max = 3 (default = 0) (1),
  - choosing a number of status bits other than 0 accesses the error bit and its positioning (position 1 to 3) in the status bits zone.
  - presence of parity bit (default is without parity bit) and the type of parity : Even or Odd④. (2)

- (1) this is chosen either by directly entering the number, or using the [▲▼] buttons.  
 (2) If "Odd" is selected, the module does not control the parity and the parity bit is then managed like a status bit.

- 2 3 • SSI transmission multi-turn **absolute encoder** : for TSX CAY •2 / 33 modules  
The **input interface** is used to enter the characteristics of the SSI frame used by the encoder. The "Frame:" field displays the composition of the frame schematically.

**Configure Input Interface**

Input interface: ABSOLUTE SSI ENCODER

Encoder:

Direct offset  Assisted offset

**Binary**  Gray code  Measurement inversion

SSI frame

Header

No. of header bits: [ 0 ]

Data

No. of encoder data bits: [ 12 ]

Frame: [ x12x ]

Status

No. of status bits: [ 0 ]

Error bit

Parity

Presence of parity bit

OK Cancel

### 2 3 Encoder field :

- Direct offset or assisted offset (default is direct offset)

For direct offset, the user provides the offset value in encoder points.

For assisted offset, the offset is calculated by the module from a position value provided by the user.

- Gray code or binary code (default is binary code).

- Measurement inversion :

This parameter defines the inversion of the incremental encoder measurement, that is the direction of evolution of the measurement corresponding to a direction of rotation of the encoder. This parameter cannot be accessed when using infinite machines.

This parameter requires certain operations to be performed. In the first instance, retain the default values (see adjustment section).


### SSI Frame field:

- Number of non significant frame **header** bits : N min = 0, N max = 4 (default is 0) (1)
- Number of **data** bits : N min = 12, N max = 25, (default is 12) (1)
- Number of **status** bits : N min = 0, N max = 3 (default is 0) (1)
- Choosing a number of status bits other than 0 accesses the error bit and its positioning (position 1 to 3) in the status bits zone.
- Presence of **parity** bit (default is without parity bit) and the type of parity : Even or Odd. (2)

### 2 3 • For an absolute encoder with parallel outputs:

An absolute encoder with parallel outputs can be connected via an ABE 7CP A11 conversion interface.

The configuration which must be entered corresponds to that for an SSI multi-turn absolute encoder.

- (1) This is chosen either by directly entering the number, or using the  buttons.
- (2) If "Odd" is selected, the module does not control the parity and the parity bit is then managed like a status bit.

3.3-4 Initial resolution

The resolution is the distance corresponding to an encoder increment. Since the value is not generally an integer, it is expressed in terms of the following ratio :

RESOL = Distance / No. of points.

- Distance = distance travelled by the moving part
- No. of points = number of encoder points corresponding to the distance travelled.

Limit values : 1 to 1 000 000

The resolution is calculated using these 2 parameters in a ratio of between 0.5 and 1000.

RESOL = Distance / No. of points

Example : for an incremental encoder with 512 points per revolution, if the distance travelled for one encoder revolution is 10 000 µm, the following must be entered (the selected length unit is µm) :

Distance : 10 000, No. of points : 512

The resolution is thus : 10 000/512 = 19.5 µm.


Notes

This resolution can be corrected in the adjustment screen. It is for this reason that it is called the initial resolution.

In the case of an incremental encoder with multiplication by 4, enter the distance corresponding to ER (see page 3/5).

3.3-5 Measurement units

This is used to select the physical units in which the speed and position measurements will be expressed.

The screen offers the following units by pressing the  button :

Unit of position	Unit of speed
µm	mm/min
mm	m/min
in.e-2 (10-2inch)	in.e+1/min (10 inch/min)
in.e-5 (10-5inch)	in.e-2/min (10-2 inch/min)

The user can select his own measurement units, and the length field can be used to enter up to 5 characters. Example : degre

However, the unit of position must be selected in such a way that the resolution value (ratio of distance / number of points) lies between 0.5 and 1000. The unit of speed will be calculated using the formula :

unit of speed = unit of position \* 1000/min

The unit of speed is not selected but calculated by the formula ; however, it is possible to alter its text.

Example : for an encoder supplying 500 points per revolution, the distance corresponding to 1 revolution is 2 mm, or 2000 µm. The resolution is expressed by the ratio 2000 / 500 (ie. in µm). The resulting unit of speed will be in mm/min.

### 3.3-6 Upper and lower limits

For a limited machine, the axis upper (LMAX) and lower (LMIN) limits correspond to the mechanical limits of the axis.  
 These limit values are themselves limited depending on the resolution value selected.

Type of encoder	Lower Limit	Upper Limit
<b>Incremental encoder</b>	$-16 \times 10^6 \times \text{RESOL} \rightarrow 0$	$0 \rightarrow 16 \times 10^6 \times \text{RESOL}$
<b>TSX CAY •2/33</b>	limited to $-10^8 \times 6$	limited to $6 \times 10^8$
<b>TSX CAY •1</b>	limited to $-10^8$	limited to $10^8$
<b>Absolute encoder (1)</b>	$-16 \times 10^6 \times \text{RESOL} \times 2^{n-25} \rightarrow 0$	$0 \rightarrow 16 \times 10^6 \times \text{RESOL} \times 2^{n-25}$
<b>TSX CAY •2/33</b>	limited to $-10^8 \times 6$	limited to $6 \times 10^8$
<b>TSX CAY •1</b>	limited to $-10^8$	limited to $10^8$

(1) n = number of encoder bits

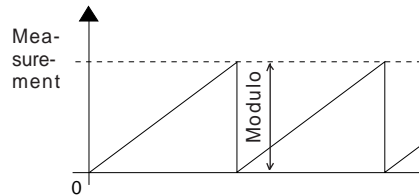
Upper Limit and Lower Limit must satisfy the condition :

- TSX CAY •1** Upper Limit - Lower Limit >  $2^{15} \times \text{RESOL}$
- TSX CAY •2/33** Upper Limit - Lower Limit >  $2^8 \times \text{RESOL}$

2 3

### 3.3-7 Modulo

For an infinite machine, the position measurement varies in the area **[0, modulo]**.



The value of the modulo is entered in encoder points in the **Max modulo** parameter and its equivalent in user units is displayed in the following field.

- Max modulo** defines the maximum permissible limit for the **Modulo** adjustable parameter. This field displays the modulo value entered by the user on screen.

The possible modulo values and associated parameters depend on the type of encoder :

- For an absolute encoder :**  
 Because the modulo is defined by the number of data bits in the SSI encoder frame, it is always a power of 2. The corresponding power of 2 is therefore entered, rather than the number of bits in the modulo.

**Example :** entering the value 12 corresponds to a max modulo of 4096.

The limit values of max modulo are also limited depending on the value of the selected resolution.

Type of encoder	Limits
<b>Incremental encoder</b>	$1000 \rightarrow 6.10^8 / \text{RESOL}$ limited to $16 \times 10^6$
<b>Absolute encoder</b>	n : 12 $\rightarrow$ 23 provided $2^n \times \text{RESOL} < 6 \times 10^8$

### 3.3-8 Maximum speed

The maximum speed VMAX should be such that the resulting frequency satisfies the following condition.

$$1.8 \text{ kHz} < F_{MAX} < 900 \text{ kHz}$$

where  $F_{MAX} = V_{MAX} \times m / \text{RESOL}$

- m = 2 with an incremental encoder x 1 or an absolute encoder
- m = 4 with an incremental encoder x 4

This condition is conveyed as the value of parameter VMAX by:

$$108 \times \text{RESOL}/m < V_{MAX} < 54,000 \times \text{RESOL}/m$$

subject to the following limits :  
 $270 < V_{MAX} < 270,000$

VMAX and RESOL are expressed in the units of the configuration screen, ie :  
RESOL in  $\mu\text{m}$  and VMAX in mm/min, RESOL in mm and VMAX in m/min, etc.

**Note :**

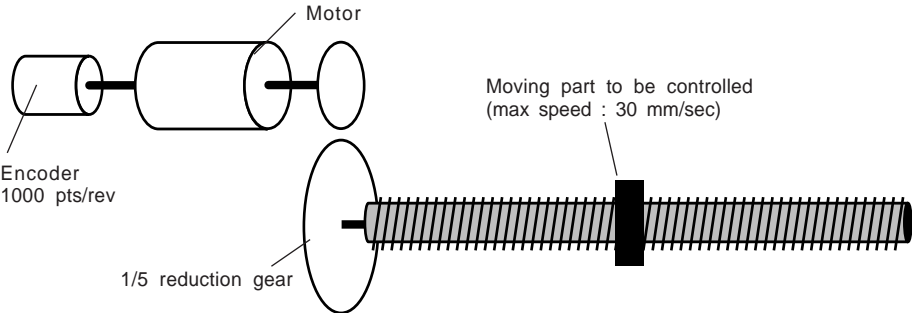
Irrespective of the speed programmed in the instructions, the module authorizes a speed equal to VMAX+10% when transients occur in order to absorb the deviation.

### 3.3-9 Maximum setpoint

The maximum setpoint UMAX is the voltage which must be applied to the input of the speed drive to obtain a speed equal to the maximum speed.

In the absence of any speed restrictions, the speed drive should be set to obtain maximum speed at a voltage as close as possible to (but below) 9 V.  
Limiting the voltage to 9 V means that during transient states there is sufficient reserve to allow an overvoltage : if there are no mechanical constraints, or restrictions imposed by the maximum acceptable frequency, select the following value : Maximum setpoint = 9 000 mV

**Example :** To control an axis with the following characteristics :





The maximum desirable linear speed is 30 mm / sec, or 1800 mm / min. The screw pitch is 5 mm.

The axis is controlled by a motor which can run at 3000 rpm driving a ballscrew via a 1/5 reduction gear. The encoder is on the motor shaft. It should be noted that an incremental encoder (without multiplication by 4) is being used.

The RESOL parameter (distance travelled by the moving part between 2 encoder increments) is equal to :  $N_e \times \text{Pitch}/N = 1/5 \times 5/1000 = 1 \mu\text{m}$

- The maximum operating speed parameter is 1800 mm / min.
- The maximum setpoint parameter is the voltage value which will enable maximum speed to be obtained. Taking into account the reduction ratio (1/5) and the screw pitch (5 mm), the maximum linear speed (1800mm/min) corresponds to a motor rotation speed of 1800 rpm. If the speed drive is adjusted to obtain a speed of 3000 rpm with an input voltage of 10 V, the voltage which will correspond to 1800 rpm (maximum setpoint is 6 000 mV).

**It is essential that the coherence of the RESOL, maximum speed and maximum setpoint parameters is maintained, otherwise incoherent servo loop behavior will result.**

---

### 3.3-10 Event

Can be used to associate an event-triggered task with the channel, and to set its number (0 to 63).

---

### 3.3-11 Inversion

This parameter can be used to avoid rewiring the analog output when the axis movement is in the opposite direction to that required.

The "Measurement inversion" parameter defines the inversion of the incremental encoder measurement, that is the direction of evolution of the measurement corresponding to a direction of rotation of the encoder.

For TSX CAY •2/33 modules, this parameter is located in the "Input interface details" dialog box.

These parameters require certain operations to be performed. In the first instance, retain the default values, then refer to the adjustment section.

### 3.3-12 Sequence check

The "sequence check" parameter is used to define the procedure to be followed if a movement without stop (G01, G11, G30) is not followed by a movement command.

- If the sequence check is enabled (default value), G01, G11 and G30 type movements not followed by a movement command are stopped (stop equivalent to the STOP command) and a command failure is generated.

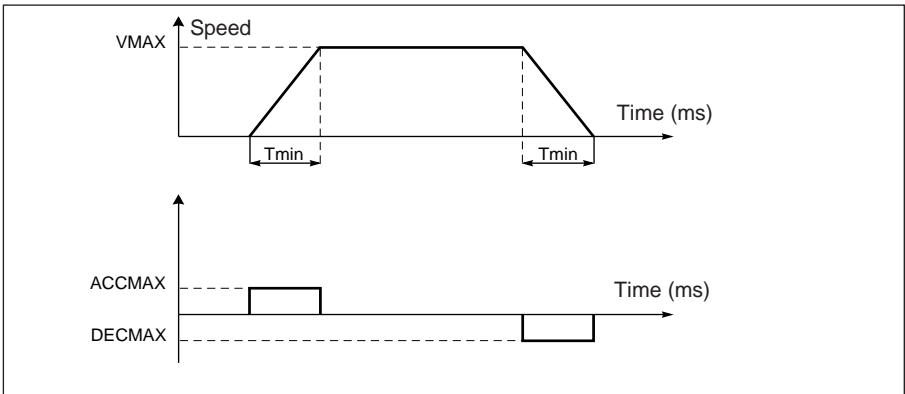
In this case, movements without stop cannot be sequenced by synchronization on the DONE bit (see 4.3-5 sequence of movement commands).

If the command following a G01, G11 or G30 type movement is type G05, G07 or G62, the stop is only triggered if the second command is not followed by a movement command.

- If the sequence check is not enabled, G01, G11 and G30 type movements not followed by a movement command continue at their target speed.  
The sequence check is only available on modules from version V1.7, providing that these have a V1.7 or later version of PL7 Junior.

### 3.3-13 Maximum acceleration (and deceleration)

This is defined by the minimum time (in ms) taken to change from zero speed to speed VMAX.



**Limits :** 16 to 10 000 ms

**3.3-14 Position follower of axis 0**

This zone is used to activate the "Follower movement of the position of another axis" function (see section 3.4-2) by checking the : **Activation** box and selecting, depending on the type of TSX CAY module :

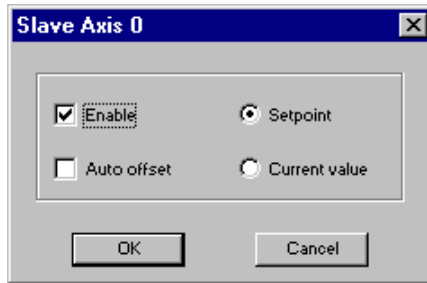
1



- the setpoint of the slave axis : Setpoint or Measurement of the master axis (axis 0)
- the ratio determining the setpoint value of the slave axis :

Setpoint of the slave axis = Ratio x Setpoint or Measurement of the master axis (Ratio is between 0.1 and 10, each of the fields comprising the Ratio parameter is between 1 and 1000).

2



- the setpoint of the slave axis : Setpoint or Measurement of the master axis (axis 0)
- the offset between master and slave : by teaching the slave axis position (select "Offset auto") or using an adjustable parameter, see section 5.3-7.

Setpoint of the slave axis = Ratio x Setpoint or Measurement of the master axis + Offset. The ratio and the offset are adjustment parameters, see section 5.3-7.

**Note** : the position follower function (slave movement) is not active for axis 0 which can only be master.

**3.3-15 Reflex inputs**





- This defines the type of event to be detected on the reflex input of the axis control module for instructions **05**, **10** and **11**.
- It also defines the type of event used for the position memorization function. This function memorizes one or two positions (PREF1 and PREF2).


Possibilities	Memorization	Icon (1)
---------------	--------------	----------

**Applications which do not require length measurement**

Rising edge and PREF1	PREF1	
Falling edge and PREF1	PREF1	

**Applications which require length measurements**

Rising edge and PREF1, then rising edge and PREF2	PREF1 PREF2	
Rising edge and PREF1, then falling edge and PREF2	PREF1 PREF2	
Falling edge and PREF1, then falling edge and PREF2	PREF1 PREF2	
Falling edge and PREF1, then rising edge and PREF2	PREF1 PREF2	









(1) the icon illustrates when the memorization takes place. For example :  position PREF1 is detected on the first rising edge of the reflex input, position PREF2 is detected on the second rising edge of the reflex input.

### 3.3-16 Reference point

An incremental encoder does not provide a position measurement, only a number of pulses proportional to the distance travelled. So that the distance can be converted into a position, a known position must be assigned to a particular point on the axis (0 is generally selected). This operation is known as setting the reference point. An axis on which this has been performed is said to be referenced.

The reference point defines the type and direction of the reference point (only when the position is measured using an incremental encoder).

The type is defined according to how the two reference detection inputs : zero marker input and cam input, are used.

Possibilities	Approach speed (1)	RP speed	Icon
<b>short cam (*)</b> and zero marker, + direction	F	F	
<b>short cam (*)</b> and zero marker, - direction	F	F/8	
<b>short cam (*)</b> , + direction	F	F	
<b>short cam (*)</b> , - direction	F	F/8	
<b>long cam</b> as travel limit & zero marker, + dir.	F	F/8	
<b>long cam</b> as travel limit & zero marker, - dir.	F	F/8	
<b>long cam</b> as travel limit, + direction	F	F/8	
<b>long cam</b> as travel limit, - direction	F	F/8	

(1) F is the speed programmed in the instruction in automatic mode or speed FMAN (defined in the adjustment screen) in manual mode. This speed can be corrected by the CMV coefficient.

(\*) Only **short cam** reference points can be used when the machine is infinite.

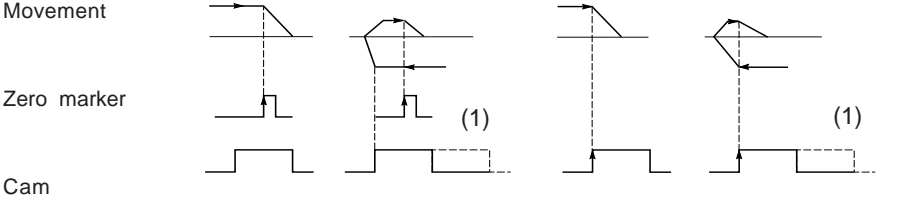
The reference point command is implemented by :

- instruction code **14**, reference point in automatic mode,
- the manual setpoint command SETRP.

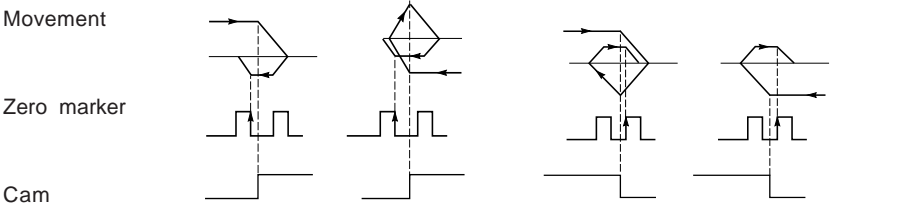
**Note** : There is also a "forced reference point" function (G62 in automatic and RP\_HERE in manual mode). It forces the position to the specified value. This action does not initiate a movement and thus does not take the selected reference point into account.

Detailed description of each reference point set up

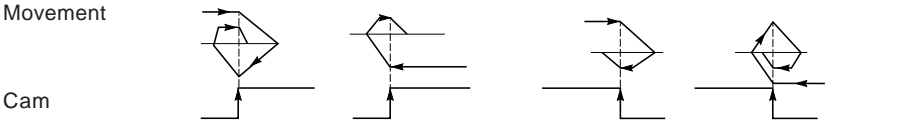
Type	Short cam/zero marker		Short cam only	
	+ direction	- direction (1)	+ direction	- direction (1)



Type	Long cam as travel limit/zero marker			
	+ direction (2)		- direction (2)	
	Start off cam	Start on cam	Start on cam	Start off cam



Type	Long cam as travel limit		- direction (2)	
	+ direction (2)	Start off cam	Start on cam	Start off cam



(1) Or start on cam  
 (2) Defines where the cam is located on the machine

**2 3 Check presence of zero marker on cam**



The short cam reference point checks for the presence of the zero marker along the cam.

When setting a + direction or - direction short cam reference point with a zero marker, if the entire short cam is traversed without the zero marker being detected, when leaving the cam the axis stops and an error is displayed.

The axis is in an unreferenced state.

**3.3-17 Recalibration**

This function can be used to compensate for a slip in the measurement. It applies to incremental encoders. Each time a moving part passes in front of the detector, the measurement is "recalibrated" to the specified value.

Possibilities	Recalibration function	Icon
No recalibration function	inactive	
Recalibration and fault when threshold crossed function	active	

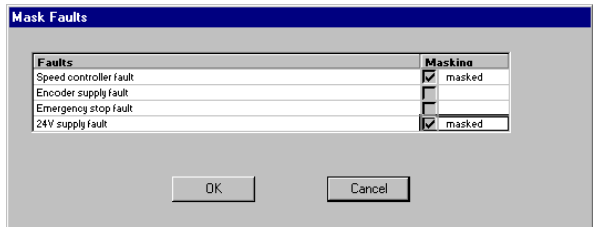
**2 3 3.3-18 Fault masking**

The "Fault masking" window can be used to mask the feedback of certain external faults during a power supply failure.

Four of the eight external faults can be masked for one channel.

These faults are :


- speed drive fault
- encoder supply fault
- 24 V supply fault
- emergency stop fault



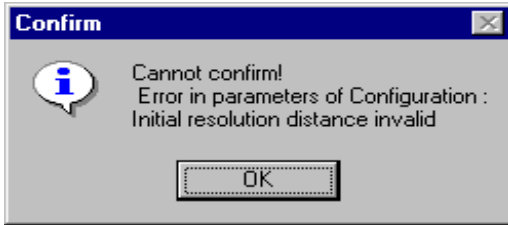
**3.3-19 Special functions**

Reserved.

### 3.4 Confirming the configuration parameters

When all the configuration parameters have been entered, confirm the configuration obtained using the **Edit/Confirm** command or select the  icon.

If one or more of the parameter values are not within permitted limits, an error message appears indicating the parameter concerned.



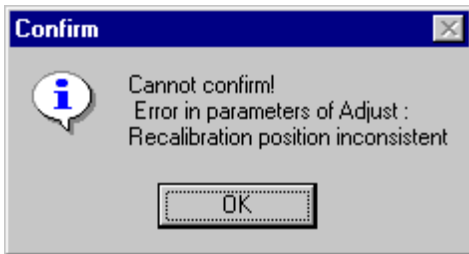
Correct the parameter then confirm.

**Note**

Incorrect parameters are displayed in red. Grayed out parameters cannot be entered because they depend on incorrect parameters (for example : an incorrect resolution prevents the minimum and maximum limits being entered).

**Important**

- The adjustment parameters are initialized as soon as there is a first request to confirm the configuration.  
It is therefore possible that following modifications to the configuration values, the adjustment parameters will no longer be correct. In this case a message indicates the parameter involved :



Access the adjustment parameters screen, correct the parameter, then confirm.

- **The configuration parameters are taken into account :**
  - when each of the configuration and adjustment parameters is correct,
  - when they are confirmed in the basic screen of the configuration editor.



## 4.1 Programming principle

Each channel of the axis control module is programmed in the following way :

- using the **SMOVE function** for movements in automatic mode.
- using **bit** (%I and %Q) and **word** (%IW, %QW and %MW) **objects** associated with the module to :
  - select the operating modes,
  - control the movements (except for automatic mode),
  - check the operating status of the module and the axis.

for more information on bit and word objects see section 13.

### Note

Bit and word objects can be accessed via their address or their symbol. These symbols must be entered in the variables editor. Symbol names are given in this manual for each of the objects.

## 4.2 Operating modes

Each axis control channel can be used in 4 modes :

- **Automatic (AUTO)** : movement commands controlled by the SMOVE.... functions are executed in this mode,
- **Manual (MAN)** : this mode enables the user to visually control the moving part from the front panel or from a man-machine interface terminal. The commands can be accessed via the output bits %Q.
- **Direct drive (DIRDRIVE)** : the output acts as a digital/analog converter, and the servo loop is not used. This mode is used to analyze the behavior of the axis independently of the servo loop during adjustment.
- **Measurement (OFF)** : in this mode the channel has no control over the moving part. It only feeds back position and current speed data. This mode is forced at start-up if the axis is configured and there is no fault.

The mode is selected using word `MODE_SEL %QWxy.i.0` (or using the debug screen selector :

Value

0	<b>OFF</b>	measurement mode, inhibition of the analog output,
1	<b>DIRDRIVE</b>	direct drive mode,
2	<b>MAN</b>	manual mode,
3	<b>AUTO</b>	automatic mode.

**Note** : for any other value of %QWxy.i.0, OFF mode is selected.

Changing mode while a movement is in progress (bit `DONE %Ixy.i.1` at 1) stops the moving part. When the moving part is completely stationary (bit `NOMOTION %Ixy.i.8` at 1), the new mode is activated.

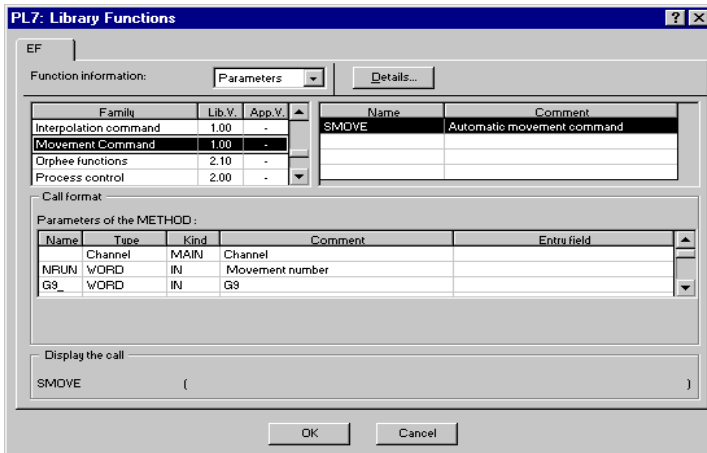
**Note** : only commands concerning the current mode are examined. Other commands are ignored (except when an SMOVE function is executed in manual mode).

## 4.3 Programming in automatic mode : SMOVE function

### 4.3-1 Programming an SMOVE function

SMOVE functions can be programmed in any program module in Ladder language (using an operation block), in Instruction list language (in square brackets) or in structured text language. The syntax is the same in all cases.

The function can be entered directly, or via the "Function Call" assisted entry screen.



### Assisted entry

In the selected program editor :

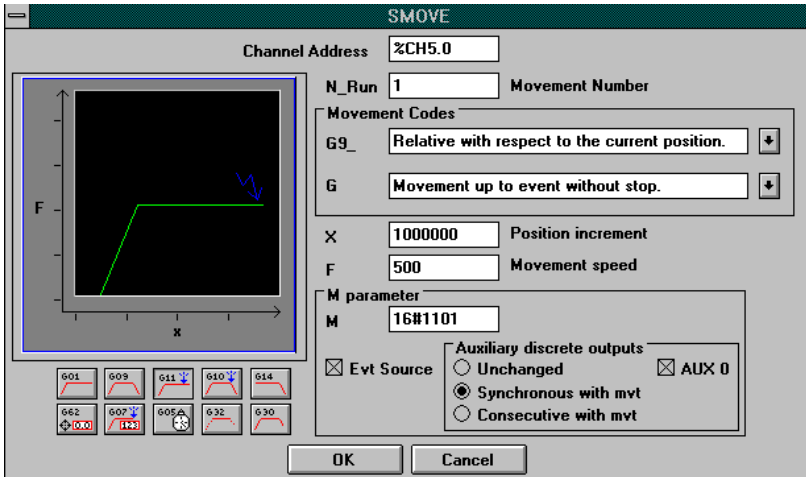
- 1 Press keys **SHIFT+ F8** simultaneously or click on the **F(...)** icon. The **Function Call** window appears.
- 2 Select the "Parameter" option.
- 3 Select the **Motion Control** family in the library.
- 4 Select the **SMOVE** function.
- 5 Press the **List...** button and fill in the various fields provided (see description on the next page) or enter the variables of the function directly in the parameter entry zone.
- 6 Confirm with **OK** or **ENTER**. The function appears.

### 4.3-2 Entering the parameters of the SMOVE function

A movement command is programmed by an SMOVE function, with the following syntax :

```
SMOVE %CHxy.i(N_Run,G9_,G,X,F,M)
```

The SMOVE function **List** screen provides assisted entry for each of the fields.



where

**%CHxy.i** = **Channel address** of the axis control module in the PLC configuration.  
**x** = rack no.  
**y** = position of the module in the rack  
**i** = channel number (0 to 1 for TSX CAY 2• modules or 0 to 3 for TSX CAY 4• modules and 0 to 2 for the TSX CAY 33)

**N\_Run** = **Movement identifier** from 0 to 32767. Number identifying the movement performed by the SMOVE function. In debug mode it identifies the current movement.

## Movement codes

**G9\_** = type of movement

**90 absolute** movement

**91 relative** movement **with respect to the current position**

**98 relative** movement **with respect to the memorized position PREF1** (position PREF1 is memorized using instruction code G07)

**60 absolute** movement in an imposed direction, (infinite machine only)

**68 relative** movement **with respect to PRef** in an imposed direction, (infinite machine only)

Select the type of movement using the scroll button to the right of field **G9\_** or enter the code directly during a direct entry operation (without going to the List screen).

**G** = Instruction code,

**09** : Move to position and stop



**01** : Move to position without stopping



**32** : Prepare machining command



**30** : Simple machining



**10** : Move until an event is detected and stop



**11** : Move until an event is detected without stopping



**14** : Reference point



**62** : Forced reference point



**05** : Await an event



**07** : Memorize the position when an event occurs



**21** : Unlimited movement with reference point on the fly

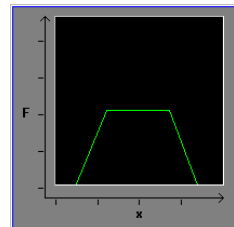
**04** : Stop a movement



2 3

Select the instruction code using the scroll button to the right of field **G**, or press the corresponding icon, or enter the code directly during a direct entry operation (without going to the List screen).

In the List screen : a graphic representing the selected movement is displayed (eg : code 09).



**X** = coordinates the position to be reached or to which the moving part is to move (in the case of moving without stopping).

This position can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this word can be indexed).

The unit in which these values are expressed is defined by the configuration parameter **Length Units** (this parameter is set in the configuration screen) eg :  $\mu\text{m}$ ,

**Note** : In the case of instructions G14 and G62 this value represents the reference point value. For instructions G07 and G05 see the detailed description.

**F** = speed of movement of the moving part. This speed can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this word can be indexed).

The unit of speed depends on the selected unit of position :  $\text{Speed} = u \times 1000/\text{min}$  where  $u$  = selected unit of a length

**Example** : if the selected unit of a length is  $\mu\text{m}$ ,  
The unit of speed is :  $\mu\text{m} \times 1000/\text{min} \rightarrow \text{mm}/\text{min}$

**Note** : for instructions G07 and G05 see the detailed description.

**M** = Word coded on 4 four-bit bytes (in hexadecimal) 16# 

--	--	--	--

- optional activation of the triggering of the application event processing for instructions : 10, 11, 05 and 07 (Four-bit byte no. 3 at 1 for activation)
- setting to 0 or 1 of the **auxiliary discrete output** associated with the channel for instructions 01,09,10 and 11

Four-bit byte no. 2 :

- 0 = **Unchanged** : no modification of the output
- 1 = **synchronized with mvt** : assignment of the output to the start of execution of the instruction
- 2 = **consecutive to mvt** : assignment of the output to the end of execution of the instruction

Four-bit byte no. 0 :

- 0 = set output to 0 (AUX 0 box not checked)
- 1 = set output to 1 (AUX 0 box checked)

- type of event awaited by instruction G05

Bit no.13 :

- 0 = awaiting a time-out or an event
- 1 = awaiting a modulo crossing number

Examples :

16#0101= no activation of triggering of application event processing and auxiliary output set to 1 when the SMOVE command is executed.

16#1020 = activation of triggering of application event processing and auxiliary output set to 1 at the end of execution of the SMOVE command

This is coded automatically in field **M** in the **List** screen using the check boxes in this screen.

### 4.3-3 Description of elementary movements

3 classes of movement can be programmed :

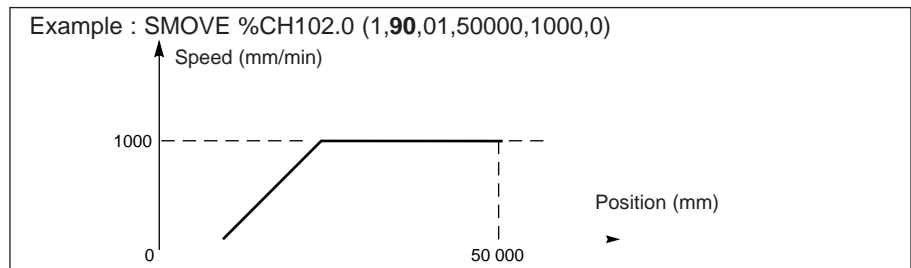
- move to a position (instruction codes 01 and 09),
- move until an event is detected (instruction codes 11 and 10),
- reference points (instruction code 14).

When programming these movements the user defines the position to be reached and the speed. The acceleration parameters (rectangular, triangular or trapezoid) are defined at configuration.

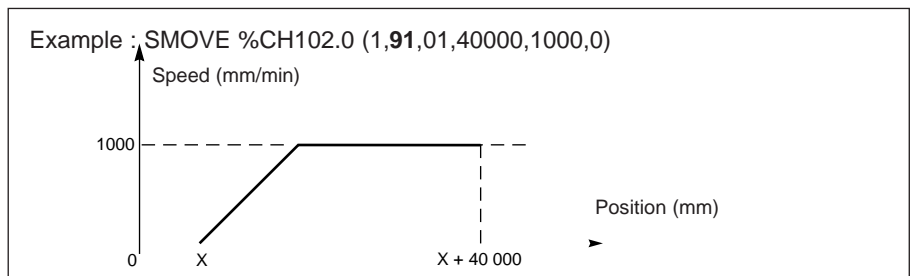
Movements can be :

#### Limited machine

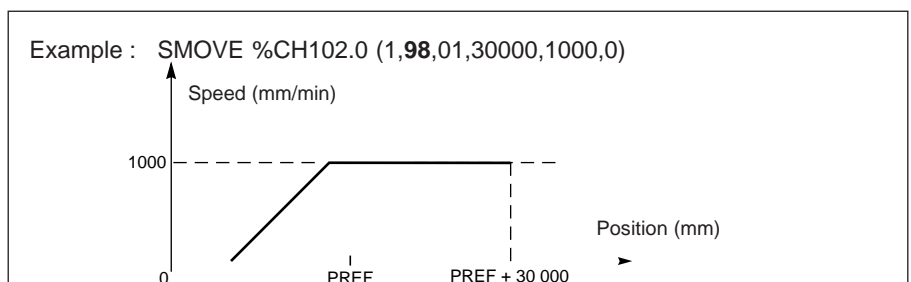
- absolute with respect to the machine reference **90**,



- relative with respect to the current position **91**,



- relative with respect to the memorized position PREF **98**

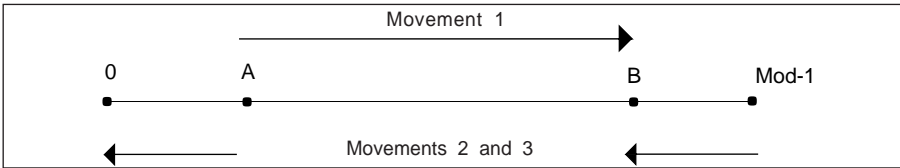


## Infinite machine

Whatever the current position and the target, it is always possible to reach the target position equally well in the + direction as in the - direction.

There are three possible ways to move from A to B :

- movement 1 increasing towards the position
- movement 2 decreasing away from the position
- movement 3, the shortest movement (the module determines the direction)

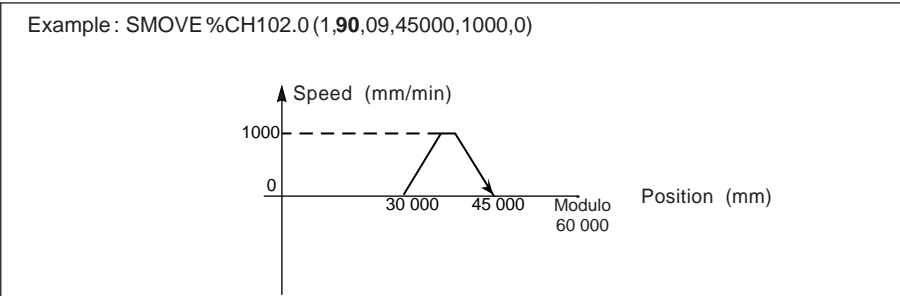


The sign of the speed is used to specify the desired direction of movement.

### Movements can be :

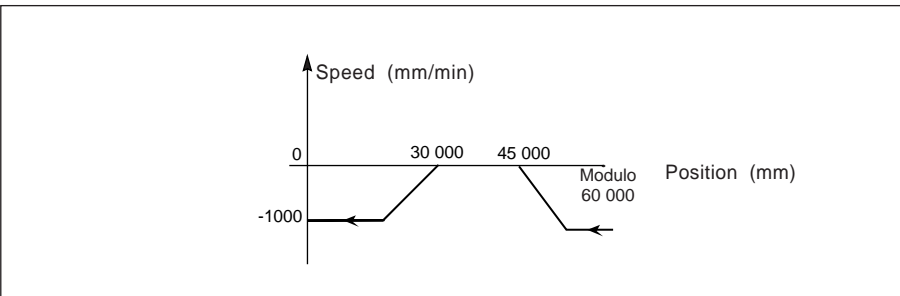
- as short as possible with respect to the machine reference **90**,

In this case the direction of movement is determined by the shortest path



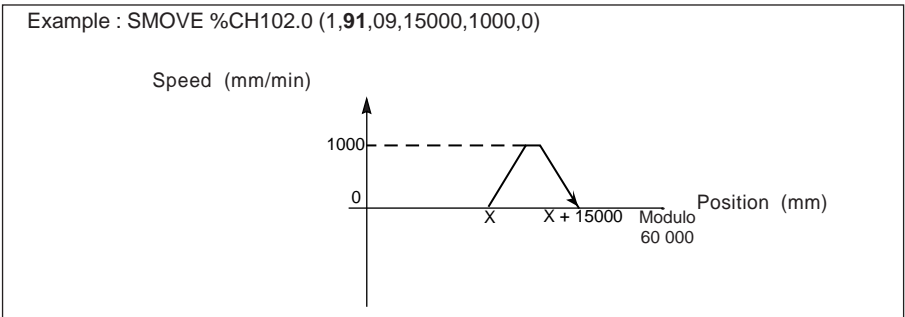
- in an imposed direction with respect to the machine reference **60**,

In this case the direction of movement is determined by the speed sign





- as short as possible with respect to the memorized position PRef **98**  
 Example : SMOVE %CH102.0 (1,**98**,09,45000,1000,0) targets the position (45 000 + PRef1).
- in an imposed direction with respect to the memorized position PRef1 **68**  
 Example : SMOVE %CH102.0 (1,**68**,09,45000,-1000,0) targets the position (45 000 + PRef1) by moving in the decreasing direction.  
 Example : SMOVE %CH102.0 (1,**68**,09,45000,1000,0) targets the position (45 000 + PRef1) by moving in the increasing direction.
- relative with respect to the current position **91**  
 In this case the direction of movement is determined by the sign of parameter X "position increment".



**Note**

The position targeted by G68 or G91 is calculated with respect to the modulo. In the examples, (1000 + PRF1) Mod ModuloValue and (X + 15000) Mod ModuloValue are targeted. "Mod" is the Modulo mathematical operator.

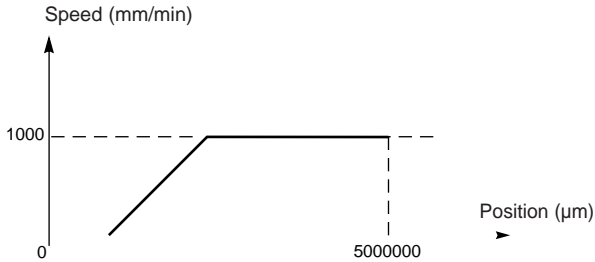
For example, if PRef1 = 40 000 and modulo = 60 000 : 45 000 + PRef1 is 25 000

## 4.3-4 Description of instructions

**Move to a position  
without stopping : instruction code 01**



Example 1 : SMOVE %CH102.0 (1,90,01,5000000,1000,0)

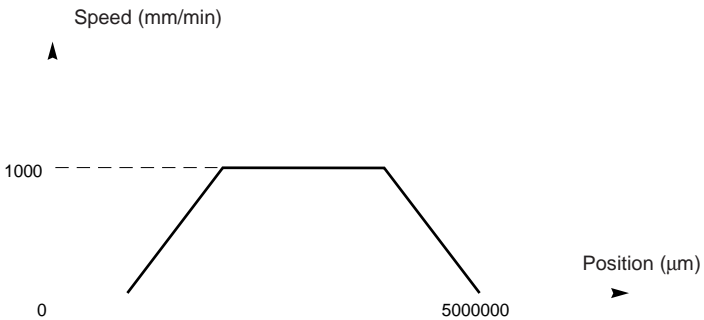


Note : if instruction 01 is not followed by any instruction, its behavior depends on the "sequence check" parameter defined during configuration (see 3.3-12).

**Move to a position  
and stop : instruction code 09**



Example 2 : SMOVE %CH102.0 (1,90,09,5000000,1000,0)



**Execution conditions for instructions 01 and 09** : see general conditions for execution.

**Move until an event**

**without stopping : instruction code 11**  
**and stop : instruction code 10**



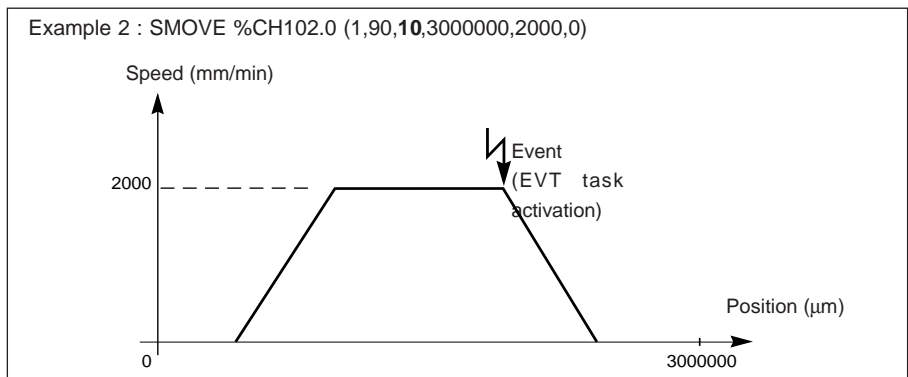
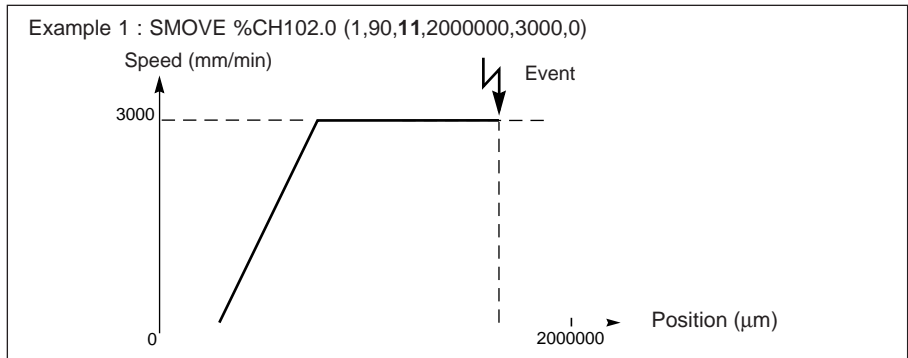
Instructions 11 and 10 are similar to 01 and 09 in that the command ends when the event is detected (or at the entered position if the event is not detected).

The event which is awaited can be :

- a rising or falling edge (depending on the selection made in the event field in the configuration screen) on the dedicated reflex input associated with the channel controlling the axis,
- a rising edge on bit EVENT\_UC (%Qxy.i.10) generated via the program.

The position parameter **MUST** be defined. If the event is not detected, the instruction ends when the requested target position is reached.

These instructions may activate an event-triggered task when an event is detected if bit 12 of parameter M is set to 1.



**Execution conditions for instructions 11 and 10 :**

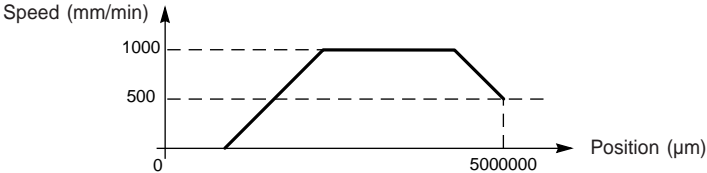
See general conditions for execution.

**Simple machining command**  
**preparation : instruction code 32**  
**execution: instruction code 30**



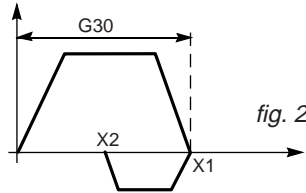
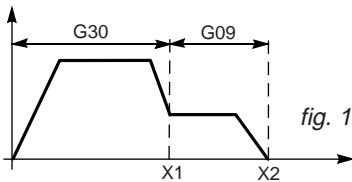
Instructions 32 and 30 are used to create a single composite machining profile  
 - of an approach speed indicated in instruction G32  
 - of a machining speed and a target position indicated in instruction G30

```
Example 1 : SMOVE %CH 102.0 (1,90,32, 0,1000,0)
            SMOVE %CH 102.0 (2,90,30, 5000000,500,0)
```

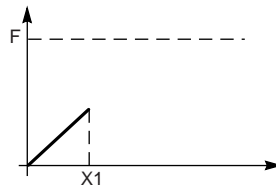
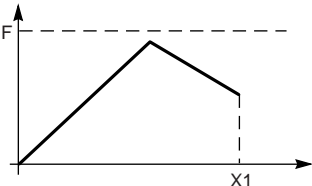


### Programming

- As command G32 is a preparation command only, it can be launched in the same PLC scan as command G30 without a check on the NEXT and DONE bits.
- As the approach speed is memorized and therefore does not change, relaunching command G32 serves no purpose. However, a G32 command must have been sent at least once before a G30 command is executed.
- Instruction G30 triggers a movement without stop whose behavior is identical to that of instruction G01 if it is not followed by a movement command.
- If the moving part is moving, instruction G30 must not cause any change in the direction of movement.
- Instruction G30 is usually following by instruction G09 (figure 1). If this sequence requires a change of direction, the moving part will stop and reverse until the G09 measurement is reached (figure 2).



- If the distance to be covered by instruction G30 does not enable the specified speed to be reached, movement will take place along one of the following trajectories :



**Reference point : instruction code 14**



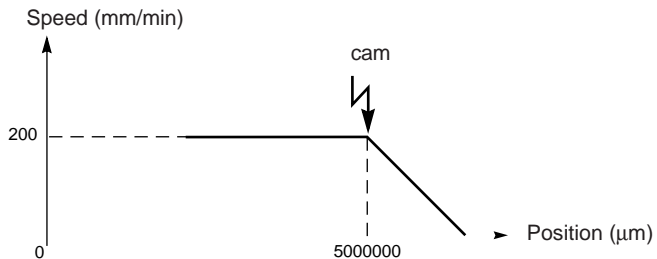
The position displayed corresponds to the coordinate to be loaded in the current value when the reference is detected.

The reference point event is detected on the cam input or the cam and zero marker inputs associated with the axis being controlled, depending on the type of reference point selected.

The type of reference point and the direction of movement are defined at configuration.

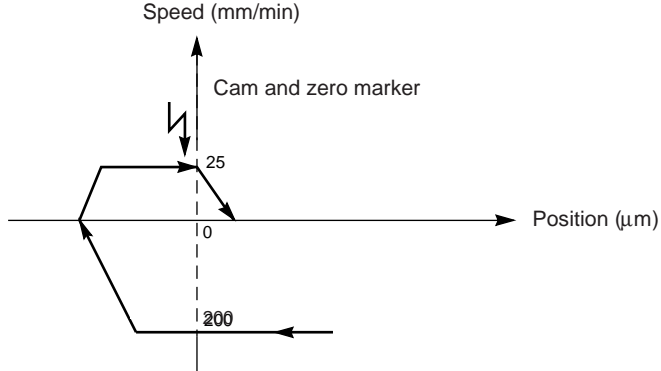
Example 1 : SMOVE %CH102.0 (1,90,14,5000000,200,0)

Type of reference point = short cam, + direction



Example 2 : SMOVE %CH102.0 (1,90,14,100,0)

Type of reference point = short cam, zero marker, - direction



Notes :

- the axis is not referenced at the start of the execution of the instruction.
- the type of movement must always be an absolute movement code 90.

**Execution conditions :**

See general conditions for execution.

## Reference point on the fly when an event occurs : instruction code 21



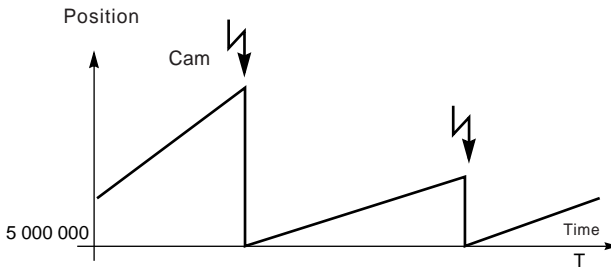
The position supplied by parameter X corresponds to the coordinate to be loaded as the current value when the reference point is detected.

The direction of movement is defined by the speed sign (the direction defined by the reference point type is not taken into account. This instruction never terminates naturally, a STOP command %Qxy.i.15 must be sent to terminate it).

The reference point event is detected on the cam input or the cam and zero marker inputs associated with the axis being controlled, depending on the type of reference point selected.

The type of reference point and the direction of movement are defined during configuration.

Example 1 : SMOVE %CH102.0(1,60,21,5000000,200,0)



### Execution conditions :

- Incremental encoder,
- Recalibration function inactive,
- Type of movement uses instruction code 60.

## Stop a movement : instruction code 04



This instruction is used to stop a G01, G30 and G11 type movement without stop as quickly as possible. It is the equivalent of a STOP command. There are no parameters associated with this instruction.

Example of use : stopping a G01 movement after a time delay of 10 s.

SMOVE %CH102.0 (1,91,01,100000,1500,16#0000)

SMOVE %CH102.0 (2,90,05,0,10000,16#0000)

SMOVE %CH102.0 (3,90,04,0,0,16#0000)

**Forced reference point : instruction code 62**

This command sets a forced reference point (without moving the moving part). The current position value is forced to the value entered in position parameter : X.

Example : SMOVE %CH102.0 (1,90,**62**,100000,0,0)

When this instruction is executed, the current position is forced to 100000.

**Note :**

Whatever the state of the axis : referenced or not referenced, this command is accepted and references the axis at the end of execution.

This command is only accepted if the moving part is stationary, NOMOTION bit =1 (%Ixy.i.8=1).

**Await an event : instruction code 05**

This instruction is used to await an event with a time period defined in parameter F in ms. If the event has not appeared within the time period, the await event command is deactivated. If parameter F is defined at 0, the waiting period is not limited.

For an infinite machine, G05 can also be used to await the crossing of a modulo number. The selection is determined by the value of bit 13 of code M :

- 0 await event
- 1 await modulo number

The event associated with the G05 command could be :

- a rising or falling edge (depending on the selection made in the event field in the configuration screen) on the dedicated reflex input associated with the channel controlling the axis,
- a rising edge on bit EXT\_EVT (%Qxy.i.10) generated by the program,
- a modulo crossing number (infinite machine).

Example : await 10 modulo crossings with activation of the event-triggered task.

SMOVE %CH102.0 (1,90,**05**,10,16#2000)

This instruction may activate an event-triggered task when an event is detected if bit 12 of parameter M is set to 1.

Bit TO\_G05 is set to 1 when the time period has elapsed with no detection of an event.

Example : a wait with a time period of 1.5 secs and with activation of the event-triggered task.

SMOVE %CH102.0 (1,90,**05**,0,1500,16#1000)

## Memorize current position when an event occurs : instruction code 07



After the execution of this instruction, when an event appears on the reflex event input of the axis control module, the current position is memorized.

Depending on the choice made at configuration (see section 3.3-14), and in position parameter X, it is possible to memorize one or two positions (PREF1 and PREF2) :

- if the without measurement option was chosen at configuration, only position PREF1 is memorized (parameter X must equal 1),
- if the with measurement option was chosen at configuration, if X= 1 the event processing will be activated when position PREF1 is memorized, if X=2 it is after positions PREF1 and PREF2 are memorized that event processing will be activated.

Type of event on event input	Timing diagram	Choice at configuration
Rising edge		Standard processing Rising edge and PREF1
Falling edge		Falling edge and PREF1
Rising edge		Rising edge and PREF1, then rising edge and PREF2
Falling edge		Falling edge and PREF1, then falling edge and PREF2
Rising and falling edge		Rising edge and PREF1, then falling edge and PREF2
Falling and rising edge		Falling edge and PREF1, then rising edge and PREF2

Event processing can be activated when the event is detected if bit 12 of parameter M is set to 1. The program carries on immediately to the next instruction.

Words %IDxy.i.9 (PREF1) and %IDxy.i.11 (PREF2) are only updated if an event task is triggered by the expected event.

The results of instruction G07 (measurement/event delay) are : **immediate** for an incremental encoder and **≤ 400 μs** for an absolute encoder.

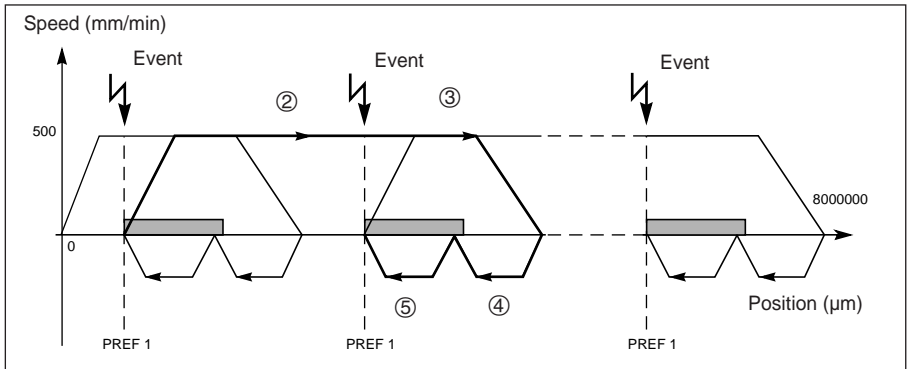


**Example of using an indexed position (repetitive movements) :**

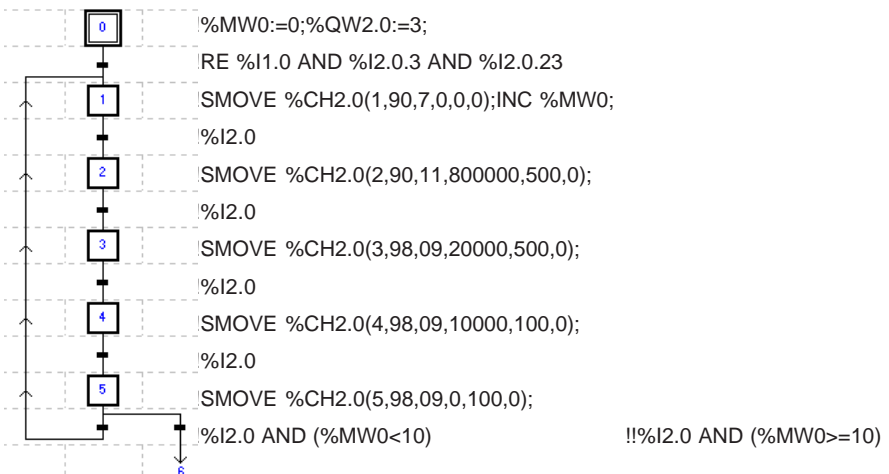
The example below gives a sequence of elementary movements to be performed 9 times :

- move until the edge of the item is detected ②,
- move to position 2000 with respect to the edge of the item ③,
- move to position 1000 with respect to the edge of the item ④,
- move to the edge of the item ⑤,

In this example it is assumed that the reference point has already been set and the moving part is in the start position.



**Note :** the sequence of elementary movements is represented in bold on the above chart. The numbers shown correspond to the program step numbers in the SMOVE function.



**Note :** All the actions must be programmed on activation.

### 4.3-5 Sequence of movement commands

A trajectory is created by programming a series of elementary movement instructions using the SMOVE function.

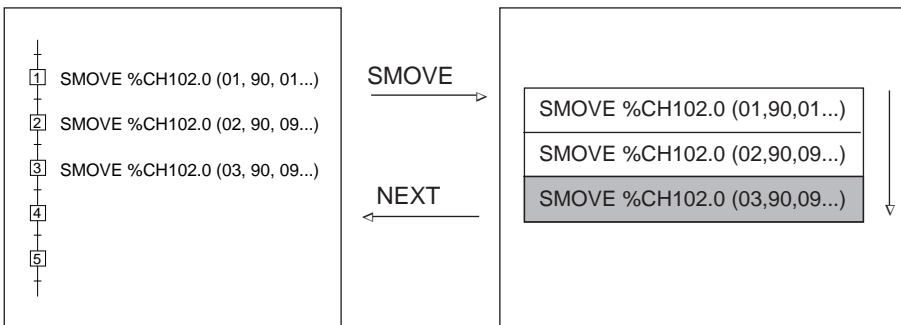
Each elementary command to execute an SMOVE function is performed once only, so the execution is programmed in :

- either Grafset : in a step on activation or on deactivation of that step,
- or structured text or Ladder language on the rising edge of a bit.

The execution report for the function is provided by the module using the NEXT and DONE bits (see the table on the next page).

The TSX CAY module has a mechanism for linking movement commands together into sequences.

Each axis of the TSX CAY module has a buffer memory which can receive 2 movement commands ahead of that which it is currently executing. Thus, when the current command has been executed, it goes directly to the first command in the buffer memory.



The link between 2 movement commands is established in the following way :

- immediately if the first movement does not include a stop,
- as soon as the moving part is in the target window or at the end of the time delay TSTOP defined in the stop control (parameter Adjustment screen) if the first movement includes a stop.

The execution time of the current instruction must be greater than the master task period so that the move from one command to the next is immediate.

**Note :**

A new command must only be transmitted to the module if the buffer memory associated with the axis to be controlled is not full.

**Bits associated with the sequencing mechanism**

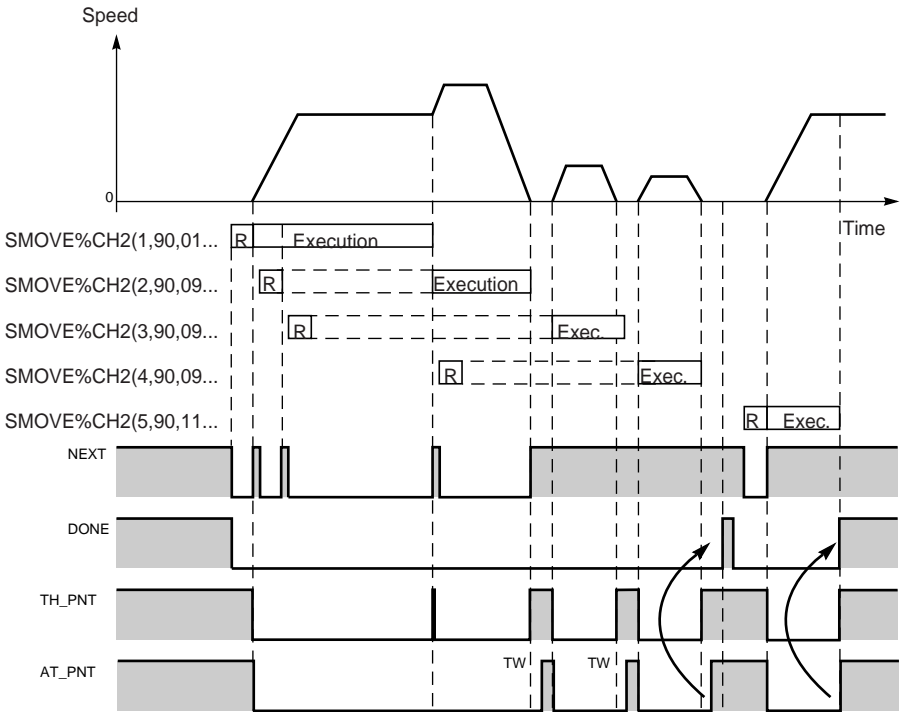
Addressing	Description
NEXT %lxy.i.0	Indicates to the user program that the module is ready to receive the next movement command.
DONE %lxy.i.1	Indicates the end of execution of the current command and that there is no new command in the buffer memory.
TH_PNT %lxy.i.10	Signals that the setpoint value has been reached.
AT_PNT %lxy.i.9	Signals that the moving part has reached the point intended by an INC command in manual mode or a movement command with stop in automatic mode. For a movement with stop this bit changes to 1 as soon as the moving part enters the target window. It is not set to 1 following a JOG command, reference point or STOP during a movement.

**Note**

Either the NEXT or the DONE bit must always be tested before an SMOVE command is executed, except in the case of command G32, which may be immediately followed by another command.

Word SYNC\_N\_RUN %lWxy.i.8 periodically provides the current step number in order to perform movement sequences.

## Example :



R = Read

For a movement with stop : DONE changes to 1 when NOMOTION changes to 1 and when the buffer memory is available.

For a movement without stop : DONE changes to 1 when TH\_POINT changes to 1 and when the buffer memory is available.

### Note

This diagram does not take the deviation into account.

## 4.4 Programming : other functions

### 4.4-1 Recalibration on the fly function

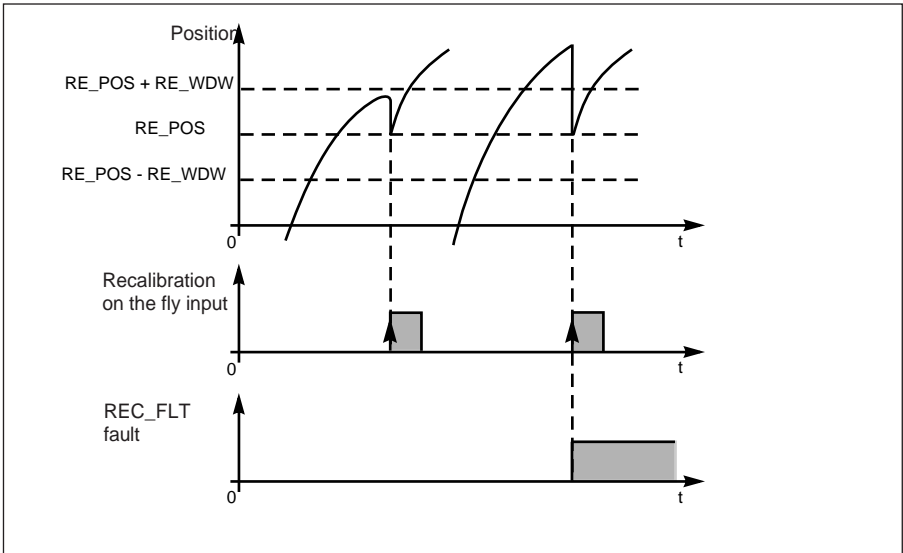
This function, available with an incremental encoder, updates the current position of the moving part each time the recalibration on the fly input detects a rising edge in forward direction or a falling edge in reverse direction. It is especially suitable for axes where the moving part has a tendency to slide so that the position value no longer reflects the actual position.

This function is enabled in the configuration screen.

When an event occurs, the axis control module presets the current value to the RE\_POS value and compares the current position with the recalibration value RE\_POS (%MDxy.i.43) defined in the adjustment screen (or by the program) :

- if the comparison indicates that the current value was outside the tolerances defined by RE\_WDW (%MDxy.i.51), a fault is signaled (bit REC\_FLT %MWxy.i.3:X12).

The moving part continues to move.



#### Execution condition :

This function is valid :

- if the axis is referenced,
- in manual, automatic or direct drive mode,
- in Drv\_Off mode.

2 3

#### Note

The value of the RE\_WDW parameter must be clearly lower than the value of the deviation error threshold DMAX1.

#### 4.4-2 Follower movement of another axis TSX CAY •1

With this function the position of one axis (called the slave axis) is governed by that of another axis (called the master axis) in the same module.

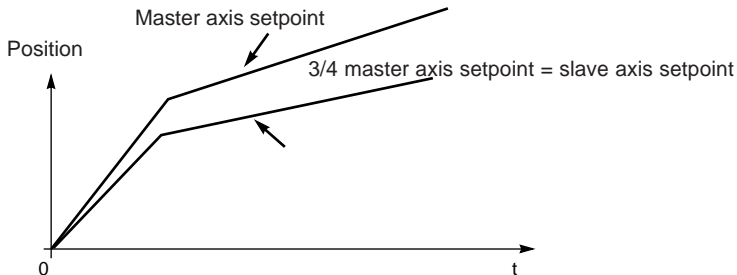
The master axis is always axis 0. A 2-axis module can have one master axis and one slave axis. A 4-axis module can have one master axis and up to 3 slave axes.

The position follower function is enabled in the configuration screen.

At programming level, the slave axis is governed by the master axis when the SLAVE bit (%Qxy.i.17) of this axis is set to 1. Bit %Ixy.i.36 indicates that the slave axis is operating correctly in tracking mode.

Feedback is provided either by the measured position or the position setpoint of the master axis (choice defined in the configuration screen). A ratio  $RATIO1/RATIO2$  is applied to obtain the final setpoint. These 2 parameters are defined in the configuration screen. In order to indicate that the slave axis correctly follows the master axis, the bit  $AT\_PNT$  (%Ixy.i.9) of the slave axis changes to 1 when its deviation is less than  $DMAX2$ .

Example : Ratio =  $3/4$ , with setpoint feedback of the master axis



#### Execution condition :

- the master axis is configured,
- the slave axis is referenced,
- no blocking fault is detected,
- the slave axis is in automatic mode,
- the master axis must be in automatic or manual mode.

If the calculated setpoint exceeds the soft limits of the axis, the moving part stops and the command fails.

#### Important :

To ensure that the position setpoint of the slave axis derived from the master axis is valid, it is necessary to ensure that a check is made that the slave is already in the Master x  $RATIO$  position before it is changed to slave mode.

Likewise, reference point type commands on the master axis should be avoided when there are slave axes (risk of a deviation error on the slave axis).

In follower mode,  $TH\_POINT$  and  $NEXT$  data is not managed, the  $PAUSE$  command is not active and modifications to  $CMV$  are not taken into account ( $CMV = 1000$ ).

#### 4.4-3 Follower movement of another axis TSX CAY •2

Using this function the position of one axis (called the slave axis) is governed by that of another axis (called the master axis) in the same module.

The master axis is always axis 0. A 2-axis module can have one master axis and one slave axis. A 4-axis module can have one master axis and up to 3 slave axes.

The position follower function is enabled in the configuration screen.

At programming level, the slave axis is governed by the master axis when the SLAVE bit (%Qxy.i.17) of this axis is set to 1. Bit IN\_SLAVE %Ixy.i.36 indicates that the slave axis is operating correctly in tracking mode.

Feedback is performed on the position measured or the position setpoint of the master axis (choice defined in the configuration screen). A ratio RATIO1/RATIO2 and an Offset are applied to obtain the final setpoint. These 3 parameters are defined in the adjustment screen.

The slave axis is linked to the master axis by the following relationship :

**SlavePositionSetpoint = MasterPosition x (Ratio1/Ratio2) + SlaveOff.**

In order to indicate that the slave axis is correctly following the master axis, bit AT\_PNT (%Ixy.i.9) of the slave axis changes to 1 when the slave has caught up with the master and remains close to it (its deviation is less than DMAX2) for more than TSTOP ms.

Differences between the TSX CAY •1 and TSX CAY 2 module for the follower function :

- the ratio can be modified from the application or using PL7 in adjust mode (the ratio is fixed during configuration for the TSX CAY •1 module)
- the offset enables the slave axis to be governed by the master axis whatever the position of the master axis. This enables object tracking type applications to be executed where an axis supporting a tool must be governed by a permanently running axis (conveyor belt) for transporting objects (applying glue, etc.).

The value of the offset can be modified from the application or the PL7 software in adjust mode. The TSX CAY •2 module provides an alignment or setting device to ensure a smooth transition to slave mode.

The function calculates the following offset :

**Slave\_position = Master\_position X Ratio + Offset.**

The "Automatic offset" configuration parameter is used to select the operating mode. The signed value of the ratio is between 0.01 and 100.

**Parameters associated with master/slave mode :**

- %MWxy.i.29 Ratio 1 and %MW xy.i.30 Ratio 2 which determine the value of the ratio between master and slave
- %MDxy.i.55 Slave Offset : value of the offset when the Automatic offset function has not been selected during configuration
- InternalSlaveOffset : value of the offset (calculated by the module and not accessible to the user) when the Automatic offset function has been selected during configuration.

### Execution conditions

- the master axis is configured in automatic or manual mode within the framework of a setpoint follower
- the slave axis is referenced and in automatic mode
- no blocking fault is detected
- the master axis must be referenced within the framework of a measurement follower (the master can be in any of the 4 modes)

If the calculated setpoint exceeds the soft stops of the axis, the moving part stops and the command fails.

$\text{Slave position setpoint} = \text{Master position}^{(1)} \times \text{Ratio1/Ratio2}$ <div style="display: flex; align-items: center; justify-content: center; margin-top: 10px;"> <span style="font-size: 2em; margin-right: 10px;">+</span> <div style="display: flex; flex-direction: column; align-items: center;"> <div style="margin-bottom: 5px;">↙ slave Offset</div> <div style="margin-bottom: 5px;">↘ Internal Slave Offset</div> </div> </div> <p style="margin-top: 10px;">(1) depending on configuration</p>
--

- **DRIVE\_OFF mode**

This mode allows a slave axis to be governed by master axis. In this case the slave is declared as a measurement follower.

- **Meaning of bit AT\_PNT and parameter DMAX2**

DMAX 2 defines the precision threshold.

This value is particularly useful in object tracking type applications where the slave axis passes through a catching up phase before meeting the condition :

$$(\text{Master\_pos} \times \text{Ratio} + \text{Offset}) - \text{DMAX2} \leq (\text{Slave\_pos}) \leq \text{Master\_pos} \times \text{Ratio} + \text{Offset} + \text{DMAX2}.$$

As soon as the condition is satisfied for a period of time at least equal to the value of the T\_STOP parameter, the AT\_PNT bit changes to 1 to indicate that the slave axis has "caught up".

### Important

To ensure that the position setpoint of the slave axis derived from the master axis is valid, it is necessary to check that the slave axis is already in the Master x RATIO position before it is changed to slave mode.

Likewise, reference point type commands on the master axis should be avoided when there are slave axes (risk of a deviation error on the slave axis).

In tracking mode, TH\_POINT and NEXT data is not managed, the PAUSE command is not active and modifications to CMV are not taken into account (CMV = 1000).



---

During configuration, the user can specify that the link between slave and master must be established with no movement.

In this case, the slave does not take the `SlaveOffset` parameter into account, it calculates an "InternalSlaveOffset" parameter (which is not communicated to the application) such as :

This prevents any movement of the slave, at the moment it becomes a slave, while the master is stationary.

### **Specific features**

These applications include :

- the presence of an infinite axis called the "master" which runs continuously
- the presence of an axis which is sometimes governed by the master and sometimes independent

#### 4.4-4 Follower movement of an external periodic setpoint

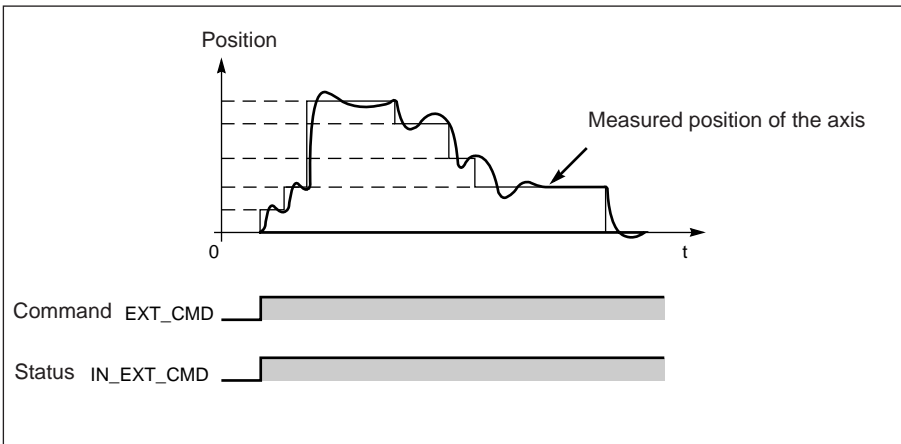
With this function the position of an axis is governed by a position written via the application program in double word PARAM (%QDxy.i.2).

This enables one axis to be governed by a pre-programmed trajectory.

It can also be used to govern an axis belonging to a module, but the update rate of the slave axis setpoint is equal to the period of the task in which the modules are managed.

This function is enabled by setting bit EXT\_CMD (%Qxy.i.18) to 1. A status bit indicates the activity of this function : bit IN\_EXT\_CMD (%Ixy.i.37).

#### Example :



For TSX CAY •2 and 33 modules, the discrete event input can be controlled in this mode.

#### Execution condition :

This function is valid if :

- the axis is referenced,
- no blocking fault is detected,
- position PARAM is within the soft limits

#### Note :

It is advisable to ensure that the axis is already at position PARAM before changing it to follower mode and to check that PARAM has a continuous and coherent evolution (risk of deviation error on the axis).

**4.4-5 Deferred "PAUSE" function**

The PAUSE (%Qxy.i.16) command suspends the sequence of movements. It only becomes active when the moving part is stationary, say at the end of a G09 or G10 instruction.

The next movement starts as soon as the PAUSE command is reset to 0. Bit ON\_PAUSE (%lxy.i.33) signals, when it is at 1, that the axis is in "PAUSE" state.

This function has 2 possible uses :

- block by block execution of the movement program,
- synchronization of the axes on an axis control module.

**Block by block execution of the movement program**

If the current instruction is an instruction with stop, activating the **Pause** command in the debug screen in automatic mode, or setting bit PAUSE (Qxy.i.16) to 1, causes the axis to change to waiting status after execution of the current instruction : the sequence of movements stops.

Movements without stop are stopped after they have been executed on reaching the soft stop.

It is therefore possible to execute movements block by block for debugging purposes by successively activating and deactivating the Pause command.

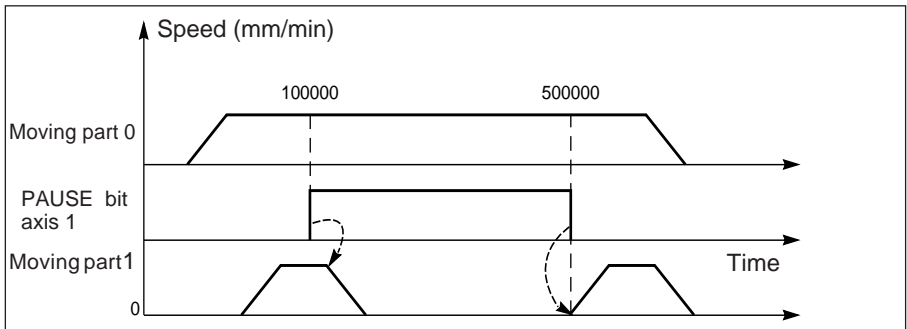
**Synchronization of several axes**

Setting bit PAUSE (%Qxy.i.16) to 1 via the program for each axis causes the axis to change to waiting status after execution of the current instruction.

When the PAUSE bit is reset to 0, the module continues to execute the instructions.

**Example :** Execution of the movement of moving part 1 is stopped when moving part 0 reaches position 100000. It is restarted as soon as moving part 0 reaches position 500000.

```
IF (%ID2.1>=100000) THEN SET %Q2.1.16;
. . . . .
IF (%ID2.1>=500000) THEN RESET %Q2.1.16;
```



**Note :**

The PAUSE command is only processed when AUTO mode is active and when the position follower functions are inactive.

#### 4.4-6 Step by step mode

This mode is used to execute a sequence of movements, stopping after each elementary instruction (step).

Movements without stop are thus transformed into movements with stop at the same position and speed. (Except for instruction G21 which never stops).  
In the case of command G30, the speed used is the approach speed.

This mode is activated by setting bit MOD\_STEP (%Qxy.i.19) to 1.

Bit ST\_IN\_STEP (%Ixy.i.39) indicates that the mode is active, ie. that the command in progress has been modified for execution in step by step mode.

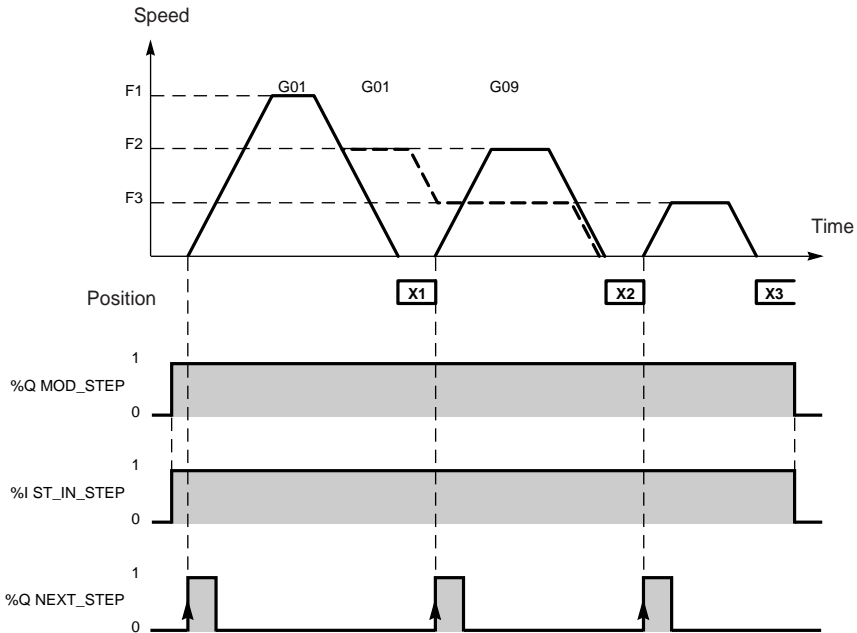
A rising edge on bit NEXT\_STEP (%Qxy.i.22) is used to launch the following step :

Example : execution in step by step mode of the following profile :

SMOVE (1,90,01,X1,F1,M)

SMOVE (2,90,01,X2,F2,M)

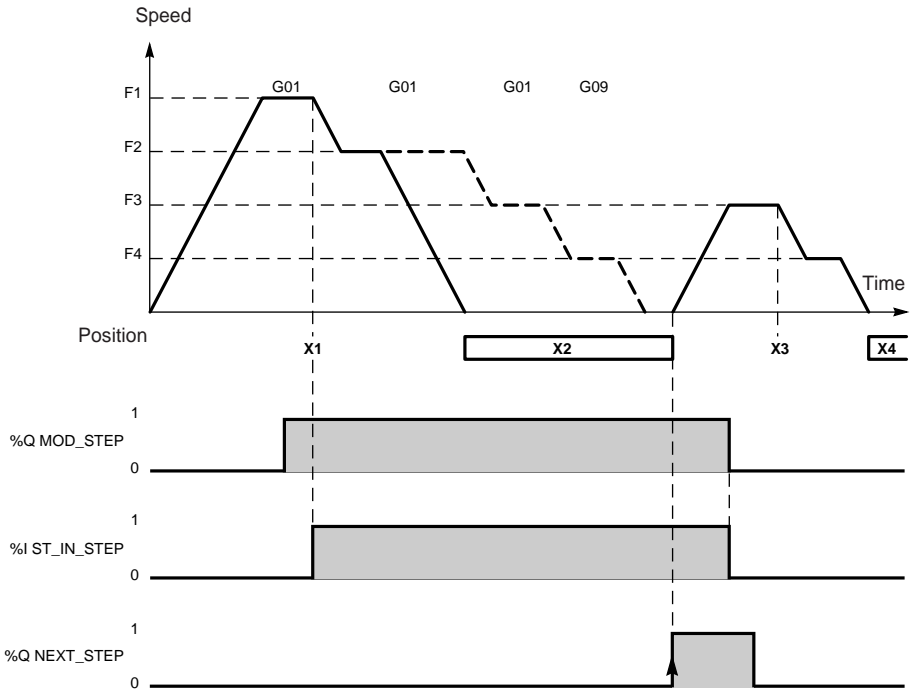
SMOVE (3,90,09,X3,F3,M)



If a movement is in progress when a request to change to step by step mode is made, the switch to this mode is made at the beginning of the next movement. However, exit from this mode is immediate, even if a movement is in progress.

Example : execution in step by step mode of the following profile :

```
SMOVE (1,90,01,X1,F1,M)
SMOVE (2,90,01,X2,F2,M)
SMOVE (3,90,01,X3,F3,M)
SMOVE (4,90,09,X4,F4,M)
```



However, if a request to exit this mode is made during deceleration corresponding to a transformed movement without stop, exit from the mode occurs only at the end of the movement.

**Notes :**

- Commands G05, G07 and G62 are executed in step by step mode
- Command G32 is not considered to be a step

#### 4.4-7 Immediate "PAUSE" function

This function is used to stop the moving part in automatic mode, while still ensuring that on the command to restart the movement the programmed trajectory is followed (with no risk of the command being refused).

This command is activated :

- via the program : by assigning the value 0 to the speed correction coefficient word CMV (%QWxy.i.1),
- via the debug screen : by assigning the value 0 to the speed correction coefficient parameter CMV.

It causes the moving part to stop according to the programmed deceleration. The pause status report is signaled by bit IM\_PAUSE (%Ixy.i.34).

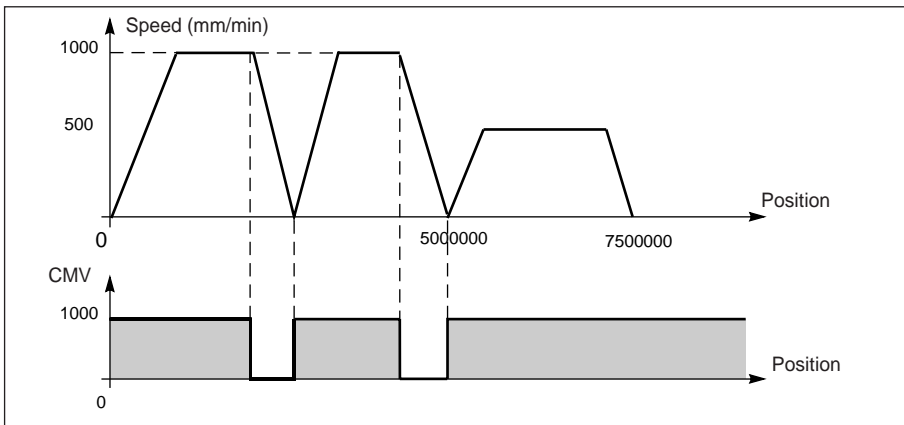
This command is deactivated :

- via the program : by reassigning the initial value (>0) to the speed correction coefficient word CMV,
- via the debug screen : by reassigning the initial value (>0) to the speed correction coefficient parameter CMV.

It causes the interrupted movement to restart at the speed corresponding to :  $F \cdot CMV / 1000$

```
Example : SMOVE %CH2.0 (1,90,10,5000000,1000,0);
          SMOVE %CH2.0 (2,90,09,7500000,500,0);
```

```
.....
IF RE %M10 THEN %MW100:=%QWxy.i.1;%QWxy.i.1:=0;
IF FE %M10 THEN %QWxy.i.1:=%MW100;
```



#### Notes :

- this command is deactivated at a STOP command or blocking fault,
- In the case of a movement without stop instruction if the target position has been reached when movement is stopped as a result of an immediate pause command, the current movement is terminated. In this case the trajectory is restarted with the next movement waiting in the stack.
- the immediate pause function has no effect if the current movement is governed by a position (slave axis or PARAM position follower).

### 4.4-8 Event processing

The channels of TSX CAY modules can activate an event-triggered task. For this the function must be enabled in the configuration screen by associating an event processing number with the channel.

#### Activating an event-triggered task

The following instructions start the transmission of an event which activates the event-triggered task :

- move until an event is detected, codes **10** and **11** : the application event processing is activated when the event is detected
- await an event, code **05** : the application event processing is activated at the end of the instruction
- memorize the current position when an event occurs, code **07** : the application event processing is activated at the end of the memorization of position PREF1 or position PREF2.
- cross the modulo for an infinite axis : application event processing is activated each time the modulo is crossed during a movement. Activation of event processing must be enabled by setting the VALIDEVTMOD parameter (%MWxy.i.62:X0) to 1.

Application event processing is activated if bit 12 of parameter M of the SMOVE function associated with the instruction is 1 (see section 5.3-2).

#### Variables which can be used by the event-triggered task

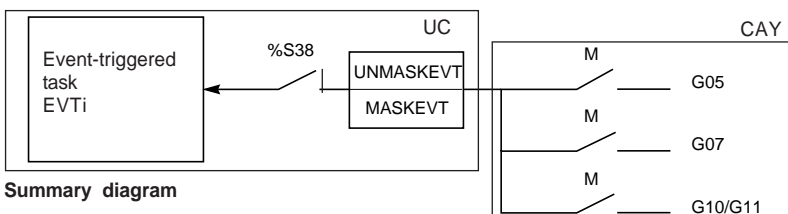
- if several event sources are chosen the following bits are used to determine what triggered the application event processing :
  - EVT\_G1 (lxy.i.50) end of G10 or G11 on event
  - EVT\_G05 (lxy.i.48) end of G05 on event
  - TO\_G05 (lxy.i.49) time period of G05 elapsed
  - EVT\_G07 (lxy.i.47) memorization of position
  - EVT\_MOD (%lxy.i.51) modulo crossing
- bit OVER\_EVT (%lxy.i.46) this bit detects a delay in the transmission of the event or loss of the event.
- value of memorized positions PREF1 (%IDxy.i.9) and PREF2 (%IDxy.i.11).

**Note : the bits and words described above are the only values updated in the event-triggered task and are only updated in the PLC when the task is activated.**

#### Masking of events

PL7 language offers 2 ways of masking events :

- MASKEVT instruction for global masking of events, (instruction UNMASKEVT unmaskes them),
- bit %S38 : 0 = global inhibition of events (the bit is normally at 1).



Summary diagram

---

## 4.5 Managing the operating modes

---

### Switching on the module

When the TSX CAY module is switched on or connected it performs a number of self-tests with the outputs in safety position (outputs at 0).

At the end of the self-tests :

- if the self-tests have detected no errors : the module tests the configuration with the outputs in safety position, and if the configuration is correct, the module changes to measurement mode (OFF),
- if the self-tests have detected any errors or if the configuration is incorrect, the module signals a fault and maintains the outputs in safety position.

### PLC in RUN

All the operating modes of the configured channels can be used.

### PLC changes from RUN to STOP (or Processor <--> module communication is lost) :

The moving part decelerates and stops, and the modules changes to measurement mode (OFF).

Reminder : bit %S13 detects when the PLC changes to STOP. It is set to 1 during the first scan after the PLC is set to RUN.

### Changing the configuration (reconfiguration)

- the moving part decelerates and stops,
- the channel is deconfigured,
- the channel tests the new configuration with the outputs in safety position,
- if the new configuration is correct, the channels changes to measurement mode (OFF),
- if the configuration is incorrect, the module signals a fault and maintains the outputs in safety position.

### Power outage and return

The moving part stops at a power outage.

At a cold start or warm restart, the channel configuration is automatically transmitted by the processor to the module, which changes to measurement mode (OFF) then to the mode requested by the program.



---

## 4.6 Fault management

---

### 4.6-1 Role

Checking for faults is of primary importance in the area of position control because of the inherent risks when moving parts are in motion.

The checks are performed internally and automatically by the module.

4 types of fault are detected :

- module faults. These are hardware faults in the module. All the axes controlled by the module are therefore affected when this type of fault occurs. They may be detected during the self-tests (when the module is reinitialized) or during normal operation (I/O fault).
- channel hardware faults external to the module (eg : encoder wiring break)
- channel application faults linked to the axes (eg : deviation).  
Continuous checks are made for axis level faults when the axis is configured.
- channel command failures. These are faults which may occur during the execution of a movement command, the transfer of the configuration, the transfer of adjustment parameters or a change of operating mode.

Note :

- checking for certain axis level faults can be enabled or inhibited by the error control parameters of the axis. These error control parameters can be adjusted in the adjustment screen.
  - in direct drive mode (DIRDRV), the application fault check is inhibited,
  - in measurement mode (OFF), the application fault check is inhibited, except for the soft stop fault.
- 

### 4.6-2 Principle

Faults are classified according to two levels of gravity :

- **critical or blocking faults** which cause the moving part to stop in the case of an axis fault or all of the moving parts managed by the module in the case of a module fault. They cause the following to occur :
    - fault indication,
    - deceleration of the moving part until the analog output is zero,
    - deactivation of the speed drive enable relay,
    - deletion of all the commands which have been memorized,
    - wait for acknowledgment.
 The fault must have disappeared and been acknowledged before the application can be restarted.
  - **non critical faults** which cause the fault to be indicated without stopping the moving part. Any actions to be performed are the responsibility of the user and are executed in the PL7 program.  
The indication of the fault disappears when the fault has disappeared and is acknowledged (the acknowledgment is not memorized and only occurs if the fault has disappeared).
-

### 4.6-3 Programming

Faults can be displayed, corrected and acknowledged from the debug screen, but it may be more useful during operation to be able to control the moving part and correct faults from an operator terminal. The application has all the necessary data and commands for this purpose.

#### Fault indication

The module provides a large amount of data in the form of status bits and words which can be accessed via the PL7 program. These bits make it possible to deal with faults hierarchically :

- so that they can be used in the main program,
- to indicate the fault.

There are 2 levels of indication :

• 1st level : general data	
<b>%Ixy.i.ERR</b>	: channel fault
<b>AX_OK</b>	: (%Ixy.i.3) no blocking fault (with stopping of the moving part) is detected
<b>AX_FLT</b>	: (%Ixy.i.2) fault (covers all faults)
<b>HD_ERR</b>	: (%Ixy.i.4) external hardware fault
<b>AX_ERR</b>	: (%Ixy.i.5) application fault
<b>CMD_NOK</b>	: (%Ixy.i.6) command failure
• 2nd level : detailed data	
module and axis fault status words (%MWxy.i.2 and %MWxy.i.3)	

Each fault is described in detail in the following sections. Status words are also described in detail in the quick reference guide.

In general it is advisable to stop the evolution of the sequential processing assigned to the axis when a blocking fault occurs and to control the moving part in manual mode while the fault is corrected. Correction of the fault should be followed by an acknowledgment.

#### Fault acknowledgment

When a fault occurs :

- fault bits AX\_FLT, HD\_ERR, AX\_ERR and bits extracted from the status words concerned by the fault change to 1.
- if the fault is with stop, the OK bit changes to 0.

When the fault disappears, the status of all the fault bits remains unchanged. The fault is memorized until it is acknowledged : bit ACK\_DEF %Qxy.i.8 is set to 1 (or the module is reinitialized). The fault must be acknowledged after it has disappeared (except in the case of soft stop faults).

If several faults are detected, the acknowledgment command only applies to those faults which have totally disappeared. Faults which remain must be acknowledged again when they have disappeared.

#### Note :

A fault can also be acknowledged when the PLC is initialized, or when a new, correct command is accepted in the case of a command failure.

**4.6-4 Summary table**

The following table summarizes the various types of fault and the associated bits.

Channel faults	Process faults :bit <b>AX_FLT</b> %lxy.i.2		
	<b>AX_OK</b> %lxy.i.3 (no blocking fault has been detected)		Command failure bit <b>CMD_NOK</b> %lxy.i.6
Bit %lxy.i.ERR	External hardware bit <b>HD_ERR</b> %lxy.i.4	Application bit <b>AX_ERR</b> %lxy.i.5	
<ul style="list-style-type: none"> <li>• Internal</li> <li>• Communication</li> <li>• Configuration</li> <li>• External hardware</li> <li>• Configuration or adjustment</li> </ul>	<ul style="list-style-type: none"> <li>• Emergency stop</li> <li>• Speed drive</li> <li>• Encoder wiring break</li> <li>• Analog output short-circuit</li> <li>• Auxil. output short-circuit</li> <li>• Encoder supply</li> <li>• Absolute encoder frame</li> </ul>	<ul style="list-style-type: none"> <li>• Soft stops</li> <li>• Overspeed</li> <li>• Recalibration</li> <li>• Deviation MAX_F1</li> <li>• Deviation MAX_F2</li> <li>• Stop fault</li> <li>• Target window</li> </ul>	Fault coded in word <b>CMD_FLT</b> %MWxy.i.7

faults in shaded boxes are non-blocking faults and have no effect on bit AX\_OK.

**4.6-5 Description of channel faults**

Bit %lxy.i.ERR covers all channel faults :

- internal fault (%MWxy.i.2:X4) = module missing, off or performing self-tests,
- communication fault (%MWxy.i.2:X6) = communication fault with the processor,
- configuration fault (%MWxy.i.2:X5) = fault between the declared position of the module in the configuration and the actual position.

**Note** :%MW words require a READ\_STS command to be updated (see section 10.7 of this manual and Common application-specific functions, Volume 1).

#### 4.6-6 Description of external hardware faults

These faults are indicated by bit **HD\_ERR** (%Ix.y.i.4), and are all blocking faults which cannot be deactivated.

##### Fault : **Emergency stop**

<b>Cause</b>	Open circuit between 0V and Emergency stop inputs on the module front panel.
<b>Parameter</b>	None
<b>Consequence</b>	The moving part is forced to stop.
<b>Indication</b>	Bit EMG_STP (%MWxy.i.3:X5) (1)
<b>Remedy</b>	Reconnect to the 24 V of the input and acknowledge the fault.

##### Fault : **Speed drive**

<b>Cause</b>	Open circuit between 24V and speed drive fault input on the module front panel.
<b>Parameter</b>	None
<b>Consequence</b>	The moving part is forced to stop.
<b>Indication</b>	Bit DRV_FLT (%MWxy.i.3:X2) (1)
<b>Remedy</b>	Correct and acknowledge the speed drive fault.

##### Fault : **Encoder wiring break**

<b>Cause</b>	Complementarity fault in data from the encoder
<b>Parameter</b>	None
<b>Consequence</b>	The axis is dereferenced (in the case of an incremental encoder) The moving part is forced to stop.
<b>Indication</b>	Bit ENC_BRK (%MWxy.i.3:X4) (1)
<b>Remedy</b>	Reconnect the offending encoder and acknowledge the fault.

**Note** : when an encoder connection fault occurs, the module no longer reads the measurement. With an absolute encoder, no more pulses are sent on the CLK line, until the fault is remedied and acknowledged.

##### Fault : **Analog output short-circuit**

<b>Cause</b>	Short-circuit detected on one of the module analog outputs.
<b>Parameter</b>	None
<b>Consequence</b>	The moving part is forced to stop
<b>Indication</b>	Bit ANA_FLT (%MWxy.i.3:X0) (1)
<b>Remedy</b>	Correct the short-circuit and acknowledge the fault.

(1) %MW words require a READ\_STS command in order to be updated (see section 10.7 of this manual and Common application-specific functions, Volume 1).

**Fault : Auxiliary output short-circuit**

<b>Cause</b>	Short-circuit detected on one of the module auxiliary outputs.
<b>Parameter</b>	None
<b>Consequence</b>	The moving part is forced to stop
<b>Indication</b>	Bit AUX_FLT (%MWxy.i.3:X1) (1)
<b>Remedy</b>	Correct the short-circuit and acknowledge the fault.

**Fault : Encoder supply**

<b>Cause</b>	The encoder is no longer powered
<b>Parameter</b>	None
<b>Consequence</b>	The axis is dereferenced (in the case of an incremental encoder). The moving part is forced to stop.
<b>Indication</b>	Bit ENC_SUP (%MWxy.i.3:X3) (1)
<b>Remedy</b>	Reconnect the supply and acknowledge the fault.

**Fault : Absolute encoder frame**

<b>Cause</b>	Fault on the SSI frame : parity or error bit
<b>Parameter</b>	None
<b>Consequence</b>	The moving part is forced to stop.
<b>Indication</b>	Bit ENC_FLT (%MWxy.i.3:X7) (1)
<b>Remedy</b>	Correct and acknowledge the fault.

**Fault : 24V supply**

<b>Cause</b>	24V supply fault
<b>Parameter</b>	None
<b>Consequence</b>	The moving part is forced to stop.
<b>Indication</b>	Bit AUX_SUP (%MWxy.i.3:X6) (1)
<b>Remedy</b>	Reconnect the supply and acknowledge the fault.

(1) %MW words require a READ\_STS command in order to be updated (see section 10.7 of this manual and Common application-specific functions, Volume 1).

#### 4.6-7 Description of application faults

These faults are indicated by bit **AX\_ERR** (%Ixy.i.5). The parameters can be accessed via the configuration editor adjustment screen. There is **no** check for faults associated with soft stops for **infinite axes** (modulo).

Fault : **Soft stops** (blocking fault which cannot be deactivated)

<b>Cause</b>	The moving part is no longer located between the 2 limit values : upper and lower soft stop limits (this check is activated as soon as the axis is referenced).
<b>Parameter</b>	Upper soft stop limit : SL_MAX (%MDxy.i.31) Lower soft stop limit : SL_MIN (%MDxy.i.33)
<b>Consequence</b>	The moving part is forced to stop.
<b>Indication</b>	<ul style="list-style-type: none"> <li>• bit %MWxy.i.3:X8 upper soft stop overshoot</li> <li>• bit %MWxy.i.3:X9 lower soft stop overshoot</li> </ul>
<b>Remedy</b>	<p>Acknowledge the fault and return the moving part from outside the soft stop limits to within the valid measurement area in manual mode. To do this, check that :</p> <ul style="list-style-type: none"> <li>• there is no movement in progress</li> <li>• manual mode has been selected</li> <li>• the STOP command is at zero</li> <li>• the axis to which this command applies is referenced</li> <li>• there is no other fault with stop on the axis.</li> </ul> <p>Moving parts are returned manually or using the JOG+ and JOG- commands.</p>

Fault : **Overspeed** (blocking fault which cannot be deactivated)

<b>Cause</b>	The speed of the moving part has exceeded maximum speed plus the overspeed threshold on one of the axes VMAX (1+OVR_SPD)
<b>Parameter</b>	Overspeed threshold OVR_SPD (%MWxy.i.23). If this parameter is at 0, the check is inhibited.
<b>Consequence</b>	The moving part stops
<b>Indication</b>	Bit %MWxy.i.3:X10
<b>Remedy</b>	Acknowledge the fault

Fault : **Deviation** (MAX\_F1 blocking, MAX\_F2 non-blocking, and can be deactivated)

<b>Cause</b>	The module compares the measured position of the moving part during movement. A fault is detected when the position error exceeds the maximum permitted deviation defined by the user.
<b>Parameter</b>	Non-critical abnormal position error MAX_F2 (%MDxy.i.47) Critical abnormal position error MAX_F1 (%MDxy.i.45) if these parameters are at 0, the check is inhibited.
<b>Consequence</b>	If deviation MAX_F2 is exceeded : the fault is indicated, If deviation MAX_F1 is exceeded : the moving part is stopped. This fault is only recognized if MAX_F1 is other than 0.
<b>Indication</b>	<ul style="list-style-type: none"> <li>• bit %MWxy.i.3:X15 deviation MAXF_2 exceeded</li> <li>• bit %MWxy.i.3:X11 deviation MAXF_1 exceeded</li> </ul>
<b>Remedy</b>	Check the servo loop and acknowledge the fault.

---

**Fault : Stop** (non-blocking fault which can be deactivated)

<b>Cause</b>	As soon as the calculated speed reference value becomes 0, the module starts a time delay T_STOP : <ul style="list-style-type: none"> <li>• if this parameter is set at 0, the fault check is inhibited</li> <li>• if this parameter is not 0, when the time delay has elapsed, the module compares the measured speed of the moving part and the stopping speed S_STOP. If the measured speed exceeds S_STOP, the group detects a stopping fault.</li> </ul>
<b>Parameter</b>	T_STOP (%MWxy.i.25) maximum stop detection time S_STOP (%MWxy.i.24) speed at which the moving part is declared to be stopped.
<b>Consequence</b>	The fault is indicated
<b>Indication</b>	• bit %MWxy.i.3:X14
<b>Remedy</b>	• correct the fault or use new settings • <b>acknowledge the fault</b>

**Fault : Target window** (non-blocking fault which can be deactivated)

<b>Cause</b>	When a move to a position with stop is requested, the module checks, using the theoretical stop, that the position reached is the required position, with a tolerance defined by the user in parameter TW (setpoint-TW-measurement-setpoint+TW) If this parameter is at 0, the check is inhibited.
<b>Parameter</b>	TW (%MDxy.i.49) target window
<b>Consequence</b>	If the moving part is not in the target window : the fault is indicated.
<b>Indication</b>	• bit %MWxy.i.3:X13 target window fault
<b>Remedy</b>	check the servo loop and acknowledge the fault

**Fault : Recalibration** (non-blocking fault which can be deactivated)

<b>Cause</b>	During a recalibration event, the difference between the current position and the recalibration reference value exceeds the recalibration threshold (if the configuration parameter "Recalibration function absent" is selected, the check is inhibited).
<b>Parameter</b>	RE_WDW (%MDxy.i.51) recalibration deviation threshold RE_POS (%MDxy.i.43) recalibration reference value
<b>Consequence</b>	If the difference exceeds the threshold : the fault is indicated.
<b>Indication</b>	• bit %MWxy.i.3:X12 recalibration fault
<b>Remedy</b>	check the servo loop and acknowledge the fault

**Note**

%MW words require a READ\_STS command in order to be updated (see section 10.7 of this manual and Common application-specific functions, Volume 1).

Fault: **Movement control** (blocking fault which can be deactivated)

<b>Cause</b>	When the analog output of a channel exceeds a limit VLIM (in absolute value), a time delay T is activated. When T is reached, a fault has been detected if the position value is the same as that of the internal cycle of the previous TSX CAY module.
<b>Parameter</b>	Analog output limit VLIM : %MWxy.i.27 The time delay T is programmed in TACC/2. TACC is the acceleration adjustment parameter : %MWxy.i.26
<b>Consequence</b>	If the fault is detected the moving part is stopped (analog output set to 0 and the speed control authorization relay open). Control is enabled only if VLIM > 0.
<b>Indication</b>	Bit %MWxy.i.3:X11 deviation MAX_F1 exceeded.
<b>Remedy</b>	Check the loop and acknowledge the fault.

#### Note

Movement control is active in direct control and manual and automatic modes.

Fault : **Zero Marker presence check**

<b>Cause</b>	When setting a short cam type reference point with a zero marker.
<b>Parameter</b>	None
<b>Consequence</b>	The axis stops.
<b>Indication</b>	Bit %lxy.i.6 Word %MWxy.i.7 = 16#0015. (CMD_FLT)
<b>Remedy</b>	Adjust the cam mechanically and restart the operation.



**4.6-8 Description of command failure faults**

A command failure fault occurs every time a command cannot be executed, either because this command is not compatible with the state of the axis or the current mode, or because at least one of the parameters does not belong in the area of validity. These faults are indicated by the Cmd Fail indicator lamp in the debug screens. The source of the command failure can be ascertained by pressing the DIAG key at channel level. They can also be accessed via the program using bit CMD\_NOK (%Ixy.i.6) and word CMD\_FLT (%MWxy.i.7).

**Fault : Command failure**

<b>Cause</b>	<ul style="list-style-type: none"> <li>• unauthorized movement command,</li> <li>• incorrect configuration or parameter transfer.</li> </ul>		
<b>Parameter</b>	-		
<b>Consequence</b>	<ul style="list-style-type: none"> <li>• immediate stop of current movement,</li> <li>• resetting of the buffer memory receiving the movement commands in automatic mode.</li> </ul>		
<b>Indication</b>	<ul style="list-style-type: none"> <li>• bit CMD_NOK (%Ixy.i.6) Movement command failure,</li> <li>• word CMD_FLT (%MWxy.i.7) Word coding the type of fault detected.</li> </ul> <div style="text-align: center; margin: 10px 0;"> <p>CMD_FLT</p> <table border="1" style="margin: auto; border-collapse: collapse;"> <tr> <td style="padding: 5px;">Configuration and adjustment parameter</td> <td style="padding: 5px;">Movement commands</td> </tr> </table> <p style="margin: 5px 0;">High order byte                      Low order byte</p> </div> <p>More detailed information is given in section 12.</p>	Configuration and adjustment parameter	Movement commands
Configuration and adjustment parameter	Movement commands		
<b>Remedy</b>	<ul style="list-style-type: none"> <li>• implicit acknowledgment on reception of a new accepted command,</li> <li>• acknowledgment is also possible using the command ACK_DEF (%Qxy.i.8).</li> </ul>		

**Note :**

In the case of linked movements in automatic mode, it is advisable to make the execution of each movement conditional on the end of execution of the previous movement, using bit AX\_FLT (%Ixy.i.2). This ensures that a link is not made to the following command when the current command has failed.

---

#### 4.6-9 Fault Masking

It is possible to individually mask 4 of the 8 external hardware faults on TSX CAY •2 and 33 modules (so that bit **HD\_ERR** is set to 1) when configuring the channel (MSK\_HDERR parameter):

- **DRV\_FLT** : speed drive fault
- **ENC\_SUP** : encoder supply fault
- **AUX\_SUP** : 24 V supply fault
- **EMG\_STP** : emergency stop fault

For each of the faults, "masking" means that the corresponding status variable %MWxy.i.2 and %I.ERR are not updated.

AX\_FLT, AX\_OK and HD\_ERR data is updated without taking the masking into account.

Any movement of the axis is **interrupted** by a stop and disabling of the speed drive.

## 4.7 Management of manual mode

Manual mode can be selected and controlled from the application-specific debug screen (see debug section) or, using the application program, from a front panel, man-machine interface terminal or supervision terminal.

In this case, the dialog is programmed in Ladder, Instruction list or structured text language, using elementary commands (movements, reference point, etc).

### 4.7-1 Selecting manual mode

This is performed by assigning the value 2 to word MOD\_SEL (%QWxy.i.0).

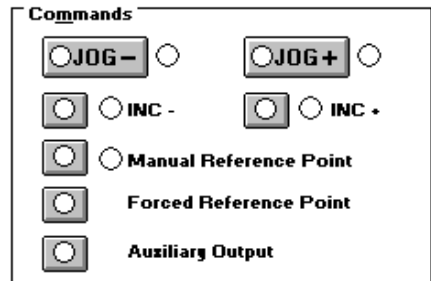
The change from the current mode to manual mode forces the moving part to stop if there is a movement in progress in another mode and is effective as soon as the moving part is stationary.

When the manual mode command is recognized, bit IN\_MANU (%Ix.y.i.22) is set to 1.

### 4.7-2 Execution of manual commands

The following elementary commands can be accessed via command bits %Qxy.i.j associated with manual mode :

- visual move in the + direction JOG\_P (%Qxy.i.1) and in the - direction JOG\_M (%Qxy.i.2),
- incremental move in the + direction INC\_P (%Qxy.i.3) and in the - direction INC\_M (%Qxy.i.4),
- manual reference point SET\_RP (%Qxy.i.5),
- forced reference point RP\_HERE (%Qxy.i.6).



These commands are equivalent to those which are accessed from the referenced axis TSX CAY 21/41 module debug screen.

#### General conditions for executing commands in manual mode :

- target position within the soft stop limits (1),
- axis with no blocking fault (bit AX\_OK : %Ix.y.i.3 = 1),
- no command being executed (bit DONE : %Ix.y.i.1 = 1),
- command STOP (%Qxy.i.15) inactive and speed drive safety relay enable bit ENABLE (%Qxy.i.9) at 1.

#### A movement can be stopped by :

- appearance of the command STOP (%Qxy.i.15) or bit ENABLE (%Qxy.i.9) at 0,
- occurrence of a blocking fault,
- change of operating mode,
- reception of a configuration.

(1) except, in the event of a soft stop fault, for the JOG\_P and JOG\_M commands, after acknowledgment of the fault.

### 4.7-3 Detailed description of manual commands

#### Visual movement command : JOG\_P and JOG\_M

Bits JOG\_P (%Qxy.i.1) and JOG\_M (%Qxy.i.2) control the movement of the moving part in a positive or negative direction. The operator should visually follow the position of the moving part. The movement continues as long as the command is present and is not inhibited by a STOP command or a fault.

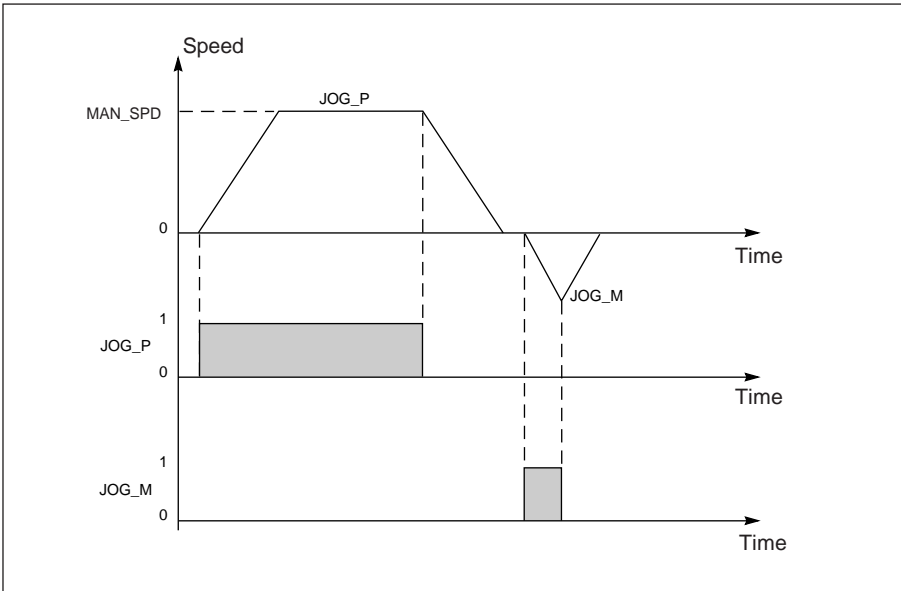
For limited axes, the JOG+ and JOG - commands trigger an automatic stop at the latest at a distance from the soft stops equal to the target window.

Commands JOG\_P and JOG\_M are taken into account on an edge and maintained active on a state, whether the axis is referenced or not.

The movement is performed at the speed of manual mode MAN\_SPD defined in the adjustment screen (or in double word %MDxy.i.35).

The speed can be adjusted while moving using coefficient CMV (%QWxy.i.1).

Any operating speed above VMAX (maximum speed of the axis defined at configuration) is peak limited to value VMAX.



#### Notes :

- these commands are also used to release the moving part when a soft stop fault is detected and has been acknowledged.
- if bit JOG\_P or M is at 1 when changing to manual mode, this command is not taken into account. It will only be taken into account after the bit changes to 0 and is reset to 1.

**Incremental movement command : INC\_P and INC\_M**

Bits INC\_P (%Qxy.i.3) and INC\_M (%Qxy.i.4) control the movement of the moving part by one position increment in a positive or negative direction.

The value of the position increment PARAM is entered in the double word QDxy.i.2 or in the TSX CAY module debug screen.

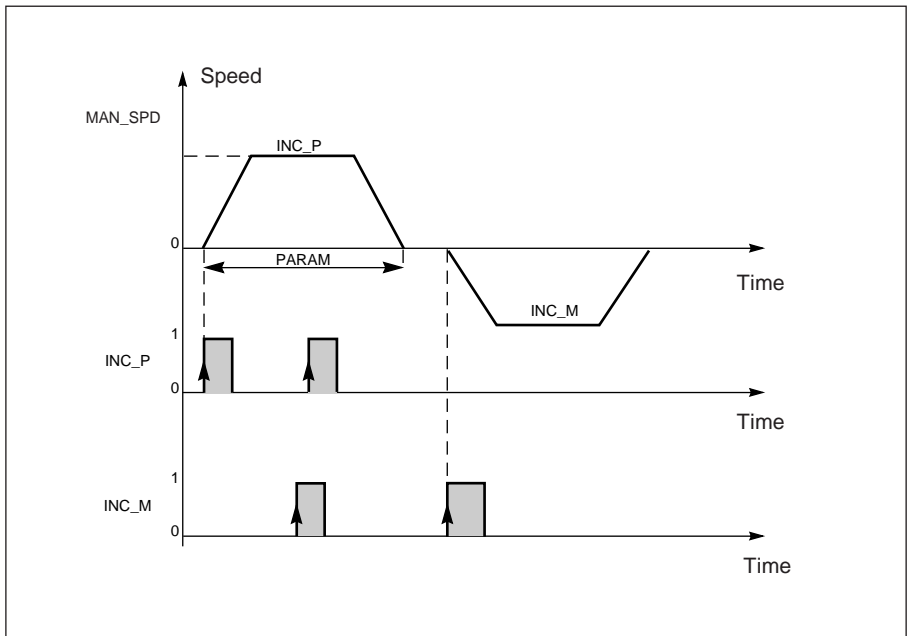
In addition to the general conditions for execution in manual mode, commands INC\_P and INC\_M are active on a rising edge when :

- the axis is referenced for limited travel machines
- the target position is within the soft stop limits.

The movement is performed at the speed of manual mode MAN\_SPD defined in the adjustment screen (or in double word %MDxy.i.35).

The speed can be adjusted while moving using coefficient CMV (%QWxy.i.1).

Any operating speed above VMAX (maximum speed of the axis defined at configuration) is peak limited to value VMAX.



## • USE AS AN INCREMENTAL ENCODER

### Reference point command : SET\_RP

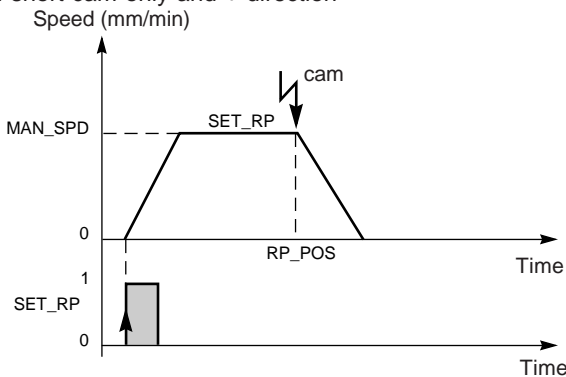
Bit SET\_RP (%Qxy.i.5) sets a manual reference point and moves the part.

The type and direction of the reference point are defined during configuration in the Reference point parameter. The reference point value is defined in the parameter RP Value in the adjustment screen (or double word RP\_POS in %MDxy.i.41).

The approach speed is the manual speed MAN\_SPD defined in the adjustment screen (or in double word %MDxy.i.35) multiplied by the speed correction coefficient CMV. The reference point speed varies according to the type of reference point selected.

Any operating speed above VMAX (maximum speed of the axis defined during configuration) is peak limited to value VMAX.

Example : short cam only and + direction



### Forced reference point command : RP\_HERE

Bit RP\_HERE (%Qxy.i.6) forces a reference point without moving the part to the value defined in parameter PARAM entered in the double word %QDxy.i.2 or in the TSX CAY module debug screen.

This command references the axis without moving the part.

#### Notes :

- command RP\_HERE does not modify the value of parameter RP\_POS.
- the value of parameter PARAM must be within the soft stop limits.
- all blocking faults are tolerated during execution of this command (with the exception of an encoder wiring break).

---

**• USE WITH AN ABSOLUTE ENCODER AND ASSISTED OFFSET****Clear reference command : SET\_RP**

This is a cancel operation with preliminary reference to a referencing command. An edge on bit SET\_RP (%Qxy.i.5) is used to change the axis to a non-referenced state in order to move the moving part without a soft stop fault. It is however not possible to overshoot in either direction a position which is outside the measurement range of the absolute encoder.

The parameter ABS\_OFF (%MDxy.i.53) is forced to 0.

**Referencing and offset calculation command : RP\_HERE**

An edge on bit RP\_HERE (%Qxy.i.6) is used to change the axis to referenced state.

If the encoder has been declared with assisted offset, the offset is recalculated so that, at the current point, it is at the position defined in parameter PARAM entered in double word %QDxy.i.2 or in the debug screen.

In this case, saving the adjustment parameters must be forced so that they are not lost on restart :

- Either using the "Save Parameters" function in the adjustment screen
- Or via the application program, using the SAVE\_PARAM function

**Notes :**

- The value of PARAM must be within the soft stops
- The offset calculation fails if an adjustment is in progress or if the axis is at referenced state
- If the resolution is modified, the offset must be recalculated

## 4.8 Managing direct drive mode (DIRDRIVE)

### 4.8-1 Selecting direct drive

This is performed by assigning value 1 to word MOD\_SEL %QWxy.i.0.

When there is a request to change mode, the moving part is stopped and then there is an effective change of mode. When the direct drive command is taken into account, bit IN\_DIRDR %Ixy.i.21 is set to 1.

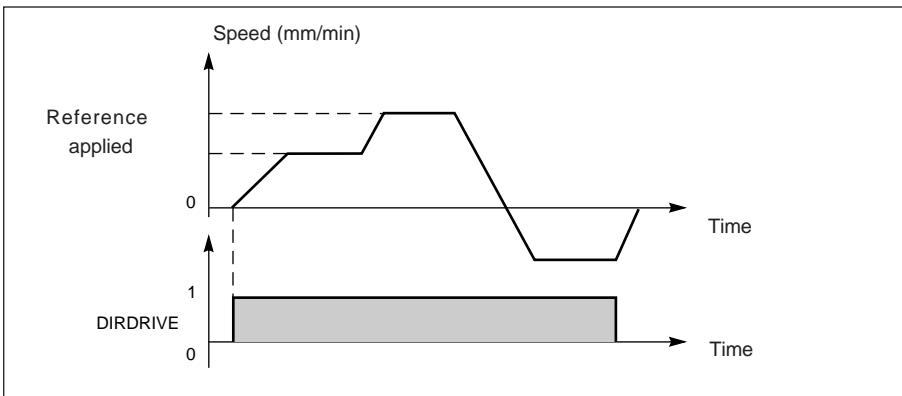
### 4.8-2 Executing commands in direct drive mode

Direct drive mode has a movement command DIRDRV %Qxy.i.0. The axis acts as a digital/analog converter and the position loop is off. The behavior of the axis can thus be analyzed independently of the control loop.

The voltage of the speed drive is controlled between -UMAX and +UMAX (value of UMAX is given in the configuration screen). It is expressed in mV, and the application of this reference is rounded up to a multiple of 1.25 mV (eg : for 1004mV requested, this gives 1003.75 mV and the screen displays 1003 mV).

The voltage reference is sent periodically by variable PARAM %QDxy.i.2. The sign of this variable gives the direction of movement. The software fault checks are inhibited (except for the soft stop check if the axis is referenced).

To avoid damage to the mechanical parts, when changing reference, the changeover to the new value is performed respecting the acceleration/deceleration value.



When the reference changes, the output reaches the new reference according to a trapezoid speed profile respecting the configured acceleration.



**General conditions for executing the DIRDRIVE command :**

- axis with no blocking fault (bit AX\_OK : %lxy.i.3 = 1),
- command STOP (%Qxy.i.15) inactive and speed drive safety relay enable bit ENABLE (%Qxy.i.9) at 1,
- voltage parameter PARAM (%QDxy.i.2) between - UMAX and + UMAX of the selected axis.

**A movement can be stopped by the :**

- appearance of the STOP command or speed drive safety relay enable bit ENABLE (%Qxy.i.9) at 0,
- occurrence of a blocking fault, or a soft stop fault,
- change of operating mode,
- reception of a configuration.

**4.9 Management of measurement mode (OFF)**

This mode must be used every time the moving part is likely to move beyond the control of the module (moving part moved by hand or controlled by an external device).

In this mode, the module remains passive. It only updates current position (%IDxy.i.0) and speed (%IDxy.i.2) data.

Measurement mode is selected by assigning value 0 to word MOD\_SEL %QWxy.i.0. It is also the mode selected by the module when the PLC is in STOP, and the default mode following channel configuration.

The OFF mode has no associated movement command.

The motion of the moving part is not controlled, and the checks for software faults are inhibited (except for the soft stop check). The position loop is off.

The speed drive enable relay is unlocked, independently of the state of bit ENABLE (%Qxy.i.9).

The command AUX\_OUT (bit %Qxy.i.11) controls the auxiliary output. The command RP\_HERE can be executed in this mode.



---

## 5.1 Operations prior to adjustment

---

### 5.1-1 Preliminary conditions

- TSX CAY module(s) installed in the PLC,
- axis control application(s) connected to the TSX CAY modules,
- terminal connected to the PLC via the terminal port or network,
- axis control configuration and program created and transferred to the PLC processor,
- PLC in RUN. It is advisable to inhibit the motion control application program (eg using a program execution condition bit for example) to facilitate adjustment operations.

---

### 5.1-2 Preliminary checks

- check the wiring,
- check that the movements can be safely performed,
- check that the travel limits are wired in accordance with safety regulations (these generally apply directly to the speed drive supply sequence),
- check the connection polarity of the tachogenerator.

---

### 5.1-3 Adjusting the speed drive

Adjust the speed drive in accordance with the manufacturer's instructions using a control station connected in place of the axis control module.

#### Adjusting the current loop

- set the maximum value for the current supplied by the speed drive to a value which is acceptable to the motor (switching dissipation) and the mechanism (accelerating torque),
- set the stability of the current loop.

#### Adjusting the speed loop

- set the maximum working speed, give the speed drive a reference equal to the maximum operating voltage (UMAX).
- set the speed loop gain,
- set the offset.

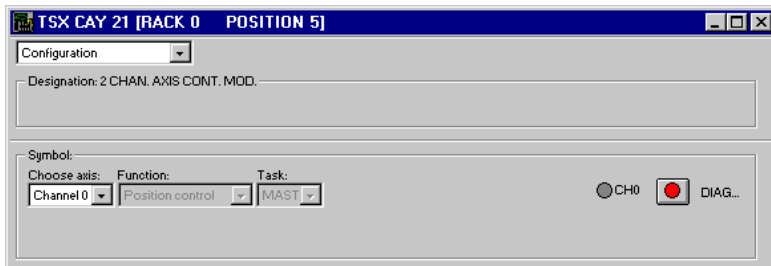
#### Adjusting the current limit as a function of the speed

- reconnect the axis control module at the end of adjustment,
  - continue to adjust the current loop.
-

## 5.2 Adjusting the configuration parameters

### 5.2-1 Access to the configuration parameters

Access configuration of the axis control module parameters by selecting the **Configuration** editor and double-clicking on the position in the rack which contains the axis control module.

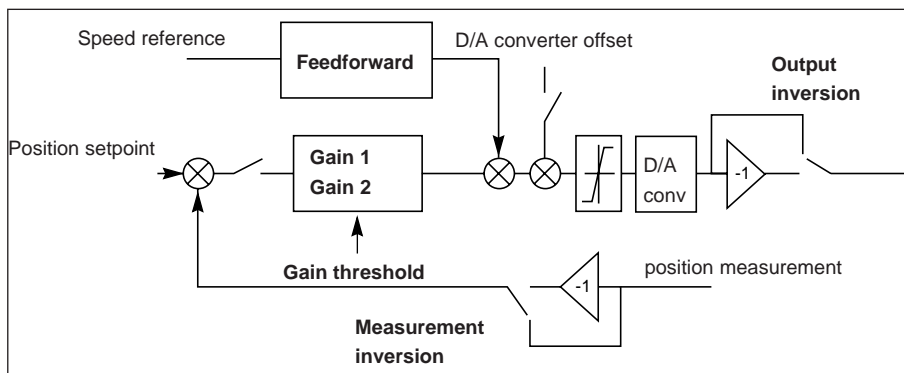


Select the rack to be adjusted from the "Axis Selection" pull-down menu : channel 0 to 3.

The zone in the lower part of the configuration window can be used to access the parameters to be adjusted.

### 5.2-2 Inversion parameters

These parameters define the inversion of the setpoint between the digital/analog converter output and the speed drive, and/or the measurement inversion (in the case of an incremental encoder).



#### Options

- no inversion (default value),
- inversion of measurement direction,
- inversion of speed drive setpoint,
- inversion of both.

### Procedure for defining the inversion parameter

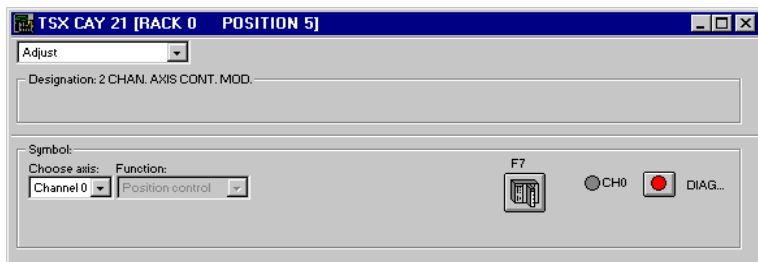
- Select Debug mode,
- Select direct drive mode (DIRDRIVE),
- Acknowledge any faults : Ack. button in Faults zone,
- Then enter in succession + 100 mV (analog output positive) and - 100 mV (analog output negative) in the PARAM field in accordance with the table below,  
**Note** : if the offset is above 100mV, adjust this first, see section 5.3-4.

Analog output	Position	Measurement	Action
Positive	increases	increases	none (connection OK)
Positive	increases	decreases	invert the measurement
Positive	decreases	decreases	invert the setpoint
Positive	decreases	increases	invert setpoint and measurement
Negative	decreases	decreases	none (connection OK)
Negative	decreases	increases	invert the measurement
Negative	increases	increases	invert the setpoint
Negative	increases	decreases	invert setpoint and measurement

## 5.3 Adjusting the parameters

### 5.3-1 Access to the adjustment parameters

The adjustment parameters are accessed via the **Adjustment** command in the **View** menu in the TSX CAY module configuration screen (or by selecting Adjustment in the pull-down list in the module zone of the parameter or debug configuration screen).



Select the channel to be adjusted from the "Select axis" pull-down menu : channel 0 to 3.

The  button is used to display either the current or the initial parameters.

The **initial parameters** are :

- parameters entered (or defined by default when the configuration is validated) in offline mode and provided when transferring the program to the PLC,
  - parameters taken into account at the last reconfiguration in online mode.
- These cannot be modified from this screen. However, they can be updated from the current adjustment parameters.

The **current parameters** are those modified and validated from the adjustment screen in online mode (or via the program, by explicit exchange). These parameters are replaced by the initial parameters at a cold restart.

It is essential that the parameters are saved following this adjustment parameter definition phase.

The lower part of the screen then displays the adjustment parameters.

<b>Corrected resolution</b> Distance <input type="text" value="551"/> Counts <input type="text" value="110"/> <input type="button" value="Correction"/> Encoder offset <input type="text" value="0"/> Pulses	<b>Position loop</b> Gain 1 <input type="text" value="1 000"/> /100s Offset <input type="text" value="0"/> mV Gain 2 <input type="text" value="1 000"/> /100s Feed forward <input type="text" value="10"/> % Threshold 1-2 <input type="text" value="500"/> /1000s Vmax
<b>Movement control</b> Following error 1 <input type="text" value="8 888"/> Following error 2 <input type="text" value="8 888"/> Recalibration position <input type="text" value="0"/> Recalibration deviation <input type="text" value="0"/> Overspeed <input type="text" value="12"/> % of Vmax VLim <input type="text" value="0"/> mV	<b>Command</b> Software hi limit <input type="text" value="0"/> Software lo limit <input type="text" value="-888 888"/> Acceleration Vmax f <input type="text" value="8"/> ms Acceleration profile <input type="text" value="Rectangle"/> <input type="button" value=""/>
<b>Stop control</b> Delay <input type="text" value="500"/> ms Speed <input type="text" value="111"/> Target window <input type="text" value="8 888"/>	<b>Magual mode parameter</b> Speed <input type="text" value="1 111"/> Ref. point value <input type="text" value="-666 666"/>

To display the entire adjustment parameters zone, deselect the **View/Module Zone** and **View/Channel Zone** commands (to restore these zones, use the same commands).

### 5.3-2 Encoder offset

This parameter only concerns absolute encoders. It is used to bring the actual position of the moving part in line with the position provided by the encoder (zero offset). For direct offset, the user enters the offset value in encoder points in the ABS\_OFF parameter.

For assisted offset, the commands RP\_HERE and SET\_RP are used.

This parameter can only be modified if "direct offset" was selected during configuration.

Encoder offset = value to be added (expressed as a number of encoder points) to the measurement from the absolute encoder to obtain the actual measurement.

Limits :  $-2^{n-1}+1$  to  $2^{n-1}-1$ , with n number of data bits of the absolute encoder.

#### Example :

Thus, if for position 0, the absolute encoder provides a measurement of 100mm, and the resolution is  $2\mu\text{m}$ , the offset value is :  $-100\ 000 / 2 = -50\ 000$  encoder points.

#### Note :

- . This parameter is set in measurement mode (DRV\_OFF)
- . If the absolute encoder has been declared with assisted offset, this parameter is not taken into account. The assisted offset procedure enables the user to avoid any calculations. However, after reading (READ\_PARAM, SAVE\_PARAM or save parameters), its value reflects the offset used by the channel.

### 5.3-3 Adjusting the resolution

This adjustment is designed to correct the error resulting on the one hand from imprecise entry of the values of the configuration parameters, and on the other any imperfections in the drive chain.

#### Procedure :

Perform the following operations in the TSX CAY Debug screen :

- 1 select **Manual** mode,
- 2 set a manual reference point if an incremental encoder is used,
- 3 select as the distance to travel (**Theoretical Distance**) a value corresponding to the largest possible movement : position 1 and enter this value in the **Param** field (300000 $\mu$ m for example),
- 4 control the movement **Inc-** or **Inc+** according to the direction of movement,
- 5 using a sufficiently accurate external device, measure the distance travelled by the moving part (**Observed Distance**).
- 6 change to measurement mode DRV\_OFF

In the adjustment screen

- 7 press the Correction button. The following dialog box appears

- enter the distance to travel in the **Theoretical Distance** field (eg : 300000 $\mu$ m)
  - enter the actual distance in the **Observed Distance** field (eg : 293000 $\mu$ m)
- 8 press the **OK** button to start automatic calculation of the resolution. The new Distance and Number of Points values are then recalculated.

Repeat operations 2, 3, 4 and 5.

- If the measured distance is lower than that required, the adjustment is terminated. If not, perform a new correction (operations 7 and 8).

#### Caution

**If the Initial Resolution and VMAX parameter values need to be modified once the adjustment has been made as outlined above, a further adjustment must be made.**

**In general, any modification to the configuration in offline mode will require a further adjustment of the resolution in online mode.**



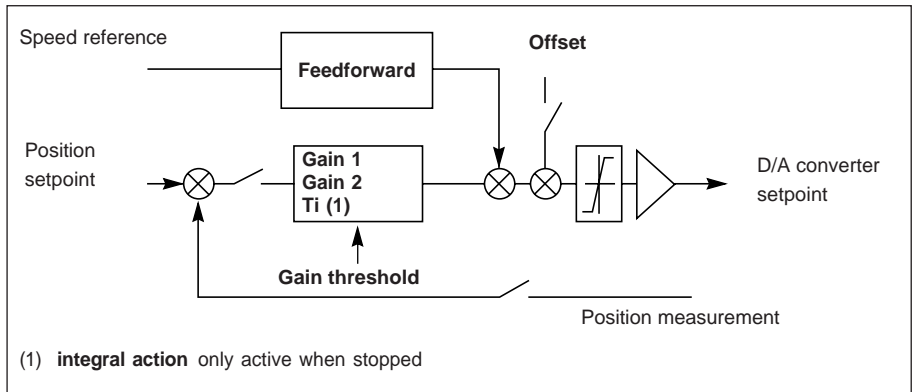
### 5.3-4 Adjusting the servo parameters

The following parameters are used to adjust the position loop.

Position loop			
Gain 1	<input type="text" value="1 000"/>	/100s	Offset <input type="text" value="0"/> mV
Gain 2	<input type="text" value="1 000"/>	/100s	Feed forward <input type="text" value="10"/> %
Threshold 1-2	<input type="text" value="500"/>	/1000 of Smax	Ti <input type="text" value="0"/> ms

### Description of the position control loop

#### Block diagram



### Calculating the references

The position and speed references are calculated as a function of the movement (speed, target position) required by the user and the parameters defined in the parameter adjustment screen.

## Description of the servo parameters

**Gain 1 and Gain 2** : Position loop gains (from 50 to 12 000 1/100s),

**Gain threshold** : change of gain threshold (from 20 to 500 % of Vmax),

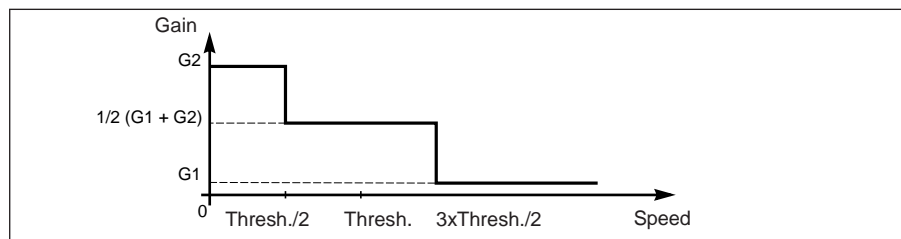
Default value Gain 1 and Gain 2 = 1000 1/100s, and gain threshold = 500 % of Vmax

Two gain values are used by the axis control module :

- **Gain 1** : gain value for high working speeds. This value prevents overshoots and instability.
- **Gain 2** : gain value for low working speeds in order to obtain very low position errors,

The position gain is applied as follows :

- if current speed  $\bullet 3 \times \text{Threshold} / 2$  : **Gain 1**
- if  $3 \times \text{Threshold} / 2 >$  current speed  $\bullet \text{Threshold} / 2$  : **Gain = (Gain 1+Gain 2)/2**
- if current speed  $< \text{Threshold} / 2$  : **Gain 2**



Using this **Gain** adjustment parameter, the module calculates the proportional gain coefficient  $KP$  :  $KP = C \times U_{MAX} \times \text{Gain}$

$C$  : constant, and  $U_{MAX}$  : value of the speed drive setpoint to achieve speed  $V_{MAX}$  ( $U_{MAX} < 9V$ ).

**Note** : generally Gain 1 = Gain 2.

**Feedforward** : feedforward gain (from 0 to 100 %),

Default value = 10%

This is expressed as a percentage. 100% corresponds to the value which would totally absorb the position error at constant speed for a speed drive which has no continuous error.

When the **Feedforward gain** increases, the position error decreases, but there is a resulting risk of an overshoot, including when approaching the stop point. It is therefore necessary to find a compromise.

### Note

In some cases, the position error passes a minimum with a possible change of sign when the **Feedforward gain** increases.

**Offset** : offset added to the value of the analog output calculated by the loop.

Min limit : -250mV

Max limit : +250mV

Default value = 0 mV

### Procedure for determining the servo parameters

In order to adjust the servo loop, specific values must be given to certain operating parameters, assuming that the values of the others correspond to the application.

Enter these parameters in the adjustment screens and confirm them in order to transmit them to the axis control module.

### Initial operation

This operation consists of setting a forced reference point (in manual mode). Forcing the reference point means that the axis is referenced from the time the application first started and thus that the following controls and functions are active :

- soft stops,
- return of moving part from soft stop overshoot.

#### Note :

Operation will only be correct if the direction of movement of the moving part is the same as the direction of measurement.

### Procedure for setting a forced reference point

- select the TSX CAY debug screen,
- select **Manual** mode,
- acknowledge the faults with the **Ack.** command,
- using an external device, measure the position of the moving part in relation to the reference point cam (not an exact measurement),
- set a forced reference point :
  - enter the measured value with its sign as the reference position value in the **Param** field,
  - select the **Forced Reference Point** command,

### Adjusting the gain at high speed (value of parameter **Gain 1**)

Since the moving part has an inertia equal to the maximum value found in the application :

- perform movements from one position, 1, to another position, 2, and vice versa. To do this :
  - select a low speed : by selecting a low value for coefficient **CMV**,
  - enter the value of the movement in the **Param** field,
  - then activate, one after the other, commands **Inc+** (position 1) and **Inc-** (position 2),
- check the position error when the moving part is stationary,

- adjust **Gain 1** to obtain an acceptable error while maintaining adequate stability (otherwise recheck the definition of the machine). Transfer each new value of **Gain 1** entered, and confirm in the parameters screen.
- select a high speed : by choosing a high value for coefficient **CMV**,
- perform movements from position 1 to position 2 and vice versa, and if necessary readjust **Gain 1**.

#### **Adjusting the gain at low speed** (value of parameter **Gain 2**)

This adjustment should be performed for machines which are subject to friction. If they are not, retain the value of **Gain 1** for parameter **Gain 2**. Set **Gain 2** above **Gain 1** to obtain a higher gain at low speed. Transfer the value and confirm in the adjustment screen :

- perform movements from one position, 1, to another position, 2, and vice versa. To do this :
  - select a very low speed of movement (low value of speed correction coefficient **CMV**),
  - enter a low movement value in the **Param** field,
  - then activate, one after the other, commands **Inc+** (position 1) and **Inc-** (position 2),
- check the position error when the moving part is stationary,
- adjust **Gain 2** to obtain an acceptable error while retaining adequate stability. Transfer each new value of **Gain 2** entered, and confirm in the adjustment screen.

**Adjusting the gain threshold** : the gain threshold is selected at the speed above which friction is eliminated.

#### **Adjusting the feedforward gain**

- perform movements from position 1 to position 2 and vice versa at speed **VMAX** and display the position error when the moving part is moving at constant speed. To do this :
  - select a high speed of movement (high value of speed correction coefficient **CMV**)
  - enter a movement value in the **Param** field,
  - then activate, one after the other, commands **Inc+** (position 1) and **Inc-** (position 2),
- adjust the **feedforward gain** to the required error value and sign,

#### **Note :**

In the event of too large an overshoot, it may be necessary to reduce the **feedforward gain** slightly.

#### **Adjusting the offset**

With the moving part stationary, change to direct drive (**DIRDRIVE**) mode and adjust the offset in the **-250mV** and **+250mV** window so that any slip of the moving part is eliminated.

---

### Adjusting the integral action

The role of this coefficient is to compensate for the different "offsets" in the system (module, speed drive, motor, mechanical parts) and their drift.

Compensation is obtained by using the **Ti** integral action adjustable parameter in the position loop.

This gain is only active when the axis is theoretically stopped (zero theoretical speed, deviation reduction phase). It is active in automatic and manual modes if there are no blocking faults on the axis (**AX\_OK =1**).

It is not active in **EXT\_CMD** and **SLAVE** automatic modes.

The principle consists of adding a permanent additional action and updating during stop phases.

The integral action is expressed in ms, in the interval [100; 5000] ms, where 0 (default value) indicates that there is no integral action.



## Command parameters

### Upper and lower soft limits : for a limited axis

Upper and lower limits of the position measurement which the moving part should not cross. In the event of an overshoot, the moving part stops with a soft stop fault.

$LMIN \leq SL\_MIN < SL\_MAX \leq LMAX$

and  $SL\_MAX - SL\_MIN > RESOL \times 256$

Default values :  $SL\_MIN = LMIN$  and  $SL\_MAX = LMAX$

2 3

### Modulo : for an infinite axis

Measurement evolution range for machines with infinite travel.

In the case of infinite axes, the adjustment must be less than or equal to the configuration modulo called "**Max modulo**".

Limits :  $Modulo \leq Max\ modulo$

Default modulo value :  $Modulo = Max\ modulo$

**Acceleration** : acceleration and deceleration values. This is defined by the time  $T_{acc}$  (in ms) taken to change from zero speed to speed  $V_{MAX}$  with a rectangular profile.




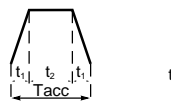

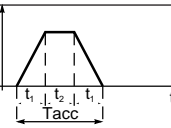

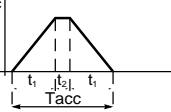

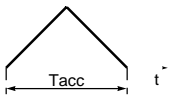
In the case of another profile it is given by the relationship :

$T_{acc} = T_{accrec} \times (2t_1 + t_2) / (t_1 + t_2)$  where  $T_{accrec}$  (value to be entered by the user) is the acceleration for a rectangular profile, and  $t_1$  and  $t_2$  are defined by the diagrams below.

Limit :  $T_{accMIN} - T_{acc} - 10000\ ms$  (where  $T_{accMIN}$  is the maximum acceleration).

Default value :  $T_{accrec} = T_{accMIN}$

**Acceleration profile** : acceleration profile applied to the moving part (rectangle by default).

Possibilities	Icon	Description
Rectangle		 <p><math>t_1=0</math> <math>T_{acc}=T_{accrec}</math></p>
Trapezoid 1		 <p><math>t_2=3t_1</math> <math>T_{acc}=1.25T_{accrec}</math></p>
Trapezoid 2		 <p><math>t_1=t_2</math> <math>T_{acc}=1.5T_{accrec}</math></p>
Trapezoid 3		 <p><math>t_1=3t_2</math> <math>T_{acc}=1.75T_{accrec}</math></p>
Triangle		 <p><math>T_{acc}=2T_{accrec}</math></p>



## Stop controls

**Delay** : as soon as the value of the speed reference calculated by the module equals 0, the module starts a time delay (equal to the parameter **Delay**). When the time delay has elapsed, the axis control module compares the measured speed of the moving part with the stop speed.

Stop time : 0 to 10 000 ms (default value : 500ms)      0 = no control of stop default

**Stop speed** : speed from which the moving part is considered to be stationary.

Stop speed : 0 to VMAX/10 and limited to 30000 (default value : min (VMAX/2, 30000))

**Target window** : tolerances on the position reached by the moving part, after the time delay defined in the Delay parameter.

Target window value :      0 to (SL\_MAX - SL\_MIN)/20 for a limited axis  
    0 = no control

Default value :              (LMAX - LMIN)/100

Target window value :      0 to Modulo/20 for an infinite axis

Default value :              Max modulo/100

### Principle for adjusting the error control parameters :

- enter the required error control parameter values, then confirm these parameters,

In the debug screen :

- select manual mode,
- select a high speed of movement
- perform movements from position 1 to position 2 and vice versa. To do this :
  - enter a movement value in the **Param** field,
  - then select, one after the other, commands **Inc+** (position 1) and **Inc-** (position 2).

The module should not change to error mode : check in the "Errors" box that the **Axis** fault is not indicated (or to obtain more Details press the **DIAG** button).

If a fault is detected :

- increase the values of the parameters (higher tolerances),
- or readjust the servo parameters then adapt the error control parameters.

Adjust the following parameters one after the other :

- **Deviation 1 and deviation 2,**
- **Stop speed and stop delay**

The speed should be below the **Stop speed** at the end of the **Stop delay**.

The **Stop delay** is counted with respect to the moment when the position reference reaches the required position value.

- **Target window,**
- **Overspeed**

for this adjustment select a speed of movement equal to  $V_{max}$ .

### 5.3-6 Manual mode parameters

**Speed** : speed of movement of the moving part in manual mode.

As in automatic mode, the actual speed of movement is adjusted by the correction coefficient CMV.

The actual speed reference= speed x CMV/1000

Limits : 10 to VMAX (0 = default value)

**Reference point value** : value loaded in the current position when the reference point is set manually

Limits :                 $SL\_MIN + 1$  to  $SL\_MAX - 1$  for a limited axis  
                               (default value =  $(SL\_MAX - SL\_MIN)/4 + SL_{MIN}$ )  
                               1 to Modulo - 1 for an infinite axis  
                               default value Modulo/4.

### 2 5.3-7 Parameters associated with master/slave axes

The link between slave and master axes is defined by a ratio and an Offset.

$\text{SlavePositionSetpoint} = \text{MasterPosition} \times (\text{Ratio1}/\text{Ratio2}) + \text{SlaveOffset}$ .

If the type of slave machine is infinite, the Modulo operator with ModuloValue (%MDxy.i.33) is applied when calculating the slave position setpoint.

The ratio defined by :

Ratio = **Ratio1/Ratio2** can be adjusted.

The dynamic values of the ratio vary in the interval [0.01 , 100]. The ratio can be negative.

**SlaveOffset** : position offset value between master and slave.

The **SlaveOffset** adjustable parameter can be used to create an offset between master and slave.

If the SlaveOffset parameter is zero :

The setpoint of the slave axis = Ratio x Setpoint or Measurement of the master axis.


The limits of SlaveOffset  $[-(2^{30-1}); 2^{30-1}]$ , the total must remain within the slave soft stop limits.

#### Note

TSX CAY •1 modules can be used to execute master/slave applications, but only without Offset and with a ratio which cannot be modified (see section 3.3-14).

## 5.4 Confirming and saving adjustment parameters

### 5.4-1 Confirming

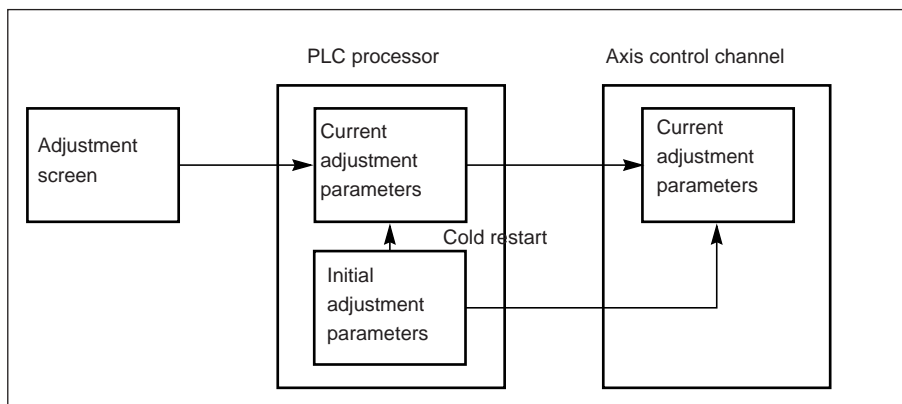
When the adjustment parameters have been entered, confirm these parameters using the **Edit/Confirm** command or activate the  icon.

If one or more of the parameter values are not within permitted limits, an error message appears indicating the parameter concerned.

Correct the incorrect parameter(s) then confirm.

If no configuration parameter has been modified, modifying adjustment parameters does not interrupt the operation of the axis but modifies its behavior.

Current parameters are modified in this way (the initial parameters remain unchanged).



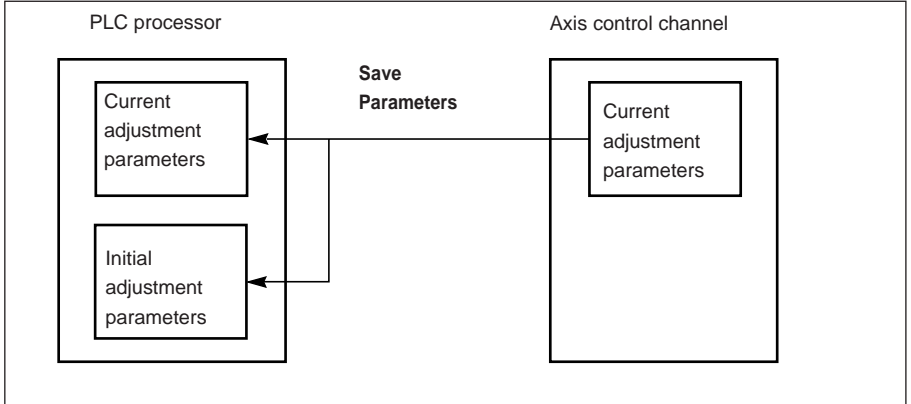
#### Note :

On a cold restart the current parameters are replaced by the initial parameters.

The initial parameters can be updated by the save command (see next page) or by a reconfiguration operation.

### 5.4-2 Save

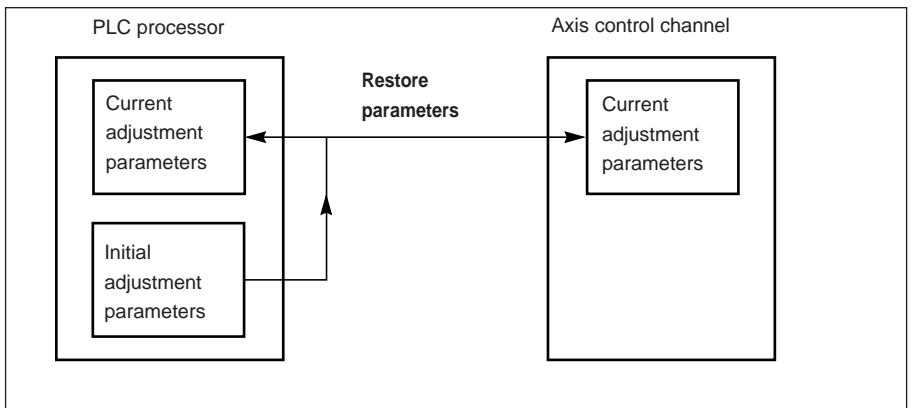
To save the current parameters (update the initial parameters) activate the **Utilities/Save Parameters** command.



Note : the SAVE\_PARAM instruction enables the application to perform this save operation.


### 5.4-3 Restore

The **Utilities/Restore Parameters** command replaces the current parameters with the initial values.

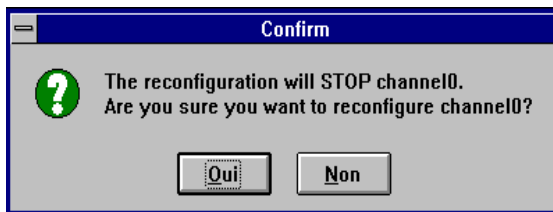


Note : the RESTORE\_PARAM instruction enables the application to perform this restore operation. This operation is also performed automatically on a cold restart.

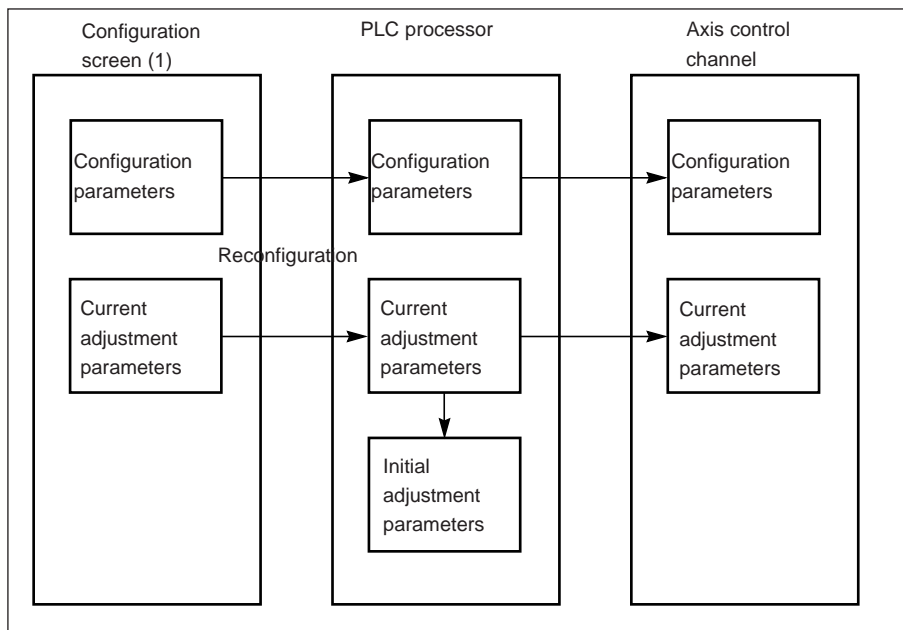
## 5.5 Reconfiguration in online mode

When the configuration parameters have been modified, confirm these parameters using the **Edit/Confirm** command or activate the  icon.

Only those parameters which are not grayed out can be modified in online mode. The other parameters (resolution, encoder type, activation of an event-triggered task) must be modified in offline mode. However, on reconfiguration, the corrected resolution becomes the initial resolution.



**Any reconfiguration in online mode stops the operation of the channel concerned, and thus the current movement.**



(1) or the adjustment screen if a configuration parameter has already been modified in the configuration screen

## 6.1 Principle of debugging an axis control program

Axis control, which is integrated in the PL7 program, uses the PL7 debug functions.

### Reminder of the options provided by PL7 :

- realtime display and animation of the program,  
In Grafcet : as each movement is programmed in one Grafcet step, it is easy to identify the current movement,
- insertion of breakpoints and execution cycle by cycle, rung by rung or statement by statement,
- access to the animation tables, which enables status bits and words to be displayed and the command bits of the SMOVE function to be controlled. It also enables bit objects to be forced, and the evolution of the Grafcet chart to be blocked.

PL7 software offers a debug screen which is specific to axis control modules and which provides the user with all the necessary data and commands.

### TSX CAY axis control function debug screens

The screenshot shows the TSX CAY 21 [RACK 0 POSITION 4] interface. It features a 'Manu' mode selector with 'Auto' and 'Dir Drive' options. A 'Global Unforcing' button is also present. The 'Channel zone' displays movement data for X and F axes, including actual and target values, following error, and setpoint. It also shows position and speed indicators. The 'Control zone' includes a 'STOP' button (F8) and various command options such as JOG-, JOG+, INC-, INC+, Manual reference point, Forced reference point, and Auxiliary output. The 'Axis' and 'Faults' sections provide status and error information.

This screen comprises 3 zones :

- module zone,
- channel zone,
- control zone for the moving part and the program. It depends on the operating mode selected via the mode switch : Automatic (Auto), Manual (Man), Direct Drive (DirDrive) or Measurement (Off).

---

## 6.2 Debug screens

---

### 6.2-1 Accessing the debug screens

The terminal must be in online mode.

Access the axis control module debug screens by selecting the **Configuration** editor and selecting and confirming (or double-clicking) the position in the rack containing the axis control module.

In online mode, the debug screen is selected by default.

---

### 6.2-2 User interface

**Command buttons :** 

- For commands on a state (1) :  
Pressing and then releasing the button activates the associated command, the indicator lamp in the button is on when this command is accepted (the corresponding command bit %Q is set to 1).


Pressing the button a second time and then releasing it deactivates the command, and the indicator lamp in the button goes off when this command is accepted (the corresponding command bit %Q is set to 0).

- For commands on an edge :  
The command is activated as soon as the button is pressed and then released. The indicator lamp in the button goes on and then off automatically.

The indicator lamp beside the button corresponds to the command being accepted by the module.

(1) except JOG commands.

**Entry field :**

All values entered in entry fields must be confirmed with the  key.

**Keyboard :**

**Shift F2** : moves from one zone to another.

**Tab** : moves from one set of commands to another within one zone.

**Arrow keys** : moves from one command to another within one set of commands.


**Space key** : activates or deactivates a command.

### Warning

There could be "conflicts" between the PL7 program which executes commands or writes variables, and the commands executed from the debug screen. The last command to be accepted is that which takes priority.



**Note**

The **Utilities/Stop Animation** command or the  icon stops the animation in the display zones and inhibits the command buttons.

The **Utilities/Animate** command or the  icon reactivates the animation.

**6.2-3 Description of the debug screens**

The debug screens have a common header, comprising module and channel zones.



**Module zone**

Indicator lamps	Status	Meaning
<b>RUN</b>	On	Module operating
<b>ERR</b>	On Flashing	Module off Communication fault
<b>I/O</b>	On	External hardware fault (encoder, speed drive, outputs)
<b>DIAG</b>	On	Fault on the module; pressing this button (a button is associated with the indicator lamp) displays a module diagnostics dialog box which gives the source of the fault.

## Channel zone

In addition to the "Select Axis" and "Function" fields (common to all screens), this zone contains the following commands and indicator lamps :

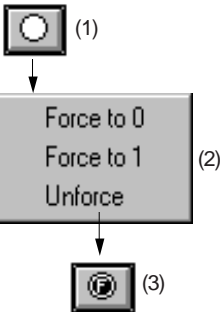
Command	Role
---------	------



Operating mode selection button. To access another mode click on the text of the mode you wish to select (or click as many times as necessary on the button). Using the keyboard : select the button using the **Tab** key and press as many times as necessary on the **space** key.

Modes can also be accessed from the **View** menu. When the selected mode is accepted by the module, the movement control zone in the required mode is displayed.

**Caution** : although it has been selected, the chosen mode may not be accepted by the module channel (for example if the PLC is stopped).



Forcing command menu. If an object can be forced, a click with the right-hand mouse button on the corresponding button (1) displays a menu (2) which accesses the forcing commands : **Force to 0**, **Force to 1** or **Unforce**.

After selecting the command by clicking on it, forcing is applied and the forcing status is displayed on the button (3) :

- F for forcing to 0,
- F in reverse video for forcing to 1.

The **Global Unforcing** button in the module zone unforces all the forced objects.

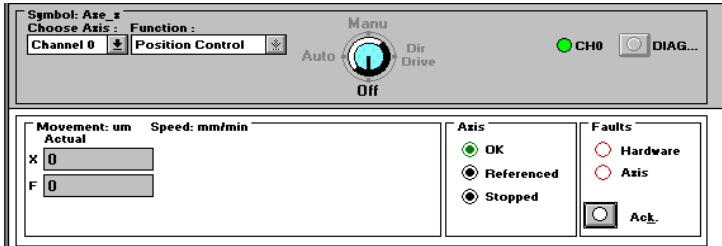


This zone displays in summary form (with a historic on some points, like an oscilloscope) the value of the analog output (between +10V and -10V).

<b>CHI</b>	On	Axis (channel) configured and not faulty
	Flashing	Axis fault
	Off	Axis not configured
<b>DIAG</b>	On	Channel fault; pressing this button (a button is associated with the indicator lamp) displays a channel diagnostics dialog box which gives the source of the fault (see section 6.3 Diagnostics).

### 6.2-4 Measurement mode (Off)

In this mode the axis control channel only feeds back current position and speed data. The movement of the moving part is not controlled. The position loop is off and the speed drive enable relay is unlocked whatever the state of the speed drive enable bit ENABLE (%Qxy.i.9).



#### Description of the information displayed

Movement : / Speed :

<b>X</b>	displays the position of the moving part in the unit of measurement defined at configuration
<b>F</b>	displays the speed of the moving part in the unit of measurement defined at configuration

#### Axis

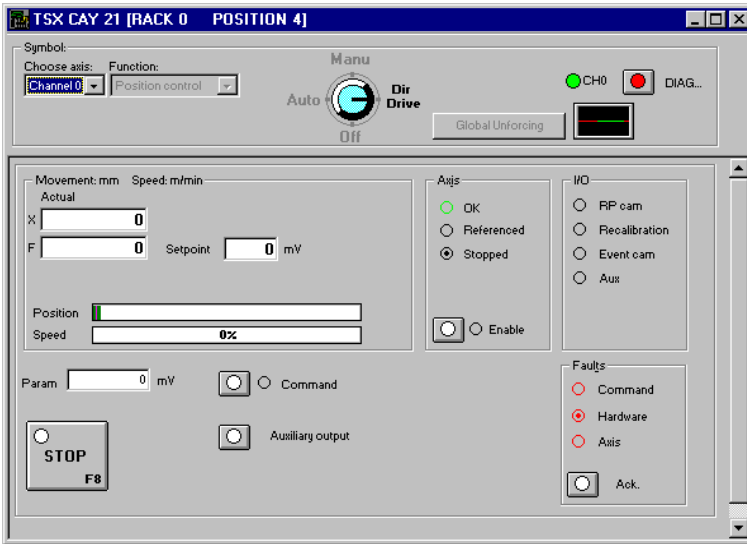
Indic. lamps	Status	Meaning
<b>Configured</b>	on	axis configured
<b>OK</b>	on	axis in operating state (no blocking fault)
<b>Referenced</b>	on	axis referenced
<b>Stopped</b>	on	the moving part is stopped

#### Faults

Indic. lamps Buttons	Status	Meaning
<b>Hardware</b>	on	external hardware fault (encoder, speed drive, outputs, etc)
<b>Axis</b>	on	application fault (deviation, soft stop fault, etc)
<b>Ack</b>		fault acknowledgment button (all faults which have disappeared are acknowledged).

## 6.2-5 Direct drive mode (Dir Drive)

In direct drive mode the voltage is used to control movement of the moving part with the servo loop off.



### Description of the commands and information displayed

Movement:/Speed

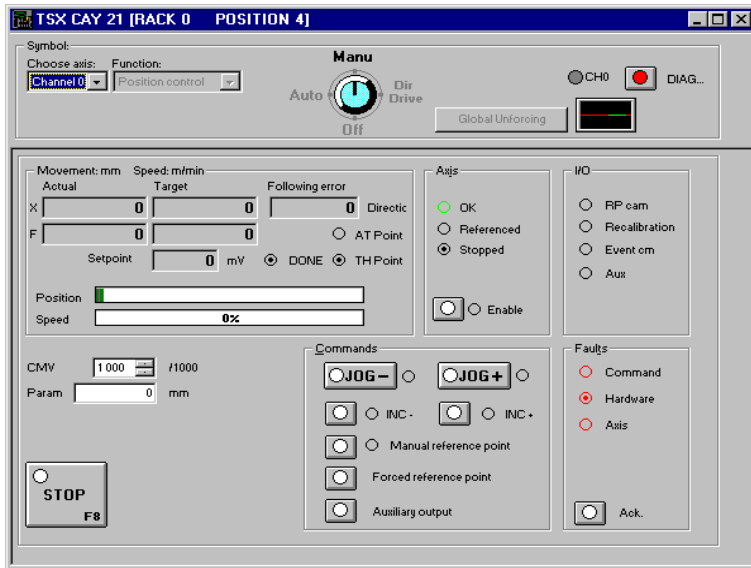
<b>X</b>	Displays the position of the moving part in the unit of measurement defined at configuration
<b>F</b>	Displays the speed of the moving part in the unit of measurement defined at configuration
<b>Setpoint</b>	Displays the value in mV of the setpoint applied to the analog output
<b>Position</b>	Bargraph showing the evolution of the moving part between the soft limits. It is red if there is an overshoot of the 100% values, otherwise it is green.
<b>Speed</b>	Bargraph showing the speed of the moving part as a % of the maximum speed. It is red if there is an overshoot of the 100% values, otherwise it is green.

Axis, I/O and fault zones see section 6.2-4, I/O see section 6.2-6. The Axis zone also contains the **Enable** command which controls the speed drive enable relay.

Command	Role
<b>STOP</b>	Sets the analog output to 0 while respecting the deceleration
<b>Param</b>	Enables the setpoint value, -9000 mV to +9000 mV, to be entered
<b>Command</b>	Applies the value entered in the <b>Param</b> field to the analog output
<b>Auxiliary Output</b>	Sets the auxiliary output to 0 or 1

### 6.2-6 Manual mode (Man)

In manual mode, movement of the moving part is controlled directly from the debug screen, using elementary commands JOG+, JOG-, INC-, etc.



#### Description of the commands and information displayed

##### Movement/Speed

<b>X Current</b>	Displays the position of the moving part in the unit of measurement defined at configuration
<b>X Target</b>	Displays the target position for the moving part to reach
<b>X Deviation</b>	Displays the difference between the calculated theoretical position and the actual position of the moving part (deviation).
<b>F Current</b>	Displays the speed of the moving part in the unit of measurement defined at configuration
<b>F Target</b>	Displays the speed reference which the moving part is to reach (manual speed corrected by coefficient CMV).
<b>Setpoint</b>	Displays the value (in mV) of the setpoint applied to the analog output
<b>Position</b>	Bargraph showing the evolution of the moving part between the limits defined in the configuration screen. It is red if there is an overshoot of the 100% values, otherwise it is green.
<b>Speed</b>	Bargraph showing the speed of the moving part as a % of maximum speed. It is red if there is an overshoot of the 100% values, otherwise it is green.

Indicators	Status	Meaning
<b>+ direction</b>		Indicates a mvt. of the moving part in a positive direction
<b>- direction</b>		Indicates a mvt. of the moving part in a negative direction
<b>AT Point</b>	on	Indicates that the current movement has been completed, and The moving part is in the target window (with commands INC_P or INC_M).
<b>TH Point</b>	on	Indicates that the theoretical setpoint has been reached
<b>DONE</b>	on	Indicates that the current movement has been completed

Axis and faults (see section 6.2-4). The Axis zone also contains the **Enable** command which is used to control the speed drive enable relay.

I/O

Indicator lamps	Meaning
<b>RP cam</b>	State of the signal (0 or 1) at the "Reference point" input
<b>Recalibration</b>	State of the signal (0 or 1) at the "Recalibration" input
<b>Evt cam</b>	State of the signal (0 or 1) at the "Event" input
<b>Aux</b>	State of the signal (0 or 1) at the auxiliary output

1 = indicator lamp on, 0 = indicator lamp off

Command	Role
<b>CMV</b> in	Field for entering the speed multiplication coefficient with a value of 0 to 2000 steps of 1/1000 (or 0.000 to 2.000).
<b>Param</b>	For entering the value of an incremental movement (INC+/INC- command) or a forced reference point.
<b>STOP</b>	Causes the moving part to stop according to the deceleration defined at configuration

---

Commands (see section 4.7-3)

---

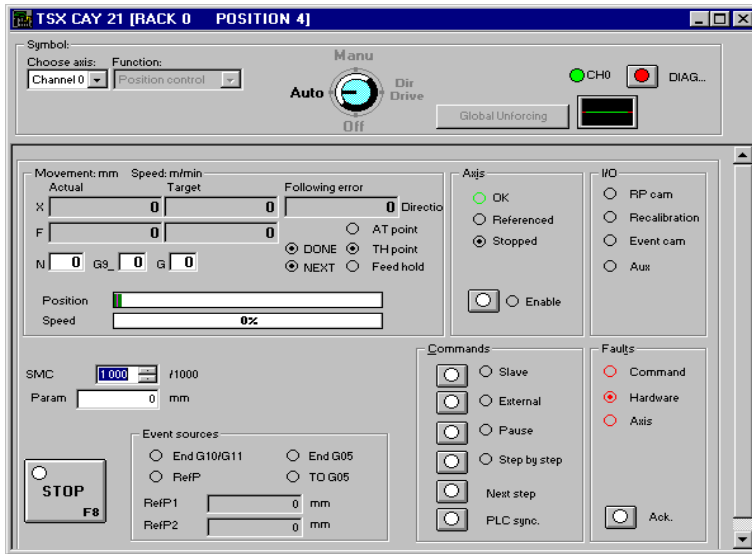
Command	Role
<b>JOG+</b>	Command for visual movement in positive direction (1).
<b>JOG-</b>	Command for visual movement in negative direction (1).
<b>INC+</b>	Command for incremental movement in positive direction for a distance entered in the <b>Param</b> field.
<b>INC-</b>	Command for incremental movement in negative direction for a distance entered in the <b>Param</b> field.
<b>Manual reference point</b>	For an incremental encoder, command to set a manual reference point. The current position takes the "RP Value" defined in the adjustment screen, having found the reference point cam which conforms to the type defined at configuration. The type of reference point is defined in the configuration screen.
<b>Forced reference point</b>	For an incremental encoder, command to set a forced reference point. The current position is forced to the value entered in the <b>Param</b> field. This type of reference point does not lead to any movement of the moving part.
<b>Clear reference</b>	For an absolute encoder, command to change the axis to a non-referenced state in order to move the moving part without a soft stop fault.
<b>Referencing</b>	For an absolute encoder with direct offset, command to change the axis to referenced state.
<b>Offset calculation</b>	For an absolute encoder with assisted offset, triggers the calculation of the encoder offset in order to make the current position coincide with the value in units of length entered in the <b>Param</b> field. The axis is referenced at the end of this calculation.
<b>Auxiliary output</b>	Sets the auxiliary output to 1 or 0.

(1) These commands remain active as long as the button is pressed.

These commands are also used to release the moving part from a soft stop overshoot (after acknowledgment of the fault).

## 6.2-7 Automatic mode (Auto)

Automatic mode is that in which the SMOVE functions are executed.



### Description of the commands and information displayed

Movement / Speed :

<b>X Current</b>	Displays position of moving part in the unit of measurement defined at configuration
<b>X Target</b>	Displays the target position for the moving part to reach (defined in the instruction) (1)
<b>X Deviation</b>	Displays the difference between the calculated theoretical position and the actual position of the moving part (deviation).
<b>F Current</b>	Displays speed of moving part in the unit of measurement defined at configuration
<b>F Target</b>	Displays the speed reference which the moving part is to reach (speed defined in the instruction corrected by coefficient CMV).(2)
<b>N G9 G</b>	Displays the instruction being executed. N = step no., G9 = type of movement, G = instruction code
<b>Position</b>	Bargraph showing evolution of the moving part between the limits defined at configuration. It is red if there is an overshoot of the 100% values, otherwise it is green.
<b>Speed</b>	Bargraph showing the speed of the moving part as a % of maximum speed. It is red if there is an overshoot of the 100% values, otherwise it is green.

Note : the number of display digits is limited to 10. For any value longer than 10 digits a series of points will be displayed.

(1) displays the number of memorizations (1 or 2) in the case of instruction G07

(2) displays the time period in the case of instruction G05



Indicators	Status	Meaning
<b>NEXT</b>	On	Indicates that the module is ready to receive a movement command
<b>DONE</b>	On	Indicates completion of the current movement(s)
<b>+ direction</b>		Indicates a mvt. of the moving part in a positive direction
<b>- direction</b>		Indicates a mvt. of the moving part in a negative direction
<b>AT Point</b>	On	Indicates that the current movement has been completed, and the moving part is in the target window (for instructions with stop).
<b>TH Point</b>	On	Indicates that the theoretical setpoint has been reached
<b>Immediate pause</b>	On	Indicates that the Immediate pause function has been activated (CMV coefficient set to 0).

Axis and faults (see section 6.2-4). The Axis zone also contains the **Enable** command which is used to control the speed drive enable relay.

I/O

Indicator lamps	Meaning
<b>RP cam</b>	State of the signal (0 or 1) at the "Reference point" input
<b>Recalibration</b>	<b>State of the signal (0 or 1) at the "Recalibration" input</b>
<b>Evt cam</b>	State of the signal (0 or 1) at the "Event" input
<b>Aux</b>	State of the signal (0 or 1) at the auxiliary output

1 = indicator lamp on, 0 = indicator lamp off

Command	Role
<b>CMV</b>	Field for entering the speed correction coefficient with a value of 0 to 2000 in steps of 1/1000 (or 0.000 to 2.000).
<b>Param</b>	Used to enter external values (position follower function).
<b>STOP</b>	Causes the moving part to stop depending on the deceleration defined at configuration.

## EVT (Event) sources

Indicator	Status	Meaning
<b>PRef</b>	On	indicates memorization of position PRef (1)
<b>PRef1</b>		displays memorized position PRef1 (1)
<b>PRef2</b>		displays memorized position PRef2 (1)
<b>End G05</b>	On	indicates the end of execution of instruction G05
<b>TO G05</b>	On	indicates that the TIME OUT defined in instruction G05 has elapsed
<b>End G10/G11</b>	On	indicates the occurrence of the event during execution of a G10 or G11 instruction.

(1)As long as event-triggered processing has been assigned to command G07.

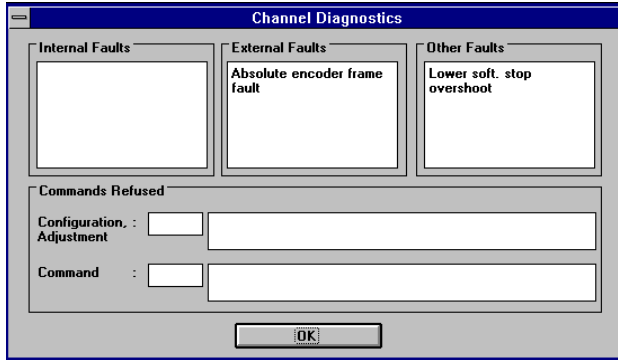
*There is no indicator associated to the "Module crossing" event.*

## Commands

Command	Role
<b>Slave</b>	Changes the axis to slave mode (follower of another axis). Axis 0 cannot be changed to slave mode.
<b>External command</b>	Changes the axis to follower of a periodic setpoint mode.
<b>Pause</b>	Command to stop the moving part at the end of the current movement with stop.
<b>Step by step</b>	Changes the axis to step by step mode.
<b>Next step</b>	Launches the waiting movement in step by step mode.
<b>Synchro CPU</b>	Initiates a CPU event

### 6.3 Diagnostics

In online mode the various Debug, Adjust and Configuration screens display the **DIAG** button, which gives detailed information on faults detected by the module.



- **Internal fault** : internal module fault which generally requires replacement of the module.
- **External fault** : fault originating in the operative part.
- **Other fault** : application fault.
- **Command failures** : the cause of the command failure and the message number are given in the field concerned. The list of command failure messages is given in section 11.

---

## 6.4 Archiving

---

When the program has been debugged in online mode :

- save the adjustment parameters if they have been modified, using the **Utilities/Save Parameters** command when the parameter adjustment screen is selected.
- save the PL7 application to disk, using the **File/Save** command.

---

## 6.5 Documentation

---

The documentation for the axis control application is included in the complete PL7 application documentation.

It contains :

- the program part,
- the CONFIGURATION and saved ADJUSTMENT parameters which have been saved.

## 7.1 Designing a man-machine interface

### 7.1-1 Control station

The programmer can use all the commands and elementary data in the form of command bits / words and status bits / words to design a simple or complex control station.

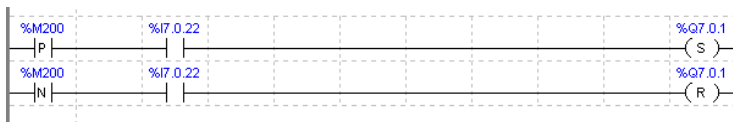
An example is given in section 1 of part B1 "Tutorial".

The programming principles are given in section 4.7, and an exhaustive list of all the bits and words is given in the Quick Reference Guide.

### 7.1-2 Man-machine interface on CCX 17

The following example enables a JOG+ visual movement to be performed in manual mode. It is also used to change the speed correction coefficient CMV on a CCX man-machine interface terminal.

This command or this modification can be performed either from the PL7 debug screen or from the CCX terminal.



%M200 corresponds to the status of the CCX terminal button activating a JOG+ command



The above instruction is used to update bit %M200 on each scan for detecting edges



%MW200 contains the coefficient setpoint CMV entered on the CCX terminal. %MW0 contains the last saved value of CMV entered on the CCX terminal.



---

## 8.1 Fault monitoring

---

The user has a number of means available to him for detecting faults :

- indicator lamps on the front panel of the module,
- diagnostic screens which can be accessed in online mode via the DIAG key from all the axis control module screens,
- debug screens,
- fault bits and status words.

---

## 8.2 Conditions for executing commands

---

### General conditions for movement commands (in auto or Man mode) :

- Axis configured and with no blocking fault,
- The speed drive enable command, ENABLE, must be active (bit %Qxy.i.9 at 1) and the STOP command not active.
- Automatic or manual mode selected as appropriate,
- For absolute position commands : this position should be between the limits SL\_MIN and SL\_MAX for limited axes and the values 0 and modulo-1 for infinite axes,
- For relative position commands : the target calculated based on the relative current position should be between limits SL\_MIN and SL\_MAX,
- The axes should be referenced except for reference point commands,
- Parameter F of the SMOVE function should be - VMAX:

### Modifying the speed correction parameter CMV

If a modification of parameter CMV implies a speed above VMAX, then the speed is limited to VMAX.

### Note :

If the "Sequence check" option was not selected during configuration, a movement without stop which is not followed by any sequencing command continues until the soft stop limits are reached.

---

## 8.3 Diagnostic help

---

This section describes the steps which should be taken in situations which the user may encounter.

**Symptom** : The TSX CAY module appears not to have taken account of the new parameters written by WRITE-PARAM.

**Diagnostics** : Program a READ\_PARAM into your application to find out the values actually used by the TSX CAY.

A WRITE\_PARAM triggered while another adjust exchange is in progress is ignored.

**Procedure to follow** : Test bit %MWxy.i.0:X2 before any adjust exchange.

**Symptom** : Event-triggered processing associated with the axis control channel is not executed.

**Diagnostics** : Check the event feedback circuit.

- Event number declared at configuration is identical to that of the event-triggered processing
- Source of the unmasked event (code M of command SMOVE)
- Events authorized at system level (%S38 = 1)
- Unmasked events at system level (UNMASKEVT())

**Procedure to follow** : see part F, common features of application-specific functions, on using events.

**Symptom** : Settings have been lost

**Diagnostics** : A cold restart loses the current adjustments made via the screen or a WRITE\_PARAM

**Procedure to follow** : Save the current adjustments using the "Save Parameters" function or the SAVE\_PARAM instruction.

**Symptom** : The operating status words %MWmy.i.1 and 2 are not consistent with the state of my axis control channel.

**Diagnostics** : These words are only updated on an explicit READ\_STS request

**Procedure to follow** : Program a READ\_STS into your application

**Symptom** : The "encoder supply" fault persists even though my encoder is supplied correctly and the current value is changing.

**Diagnostics** : The encoder supply feedback signal is incorrectly wired.

**Procedure to follow** : See the installation manual for the different ways of wiring this signal.



---

**Symptom** : The commands in the debug screen have no effect.

**Diagnostics** : The application or the task is in STOP mode.

**Procedure to follow** : Set the application or task to RUN.

**Symptom** : Some commands in the debug screen cannot be modified.

**Diagnostics** : These bits are written by the application.

**Procedure to follow** : Use bit forcing (for %Qxy.i.r objects) or design the application so that it does not write these bits automatically (modification on a transition and not on a state).

**Symptom** : It is not possible to enter more than 3 characters in the numerical fields in the adjustment and configuration screens.

**Diagnostics** : A thousands separator has not been selected in the Windows control panel.

**Procedure to follow** : in the control panel, select the "International" icon in the "Numbers format" field, activate the "Modify" command and select a thousands separator.

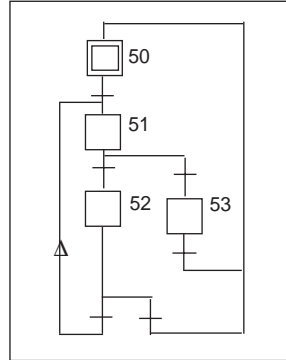


## 9.1 Teaching the positions

The PL7 program in the following example :

- teaches 16 positions in the first chart
- describes their use in the second chart

### Chart for teaching positions



#### STEP 50 ACTION ON ACTIVATION

<memorizes %MW99 in order to use it as a limit

```
!   %MW98:=%MW99;
```

<Initializes the index during the teach phase

```
!   %MW99:=-1;
```

TRANSITION: X50->X51

```
!   RE %I2.0
```

#### STEP 51 ACTION ON ACTIVATION

<updates the index

```
!   %MW99:=%MW99+1;
```

<teach positions

```
!   %MD200[%MW99]:=%ID7.0;
```

TRANSITION: X51->X52

```
!   %MW99<=16
```

TRANSITION: X51->X53

```
!   %MW99>16
```

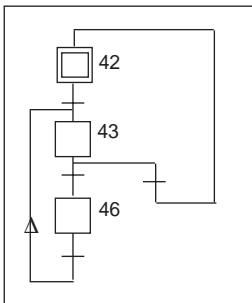
TRANSITION: X53->X50

```
!   RE %I2.1
```

TRANSITION: X52->X51

```
!   RE %I2.0
```

## Chart describing the use of the positions



## STEP 42 ACTION ON ACTIVATION

<initializes %MW97 as the execution index

```
!   %MW97:=-1;
```

## TRANSITION: X42-&gt;X43

```
!   RE %I2.2
```

## STEP 43 ACTION ON ACTIVATION

<increments the execution index

```
!   %MW97:=%MW97+1;
```

<executes the next segment

```
!   SMOVE %CH7.0(%MW97,%KW8,%KW1,%MD200[%MW97],150000,0);
```

%KW8 : 90 movement with absolute value

%KW1 : 09 go to point and stop

## TRANSITION: X43-&gt;X46

```
!   %I7.0.0 AND (%MW97<%MW98) AND NOT %I7.0.2
```

## TRANSITION: X43-&gt;X42

```
!   (%I7.0.1 AND(%MW97>=%MW98))OR %I7.0.2
```

## TRANSITION: X46-&gt;X43

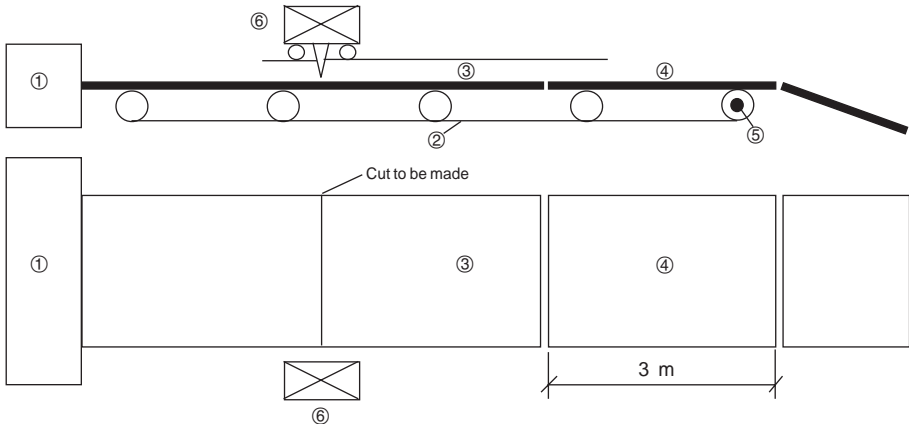
```
!   TRUE
```

## 9.2 Example of use for a TSX CAY module for cutting a metal sheet which arrives continuously

### 9.2-1 Description of the application

This example describes the principle of "on the fly" cutting for a metal sheet which arrives continuously on a cutter :

- the metal sheet exits the roller continuously at a speed of 1 m/s,
- a conveyor belt moves the sheet towards the cutter,
- the sheet is 2 m wide and must be cut into lengths of 3 m, with a cutting precision of 0.5 mm,
- when the position of the product is detected, the carriage which carries the cutter changes to position slave mode. Once it has been synchronized with the cut to be made, the carriage is governed by the movement of the belt and an internal discrete output triggers the cut,
- at the end of the cut, the tool rises and returns to its original position (wait position).

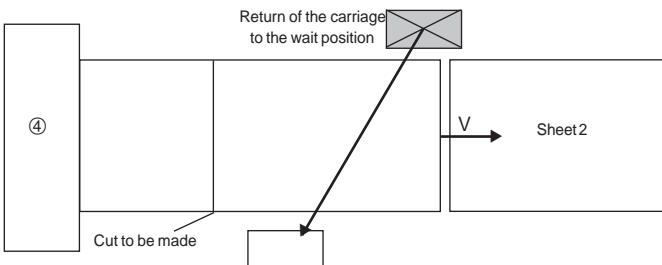
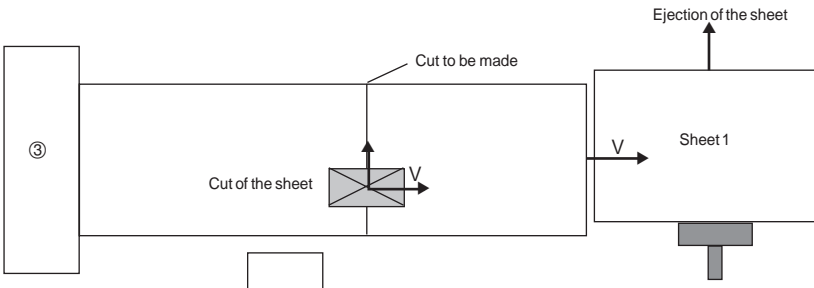
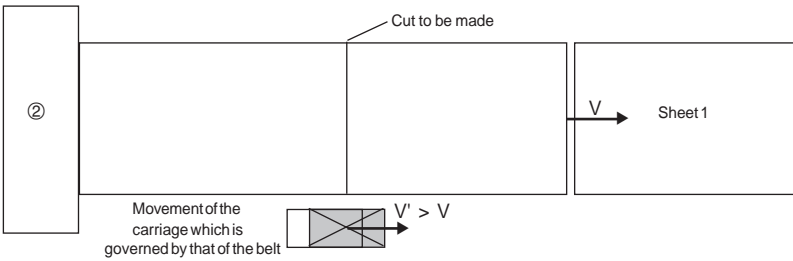
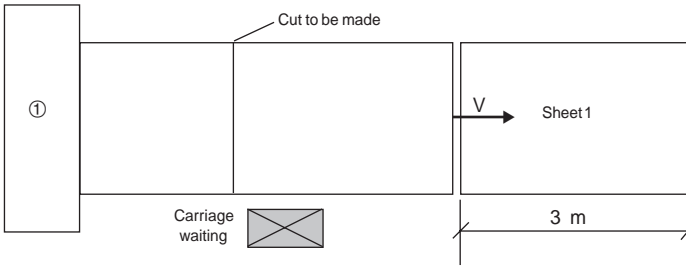


- |                          |                        |
|--------------------------|------------------------|
| ① Roller                 | ④ Cut sheet            |
| ② Conveyor belt          | ⑤ Incremental encoder  |
| ③ Continuous metal sheet | ⑥ Carriage with cutter |

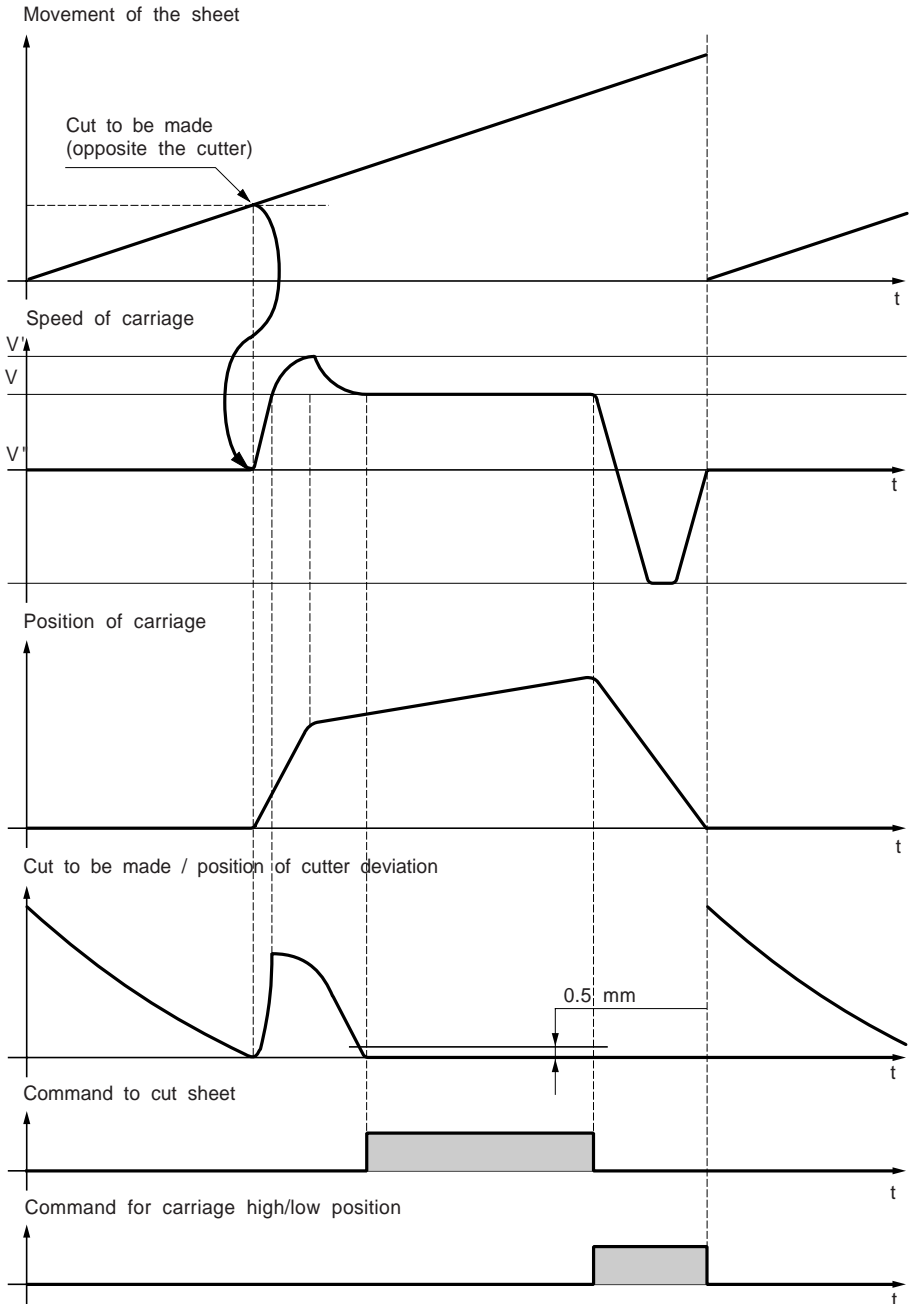
The various phases of the application are as follows :

- ① The carriage which carries the cutter is waiting for the next cut to be made (when 3 m of metal sheet has moved past the cutter).
- ② When the next cut to be made is in line with the tool, the carriage starts to move in the direction of the belt. At first, its speed is greater than that of the belt so that the cutter reaches the position where the cut is to be made (there is no anticipation of startup); its speed is then is governed by that of the belt.
- ③ As well as being is governed by the movement of the belt, the movement of the carriage is also perpendicular to the sheet so that the cut can be made.
- ④ When the cut is completed, the carriage rises and returns to its wait position.

## Various phases of the application



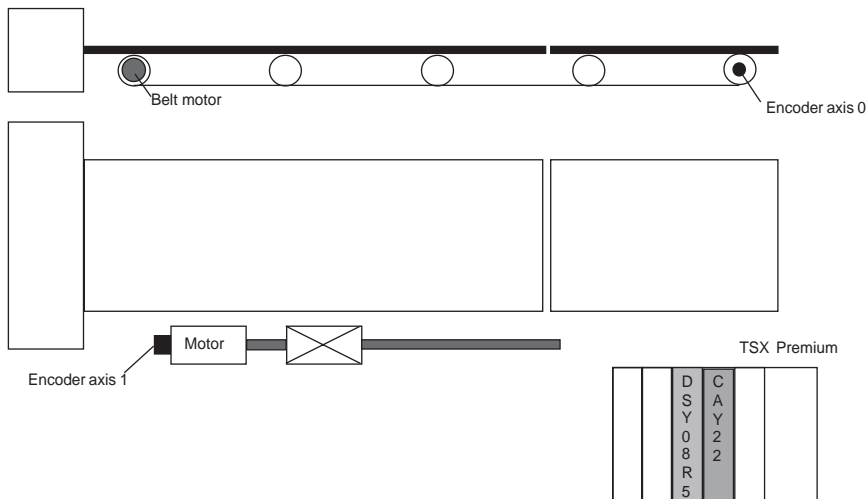
**Timing diagrams**



## 9.2-2 Configuring the application

This application can be configured using a TSX Premium PLC equipped with the following modules :

- a TSX CAY 22 module for controlling the movement of the belt and the carriage,
- a TSX DSY 08 R5 output module for controlling the cutting of the sheet and the high/low position of the carriage.



### Configuration of the TSX CAY 22 module

Channel	Axis 0	Axis 1
<b>Mode</b>	OFF (measurement)	AUTO <ul style="list-style-type: none"> <li>• Slave during phases 2 and 3 (catching the cut to be made and cutting the sheet),</li> <li>• Free during phase 4 (return of the carriage to the wait position).</li> </ul>
<b>Machine</b>	Infinite	Limited
<b>Input interface</b>	Incremental encoder Distance : 2 400 000 $\mu\text{m}$ No. of points : 40 000	SSI absolute encoder
<b>Modulo</b>	50 000 points 3 000 000 $\mu\text{m}$	
<b>Event</b>	EVT0	
<b>DMAX2</b>		500 $\mu\text{m}$ (precision of the cut)
<b>RATIO</b>		1
<b>Work zone</b>		1 000 000 $\mu\text{m}$ (0 to 1 m)
<b>Offset</b>		No automatic offset



### 9.2-3 Programming the application

**Initialization** : make a forced reference point setting for axis 0. To do this, the belt must be stopped (NOMOTION = 1).

```
IF %I103.0.8 THEN SMOVE %CH103.0(1,90,62,%MD1,0,0)
```

**Event-triggered task** : this task is **compulsory**. It makes it possible to change axis 1 to slave (SLAVE = 1) when the modulo crossing (EVT\_MOD = 1) is detected.

```
IF %I103.0.51 THEN SET %Q103.1.17
```

**Taking the length to be cut into account** (in POST-processing) :

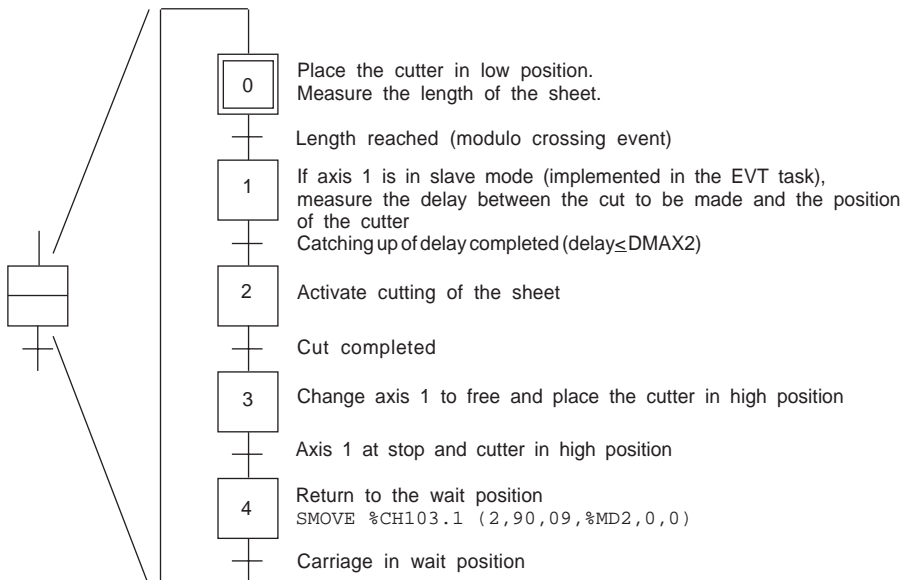
- The length to be cut is defined by the MODULO parameter (in encoder points). To modify this length, change the value of double word %MD103.0.31,
- If the carriage is in the wait position (phase 1), carry out a WRITE\_PARAM :

```
IF X0 THEN WRITE_PARAM %CH103.0
```

**Error handling** (in POST-processing) :

- During the cut (phase 3), if the deviation between the cut to be made and the position of the cutter is greater than DMAX2, then raise the cutter, return to the wait position and signal the error,
- In the event of an error on axis 0, raise the cutter, return to the wait position and signal the error,
- In the event of an error on axis 1, raise the cutter, return to the wait position and signal the error,

### Grafcet chart of the application





## 10.1 SMOVE programming function

```
SMOVE %CHxy.i(N_Run,G9_,G,X,F,M)
```

**%CHxy.i** = address of the axis control module in the PLC configuration

**x** = rack number

**y** = position of the module in the rack

**i** = channel number

**N\_RUN**= 0 to 32767, number identifying the movement performed by the SMOVE function. Identifies the current movement in debug mode.

**G9\_** = type of movement

**90** movement to an absolute position value

**91** movement to a relative value with respect to the current position

**98** movement to a relative value with respect to memorized position PREF (position PREF is memorized using instruction code G07)

**60** absolute movement in an imposed direction

**68** relative movement with respect to PREF in an imposed direction

**G** = instruction code,

**09** : move to the position and stop

**01** : move to the position without stopping

**32** : prepare machining command

**30** : simple machining

**10** : move until an event is detected and stop

**11** : move until an event is detected without stopping

**14** : reference point

**05** : await an event

**07** : memorize the current position when an event occurs

**62** : forced reference point

**21** : move without stopping with reference point on the fly on an event

**04** : stop a movement

**X** = coordinates of the position to be reached or towards which the moving part must move (in the event of a move to a position without stopping).

This position can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this word can be indexed).

The unit in which these values are expressed is defined by the configuration parameter **Length Units** (this parameter is set in the configuration screen) : mm (default unit),

**F** = speed of movement of the moving part. This speed can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this word can be indexed).

The unit of speed depends on the selected unit of position :

Speed = u x 1000/min where u = selected unit of length

Example : if the unit of length chosen is the mm (default unit) the unit of speed is : mm x 1000/min --> mm/min

M

- = Word coded on 4 four-bit bytes (in hexadecimal) 16# 

3	2	1	0
- optional activation of the triggering of the application event processing for instructions : 10, 11 05 and 07 (bit 12 at 1 for activation)
  - setting the **auxiliary discrete output** associated with the channel to 0 or 1 for instructions : 01, 09, 10, and 11
- Four-bit byte no. 2 :
- 0 = **Unchanged** : no modification of the output
  - 1 = **synchronous with the mvt** : assignment of the output at the start of execution of the instruction
  - 2 = **following the mvt** : assignment of the output at the end of execution of the instruction
- Four-bit byte no. 0 :
- 0 = forcing of the output to 0 (AUX 0 box not checked)
  - 1 = forcing of the output to 1 (AUX 0 box checked)
- type of event awaited by instruction G05
- Bit no. 13 :
- 0 = awaiting a time-out or an event
  - 1 = awaiting a modulo crossing number

## 10.2 General module data

**%Ixy.MOD.ERR      Module fault**

**%MWxy.MOD.2:Xj      Module standard status word**

bit 0	internal fault (module off)
bit 1	operating fault (see channel status word)
bit 3	module performing self-tests
bit 5	hardware or software configuration fault
bit 6	module absent

## 10.3 Internal command data (implicit exchanges)

**%Qxy.i.j**

**Processor --> CAY**

bit 0	<b>DIRDRV</b>	State movement command in direct drive mode
bit 1	<b>JOG_P</b>	State unlimited manual movement in positive direction
bit 2	<b>JOG_M</b>	State unlimited manual movement in negative direction
bit 3	<b>INC_P</b>	Edge incremental movement (PARAM) in positive direction command
bit 4	<b>INC_M</b>	Edge incremental movement (PARAM) in negative direction command
bit 5	<b>SET_RP</b>	Edge set manual reference point (RP_POS=reference value) or change to non-referenced state
bit 6	<b>RP_HERE</b>	Edge force reference point to a value defined in PARAM or change to referenced/offset calculation state
bit 8	<b>ACK_DEF</b>	Edge acknowledge faults
bit 9	<b>ENABLE</b>	State enable axis speed drive safety relay
bit 10	<b>EXT_EVT</b>	Edge command to trigger an event from the processor
bit 11	<b>AUX_OUT</b>	State auxiliary output command
bit 15	<b>STOP</b>	State immediate stop command (stop moving part)
bit 16	<b>PAUSE</b>	State command to suspend movement at the end of the current movement
bit 17	<b>SLAVE</b>	State current setpoint = position of axis 0
bit 18	<b>EXT_CMD</b>	State current setpoint = processor setpoint
bit 19	<b>MOD_STEP</b>	State command to switch to step by step mode
bit 22	<b>NEXT_STEP</b>	Edge command to launch next step

**%QWxy.i.0      MODE\_SEL      Mode selector**

Value

0	<b>DRV_OFF</b>	measurement mode : inhibition of D/A converter output
1	<b>DIRDRIVE</b>	direct drive mode : direct voltage control
2	<b>MANU</b>	manual mode
3	<b>AUTO</b>	automatic mode

**%QWxy.i.1      CMV      Speed correction**

Value : value of speed correction setpoint from 0 to 2 in steps of 1/1000

**%QDxy.i.2      PARAM      Value of the movement increment**

## 10.4 Internal status data (implicit exchanges)

<b>%lx<i>y.i.j</i></b>	<b>processor &lt;-- CAY</b>
bit 0	<b>NEXT</b> ready to receive a new movement command (in AUTO)
bit 1	<b>DONE</b> all the instructions have been executed : no more instructions in the stack
bit 2	<b>AX_FLT</b> error on the axis
bit 3	<b>AX_OK</b> no fault causing the moving part to stop
bit 4	<b>HD_ERR</b> presence of a hardware fault
bit 5	<b>AX_ERR</b> presence of an application fault
bit 6	<b>CMD_NOK</b> command failure
bit 8	<b>NOMOTION</b> moving part stationary
bit 9	<b>AT_PNT</b> moving part positioned on target (in the target window, on an instruction with stop)
bit 10	<b>TH_PNT</b> theoretical setpoint reached
bit 12	<b>CONF_OK</b> configured axis
bit 14	<b>REF_OK</b> reference point set (axis referenced)
bit 15	<b>AX_EVT</b> copies the physical event inputs
bit 16	<b>HOME</b> copies the reference point CAM physical input on the module
bit 17	<b>DIRECT</b> indicates the direction of movement
bit 18	<b>IN_REC</b> copies the recalibration on the fly input
bit 20	<b>IN_DROFF</b> measurement mode active
bit 21	<b>IN_DIRDR</b> direct drive mode active
bit 22	<b>IN_MANU</b> manual mode active
bit 23	<b>IN_AUTO</b> automatic mode active
bit 26	<b>ST_JOG_P</b> unlimited movement in + direction in progress
bit 27	<b>ST_JOG_M</b> unlimited movement in - direction in progress
bit 28	<b>ST_INC_P</b> incremental movement in + direction in progress
bit 29	<b>ST_INC_M</b> incremental movement in - direction in progress
bit 30	<b>ST_SETRP</b> current manual reference point
bit 31	<b>ST_DIRDR</b> direct drive movement in progress
bit 32	<b>IN_INTERPO</b> interpolated movement in progress
bit 33	<b>ON_PAUSE</b> movement sequencing suspended
bit 34	<b>IM_PAUSE</b> movement suspended (immediate PAUSE)
bit 36	<b>IN_SLAVE</b> current setpoint = axis 0 position
bit 37	<b>IN_EXT_CMD</b> current setpoint = processor setpoint
bit 39	<b>ST_IN_STEP</b> step by step mode in progress
bit 40	<b>DRV_ENA</b> Image of speed drive enable output
bit 41	<b>IN_AUX0</b> Image of AUX 0 output
bit 46	<b>OVR_EVT</b> event overrun
bit 47	<b>EVT_G07</b> event source : position memorization
bit 48	<b>EVT_G05</b> event source : end of G05 on detection of an event
bit 49	<b>TO_G05</b> event source : G05 timeout elapsed
bit 50	<b>EVT_G1</b> event source : end of G10 or G11 on detection of an event
bit 51	<b>EVT_MOD</b> modulo crossing
bitERR	<b>ERROR</b> channel fault

If channels 0, 1 and 2 are interpolated, the **IN\_INTERPO** bits are set to 1(%lx*y.0.32*, *lx.y.1.32* and *lx.y.2.32*).

<b>%IDxy.i.0</b>	<b>X_POS</b>	<b>measured position</b>
<b>%IDxy.i.2</b>	<b>SPEED</b>	<b>measured speed</b>
<b>%IDxy.i.4</b>	<b>FOL_ERR</b>	<b>current position error</b>
<b>%Wxy.i.6</b>	<b>ANA_OUT</b>	<b>current analog output</b>
<b>%Wxy.i.7</b>	<b>SYNC_N_RUN</b>	<b>current step number</b>
<b>%IDxy.i.9</b>	<b>PREF1</b>	<b>Value of register PREF1</b>
<b>%IDxy.i.11</b>	<b>PREF2</b>	<b>Value of register PREF2</b>

## 10.5 Internal status data (explicit exchanges)

### %MWxy.i.0:Xj exchange management

bit X0	<b>STATUS</b>	exchange of status parameters in progress (STATUS)
bit X1	<b>COMMAND</b>	exchange of command parameters in progress.
bit X2	<b>ADJUST</b>	exchange of adjustment parameters in progress.
bit X15	<b>CONFIG</b>	reconfiguration of module in progress

### %MWxy.i.1:Xj exchange report

bit X1	<b>CR_RPT</b>	report on exchange of command parameters.
bit X2	<b>ADJ_RPT</b>	report on exchange of adjustment parameters.
bit X15	<b>CONF_FLT</b>	configuration fault

### %MWxy.i.2:Xj Channel operating status

bit X0	<b>EXT_FLT</b>	external fault (same as bit HD_ERR)
bit X4	<b>MOD_FLT</b>	internal fault : module absent, off or performing self-tests
bit X5	<b>CONF_FLT</b>	hardware or software configuration fault
bit X6	<b>COM_FLT</b>	communication fault with CPU
bit X7	<b>APP_FLT</b>	application fault : incorrect configuration, or command
bit X8	<b>CH_LED_LOW</b>	Status of channel indicator lamps
bit X9	<b>CH_LED_HIGH</b>	Status of channel indicator lamps

### %MWxy.i.3:Xj Axis operating status

Hardware faults : %lxy.i.4 **HD\_ERR** (includes all the faults below)

bit X0	<b>ANA_FLT</b>	analog output short-circuit fault
bit X1	<b>AUX_FLT</b>	auxiliary output short-circuit fault
bit X2	<b>DRV_FLT</b>	speed drive fault
bit X3	<b>ENC_SUP</b>	encoder supply fault
bit X4	<b>ENC_BRK</b>	encoder wiring break fault
bit X5	<b>EMG_STP</b>	emergency stop fault
bit X6	<b>AUX_SUP</b>	24 V supply fault
bit X7	<b>ENC_FLT</b>	serial absolute encoder or bit E parity error

Application faults : %lxy.i.5 **AX\_ERR** (includes all the faults below)

bit X8	<b>SLMAX</b>	maximum soft stop overshoot
bit X9	<b>SLMIN</b>	minimum soft stop overshoot
bit X10	<b>SPD_FLT</b>	overspeed fault
bit X11	<b>FE1_FLT</b>	position error fault <b>MAX_F1</b>
bit X12	<b>REC_FLT</b>	recalibration fault
bit X13	<b>TW_FLT</b>	target window fault
bit X14	<b>STP_FLT</b>	stop fault
bit X15	<b>FE2_FLT</b>	position error fault <b>MAX_F2</b>

%MWxy.i.4	<b>N_RUN</b>	current step number
%MWxy.i.5	<b>G9_COD</b>	current type of movement
%MWxy.i.6	<b>G_COD</b>	current instruction
%MWxy.i.7	<b>CMD_FLT</b>	command failure report
%MDxy.i.9	<b>T_XPOS</b>	position target to be reached
%MDxy.i.11	<b>MAX_FER</b>	maximum position error
%MDxy.i.13	<b>T_SPEED</b>	target speed

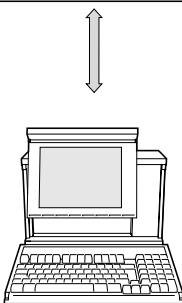
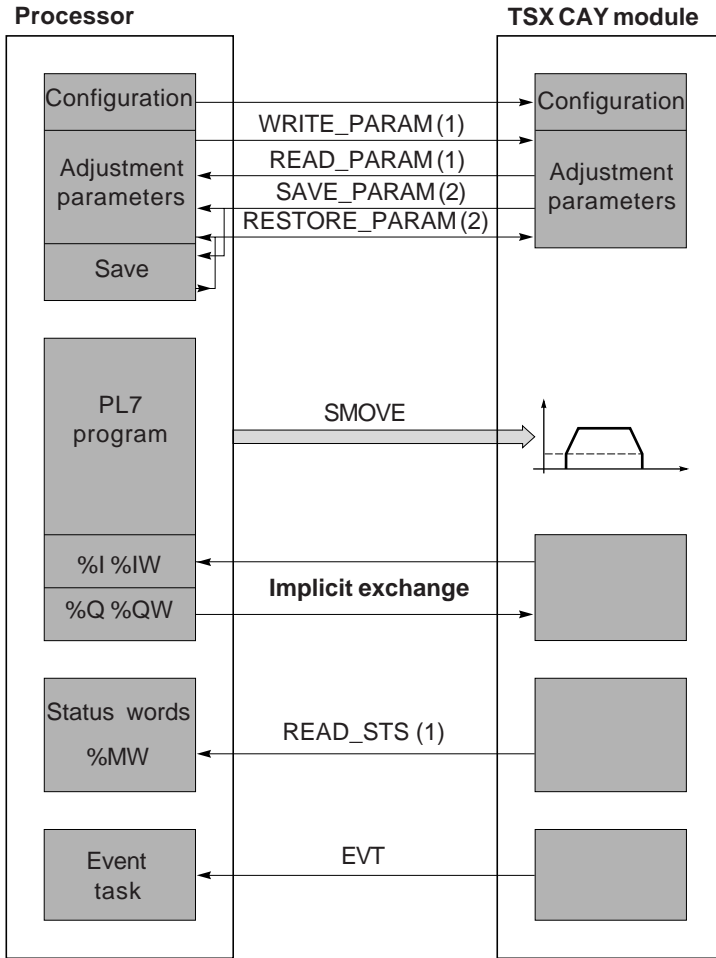
## 10.6 Adjustment parameters (explicit exchanges)

%MWxy.i.j or %MDxy.i.j

%MWxy.i.15	<b>SLOPE</b>	acceleration profile 0 = rectangle, 1 to 3 = trapezoid, 4 = triangle
%MWxy.i.16	<b>KPOS1</b>	gain 1 of position loop 0 to 120.00 (in 1/s)
%MWxy.i.17	<b>KPOS2</b>	gain 2 of position loop 0 to 120.00 (in 1/s)
%MWxy.i.18	<b>SP_THR</b>	change of gain threshold : 20 to 500Vmax/1000
%MWxy.i.19	<b>IPOS</b>	integral action Ti : integral time 0 to 5000 ms - 0 no integral action (TSX CAY •2/33)
%MWxy.i.19		reserved
%MWxy.i.20		reserved
%MWxy.i.21	<b>KV</b>	loop feedforward gain : 0 to 100 %
%MWxy.i.22	<b>OFFSET</b>	loop D/A converter offset : -150 to 150 mV
%MWxy.i.23	<b>OVR_SPD</b>	overspeed threshold : 0 to 20 %
%MWxy.i.24	<b>S_STOP</b>	stop speed : 0 to VMAX/10 or 30000
%MWxy.i.25	<b>T_STOP</b>	maximum time delay for detection of stop : 0 to 10000 ms
%MWxy.i.26	<b>TACC</b>	acceleration/deceleration time : TACCMIN to 10000 (in ms)
%MWxy.i.27	<b>VLIM</b>	motion control activation threshold
%MWxy.i.29	<b>RATIO1</b>	slave axis ratio (TSX CAY •2)
%MWxy.i.30	<b>RATIO2</b>	slave axis ratio (TSX CAY •2)
%MDxy.i.31	<b>SL_MAX</b>	upper soft stop limit : SLMIN to LMAX for limited axis - modulo in points for infinite axis
%MDxy.i.33	<b>SL_MIN</b>	lower soft stop limit : LMIN to SLMAX for limited axis - modulo value in user units for infinite axis
%MDxy.i.35	<b>MAN_SPD</b>	speed in manual mode 10 to VMAX
%MDxy.i.37	<b>K_RES1</b>	resolution multiplier 1 to 1000 000
%MDxy.i.39	<b>K_RES2</b>	resolution divisor 1 to 1000 000
%MDxy.i.41	<b>RP_POS</b>	reference point value in manual mode SLMIN to SLMAX
%MDxy.i.43	<b>RE_POS</b>	recalibration reference value : SLMIN to SLMAX
%MDxy.i.45	<b>MAX_F1</b>	position error threshold 1 : 0 to (SLMIN-SLMAX)/4
%MDxy.i.47	<b>MAX_F2</b>	position error threshold 2 : 0 to (SLMIN-SLMAX)/4
%MDxy.i.49	<b>TW</b>	target window : calculation from 0 to (SLMIN-SLMAX)/20
%MDxy.i.51	<b>RE_WDW</b>	recalibration deviation threshold : 0 to (SLMIN-SLMAX)/20
%MDxy.i.53	<b>ABS_OFF</b>	absolute encoder offset
%MDxy.i.55	<b>SLAVE_OFF</b>	axis follower offset (TSX CAY •2)
%MDxy.i.62:X0	<b>VALIDEVTMOD</b>	event confirmation on modulo crossing (TSX CAY•2 33)

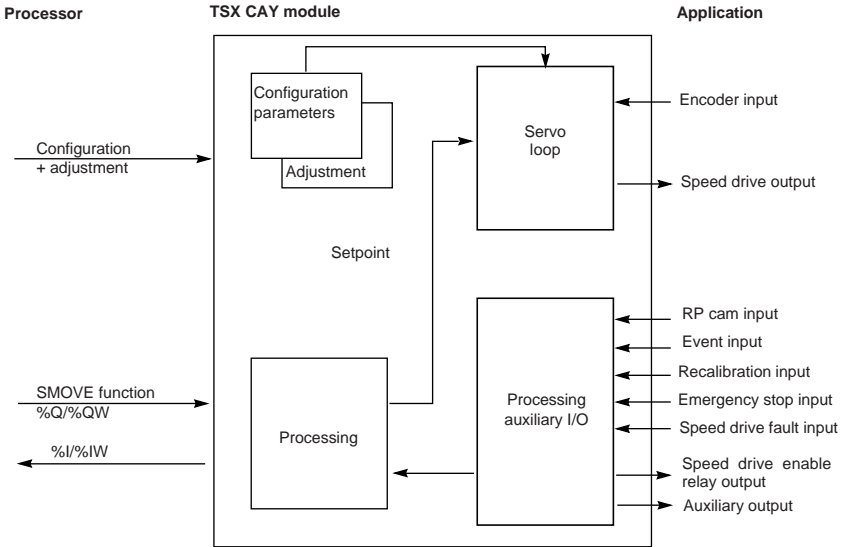


10.7 Block diagram of data exchanges



(1) read or write from the adjustment screen or from the application program using explicit exchange instructions  
 (2) save or restore using the Save Parameters or Restore Parameters commands in the PL7 Utilities menu or using the SAVE\_PARAM or RESTORE\_PARAM instructions.

## 10.8 Block diagram of the TSX CAY module



## 11.1 List of CMD\_FLT error codes

A list of messages explaining the CMD\_FLT (%MWxy.i.7) command failure word is given on the following pages.  
This word is read by explicit exchange.

Messages also appear in clear text in the Diagnostics dialog boxes which can be accessed using the **DIAG** key (see section 6.3).

The CMD\_FLT word is broken down into two bytes. Each byte corresponds to a specific class of error.

%MWxy.i.7	Configuration and adjustment parameter	Movement command
	High order byte	Low order byte

High order byte : error in the configuration and adjustment parameters. (XX00)

Low order byte : failure to execute the movement command. (00XX):

Example : **0004**

└── Low order byte : JOG+ command error

---

The values in brackets are the hexadecimal code values.

**Configuration** (in the high order byte of word %MWxy.i.7)

- 3 (3) = Event priority configuration error
- 4 (4) = Machine configuration error (infinite, limited)
- 5 (5) = Encoder type configuration error
- 6 (6) = Reference point configuration error
- 7 (7) = Maximum setpoint configuration error
- 8 (8) = Maximum acceleration configuration error
- 9 (9) = Event configuration error
- 10 (A) = Follower ratio multiplier configuration error
- 11 (B) = Follower ratio divisor configuration error
- 12 (C) = Recalibration configuration error
- 18 (12) = Speed configuration error
- 19 (13) = Upper limit configuration error
- 20 (14) = Lower limit configuration error
- 21 (15) = Initial resolution configuration error (distance)
- 22 (16) = Initial resolution configuration error (number of points)
- 25 (19) = Unit of length configuration error
- 26 (1A) = Unit of speed configuration error
- 27 (1B) = Resolution and speed ratio configuration error
- 28 (1C) = Incompatible limits configuration error
- 29 (1D) = Follower ratio configuration error

**Adjustment parameter** (in the low order byte of word %MWxy.i.7)

- 82 (52) = Acceleration profile parameter error
- 83 (53) = Gain 1" parameter error
- 84 (54) = Gain 2" parameter error
- 85 (55) = Threshold 1 to 2" parameter error
- 88 (58) = Feedforward parameter error
- 89 (59) = Offset parameter error
- 90 (5A) = Overspeed parameter error
- 91 (5B) = Stop control speed parameter error
- 92 (5C) = Stop control delay parameter error
- 93 (5D) = Acceleration parameter error
- 94 (5E) = VLIM parameter error
- 98 (62) = Upper soft stop parameter error
- 99 (63) = Lower soft stop parameter error
- 100 (64) = Manual mode speed parameter error
- 101 (65) = Corrected resolution parameter error (distance)
- 102 (66) = Corrected resolution parameter error (number of points)
- 103 (67) = Reference point value parameter error
- 104 (68) = Recalibration position value parameter error
- 105 (69) = Deviation 1" parameter error
- 106 (6A) = Deviation 2" parameter error
- 107 (6B) = Target window parameter error
- 108 (6C) = Recalibration deviation parameter error
- 109 (6D) = Encoder offset parameter error
- 113 (71) = Resolution ratio parameter error
- 114 (72) = Incompatible soft limits parameter error
- 115 (73) = Max. speed and resolution ratio parameter error
- 116 (74) = Encoder mult., VMax and resolution ratio parameter error
- 117 (75) = Resolution ratio at upper limit parameter error
- 118 (76) = Resolution ratio at lower limit parameter error
- 119 (77) = Resolution ratio at limit distance parameter error
- 120 (78) = Resolution correction parameter error (Mode <> OFF)
- 121 (79) = Encoder offset modification parameter error (Mode <> OFF)
- 122 (7A) = Recalibration position modification parameter error (Mode <> OFF)

## Movement command failure (in the low order byte of word %MWxy.i.7)

- 1 (1) = Insufficient conditions manual command error (Mode, Value, etc)
- 2 (2) = Current manual movement manual command error
- 3 (3) = Simultaneous commands manual command error
- 4 (4) = JogP manual command error
- 5 (5) = JogM manual command error
- 6 (6) = IncP manual command error
- 7 (7) = IncM manual command error
- 8 (8) = Manual reference point manual command error
- 9 (9) = Forced reference point manual command error
- 10 (A) = Encoder offset calculation error
- 16 (10) = Insufficient conditions Auto command error (parameters)
- 17 (11) = Current auto movement Auto command error (Slave and External control commands activated simultaneously with a movement)
- 18 (12) = Insufficient conditions movement command error (Mode)
- 19 (13) = G01" movement command error (1)
- 20 (14) = G09" movement command error (1)
- 21 (15) = G10" movement command error (1)
- 22 (16) = G11" movement command error (1)
- 25 (19) = G14" movement command error (1)
- 26 (1A) = G05" movement command error (1)
- 27 (1B) = G07" movement command error (1)
- 28 (1C) = G62" movement command error (1)
- 29 (1D) = Movement execution command error
- 30 (1E) = Slave Auto command error
- 31 (1F) = External control Auto command error
- 32 (20) = Current slave mode Auto command error
- 33 (21) = External control current Auto command error
- 34 (22) = Current external movement command error on slave axis
- xx (xx) = No zero marker on the cam error for a short cam type reference point with zero marker
- 35 (23) = Stack full error
- 36 (24) = Sequence check error
- 37 (25) = SMOVE G30 (1) command error
- 38 (26) = Change to next step error
- 48 (30) = DIRDRIVE command error, command incomplete
- 80 (50) = SMOVE G30 command error : already at the position
- 81(51) = SMOVE G30 command error : change of direction

(1) indicates that one of the parameters of the SMOVE function is incorrect. Examples : incorrect movement type code, position outside soft stop limits, speed above VMAX, etc.

<b>Section</b>	<b>Page</b>
<b>1 Introduction to interpolation</b>	<b>1/1</b>
1.1 Introduction to interpolation	1/1
1.2 Introductory example	1/5
1.2-1 Description of the position control example	1/5
1.2-2 Software declaration of the PLC configuration used	1/5
1.2-3 Entering the configuration parameters of each axis	1/5
1.2-4 Configuring the interpolator	1/6
1.2-5 Entering the symbols	1/7
1.2-6 Programming	1/7
1.2-7 Transferring the program	1/12
1.2-8 Adjustment and debugging	1/13
1.2-9 Archiving	1/16
1.3 Methodology for setting up interpolation	1/17
<b>2 Configuring interpolation</b>	<b>2/1</b>
2.1 Accessing parameters in the configuration screen	2/1
2.2 Entering parameters	2/1
2.3 Confirming the configuration parameters	2/2

Section	Page
<b>3 Programming</b>	<b>3/1</b>
3.1 Programming interpolated movements : XMOVE function	3/1
3.1-1 Programming an XMOVE function	3/1
3.1-2 Entering parameters for the XMOVE function	3/2
3.1-3 Description of elementary movements	3/5
3.1-4 Description of the instructions	3/6
3.1-5 Sequencing movement commands	3/9
3.1-6 Using the XMOVE function and the SMOVE function together	3/13
3.1-7 Interpolator channel automatic mode	3/14
3.1-8 Event processing	3/15
3.2 Fault management	3/16
3.2-1 Role	3/16
3.2-2 Principle	3/17
3.2-3 Programming	3/18
3.2-4 Description of command failure faults	3/19
3.3 Managing OFF mode	3/20
<b>4 Adjusting the axes</b>	<b>4/1</b>
4.1 Accessing the adjustment parameters	4/1
4.2 Acceleration profile	4/2
4.3 Points through which the axes pass	4/3



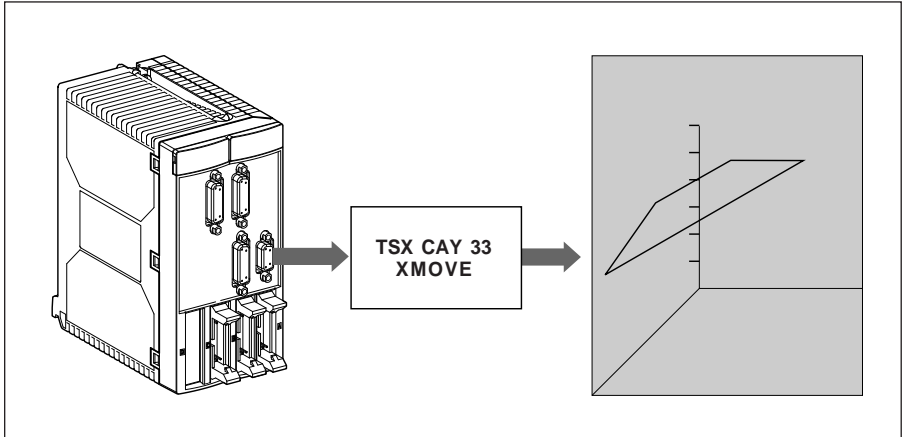
<b>Section</b>	<b>Page</b>
<b>5 Debugging a program with interpolation</b>	<b>5/1</b>
5.1 Principle of debugging a program with interpolation	5/1
5.2 Debug screen	5/2
5.2-1 Accessing the debug screen	5/2
5.2-2 User interface	5/2
5.2-3 Description of the debug screens	5/3
5.2-4 Debugging the interpolation of channels X, Y and Z	5/4
5.2-5 OfF mode	5/5
5.2-6 Automatic mode (Auto)	5/7
5.3 Diagnostics	5/9
<b>6 Quick reference guide</b>	<b>6/1</b>
6.1 Programming the XMOVE interpolation function	6/1
6.2 Interpolator data	6/2
6.3 Interpolation adjustment parameters	6/3
6.4 Internal status data	6/4
<b>7 List of CMD_FLT error codes</b>	<b>7/1</b>
7.1 List of CMD_FLT error codes	7/1

**Section****Page**

## 1.1 Introduction to interpolation

Part B2 describes the setting up of an application requiring **linear** interpolation between 2 or 3 axes.

The interpolation function is only available on the **TSX CAY 33** module.



The TSX CAY 33 module has 3 physical channels (0, 1 and 2) to be associated with axes X, Y and Z, and a logical channel (channel no. 3) dedicated to interpolation. Setting up an application with interpolated axes requires the prior configuration of each axis as an independent axis (see part B1).

Interpolation can either be performed between 2 axes (0 and 1) in the (X, Y) plane, or between 3 axes (0, 1 and 2) in space.

During 2-axis interpolation, the 3rd axis (no. 2) can be used as an independent axis.

### Restriction :

The TSX CAY 33 module does not provide circular interpolation. However, to go from point A to point B following a circular trajectory, it is possible to approximate this type of trajectory using a series of straight segments.

### Configuring interpolated axes

The number of interpolated axes is defined in the configuration of the interpolator (channel 3) when channels 0 to 2 have been configured as independent axes.

By specifying **2-dimensional** the user implicitly declares that the interpolation is to be performed in a space limited to the plane (X, Y), where X : axis 0 and Y : axis 1.

In this case, channel 2 of the module remains available and can be used as an independent axis.

By specifying **3-dimensional** the user implicitly declares that the interpolation is to be performed in a three-dimensional space (XYZ) and/or in the planes which make up this space (XY, YZ or XZ), where X : axis 0, Y : axis 1, Z : axis 2.

The configuration and adjustment parameters of the 2 or 3 axes which the user wishes to interpolate are defined independently and individually for each of the axes in the group via the position control function. This configuration of axes X, Y (and Z) is necessary in order to access the configuration of the interpolator.

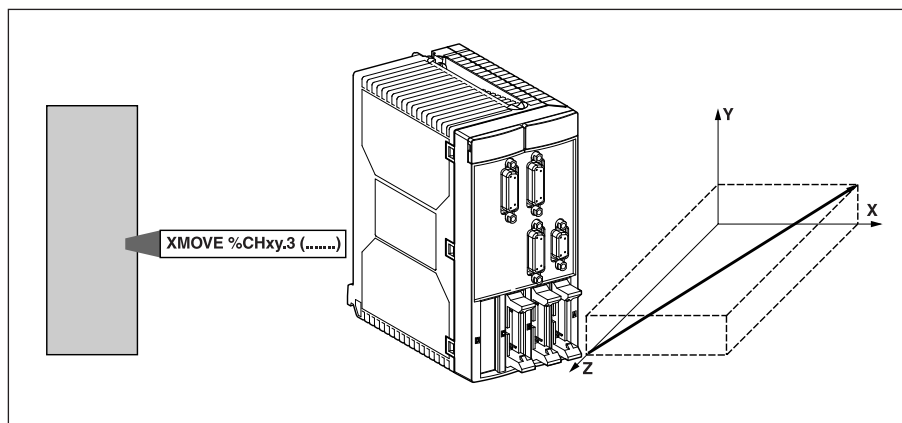
### The interpolator channel

The interpolator channel has an XMOVE command specifically for interpolated movements.

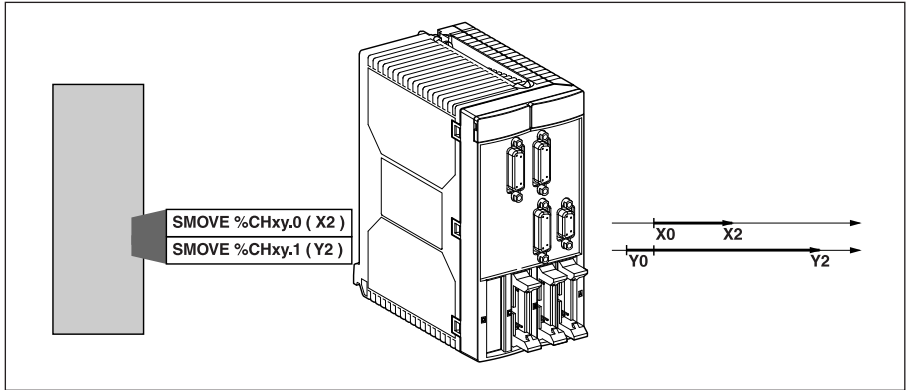
This interpolator command does not replace the axis commands. It is an additional command.

Axes are only interpolated during the execution of an XMOVE command. Apart from XMOVE commands, they can be controlled independently by SMOVE commands.

### Interpolated axis movement control



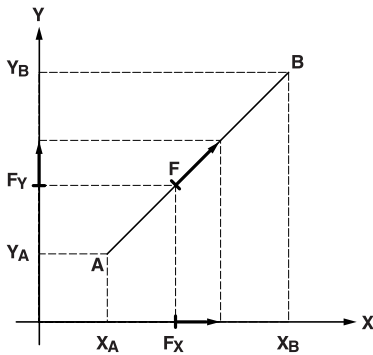
## Independent axis movement control



### Speed

The speed specified in the XMOVE command is the required speed in the direction of the movement. The speed of movement of each of the axes is calculated by projection.

### Example for a 2-axis system :



The moving part must move from point A ( $X_A, Y_A$ ) to point B ( $X_B, Y_B$ ) at a speed  $F$ , which is projected on X and Y as  $F_x$  and  $F_y$  respectively.

Based on the value  $F$  provided in the XMOVE instruction, the interpolator calculates the projections according to the following formulae :

$$F_x = F * |X_B - X_A| / \Delta X$$

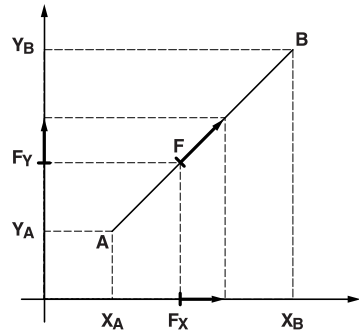
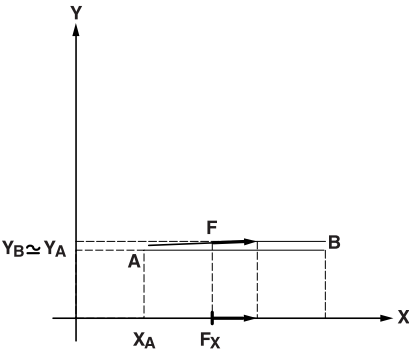
$$F_y = F * |Y_B - Y_A| / \Delta X$$

where  $\Delta X = [(X_B - X_A)^2 + (Y_B - Y_A)^2]^{1/2}$

Speed F is limited to a maximum value, depending on the one hand on the maximum speeds of each of the axes involved in the movement, and on the other on the contribution of each of the axes to the movement.

Example of a movement where the contribution of axis Y is negligible

Example of a movement where the contribution of both axes is equal



$$F_x \approx 0$$

$$F \leq V_{MAX\_axisX}$$

$$|X_b - X_a| = |Y_b - Y_a|$$

$$F_y = F_x$$

$$F \leq (F_x^2 + F_y^2)^{1/2}$$

**Acceleration**

For each XMOVE movement, the duration of the acceleration phase depends on :

- the speed variation to be obtained
- the Tacc parameters of the axes involved in the movement
- the contribution of axes X, Y and Z

The resulting calculated acceleration is the highest acceleration with which the movement can be performed while observing the restrictions of the various axes (it is the most restrictive axis which imposes the acceleration period).

The acceleration profile is defined by the **SLOPE** parameter of channel 3. This imposes a common profile for all the axes during an XMOVE, independently of the value of the **SLOPE** parameter of axes X, Y, Z.

---

## 1.2 Introductory example

---

### 1.2-1 Description of the position control example

The same application example as that described in part B1 section 1.1 is used, but the movement of the clamp in the plane (X, Y) will be controlled by a system of 2 interpolated axes.

---

### 1.2-2 Software declaration of the PLC configuration used

Perform the operations described in part B1 section 1.3-1.

---

### 1.2-3 Entering the configuration parameters of each axis

Each of the axes used for the interpolation must have already been configured. To do this, perform the operations described in part B1, section 1.3-2.



### 1.2-4 Configuring the interpolator

Channel 3, for which there is no corresponding physical axis, is used to perform the interpolation between axes 0 and 1.

Select the INTERPOLATION function and the MAST task.

Enter the values of the parameters in the screen below.

Parameter	Description	Value
Dimension	Number of interpolated axes	2
Stop function	Effect of the STOP command	XMOVE
Stop on fault	Consequence of a fault	INTERPOLATED

Confirm the entry with the  icon then, in the main screen of the configuration editor, confirm the configuration with the  icon.



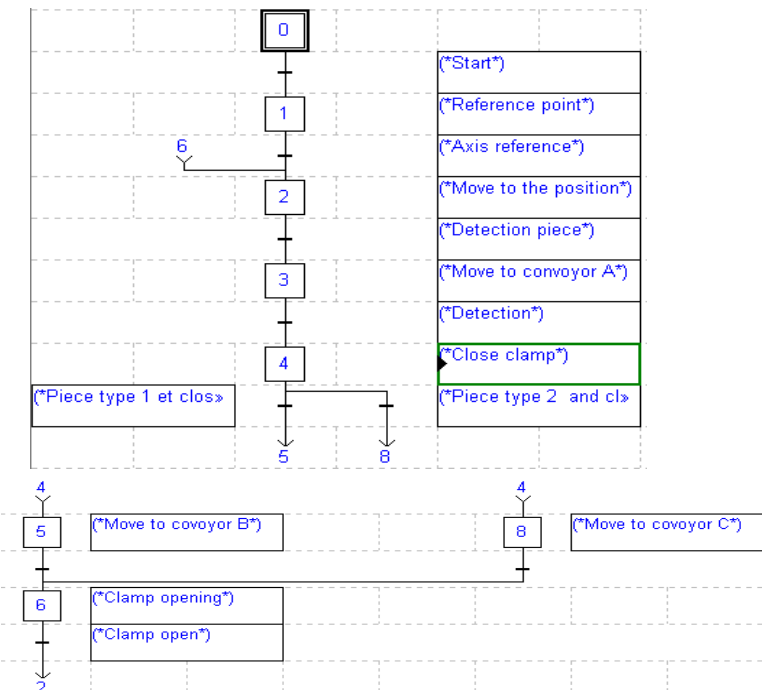
### 1.2-5 Entering the symbols

Symbol	Object	Value	Role
Cycle	%M0		Condition of the machine in work mode
Speed_r_p_x	%KD0	1000	Reference point speed following the X axis
Speed_wait	%KD4	1200	Speed towards wait position
Speed_pos_a	%KD8	1500	Speed towards conveyor position A
Speed_pos_b	%KD12	1200	Speed towards conveyor position B
Speed_pos_c	%KD16	1800	Speed towards conveyor position C

### Symbols linked to interpolation

Symbol	Object	Symbol	Object
INTERPO	CH103.3	ERROR_INT	%I103.3.2
NEXT_INT	%I103.3	OK_INT	%I103.3.3
DONE_INT	%I103.3.1	AT_POINT_INT	%I103.3.9

### 1.2-6 Programming



For simultaneous movements of axes X and Y (steps 2, 3, 5, 8 and the associated transitions) the XMOVE movement command (associated with channel 3) is used instead of the single SMOVE movement commands addressed to each of the axes (X and Y).

**Step 0 :****Transition X0 > X1**

!(\*Channel X not faulty, clamp open, Auto\_man switch on Auto, start cycle, channel Y not faulty and automatic mode active\*)

**NOT Error AND NOT Sensor\_3 AND NOT Auto\_man AND Cycle AND NOT Error\_y AND Mode\_auto**

**Step 1 : Action on activation**

!(\*Reference point following the X\* axis)

**SMOVEAxis\_x(1,90,14,0,Speed\_r\_p\_x,16#0000);**

**Transition X1 > X2**

!(\*Test : axis X ready and referenced\*)

**Done AND Calib**

**Step 2 : Action on activation**

!(\*Move to wait position (Xwait, Ywait\*)

**XMOVE INTERPO (2,90,9,0,x\_wait,y\_wait,0,Speed\_wait,16#0000);**

**Transition X2 > X3**

!(\*Moving part in wait position and part detected on conveyor A\*)

**Sensor\_1 AND Cycle AND Next\_INT**

**Step 3 : Action on activation**

!(\*Move to conveyor A\*)

**XMOVE INTERPO (3,90,10,0,150000,280000,0,Speed\_pos\_a,16#0000);**

**Transition X3 > X4**

!(\*Moving part in position to pick up part detected on conveyor A\*)

**ATPOINT\_INT AND Next\_INT**

**Step 4 : Continuous action**

!(\*Close clamp\*)

**SET Clamp;**

**Transition X4 > X5**

!(\*Type 1 part and clamp closed\*)

**Sensor\_2 AND Sensor\_3**

**Step 5 : Action on activation**

!(\*Move to conveyor B\*)

**XMOVE INTERPO (4,90,9,0,X\_b,Y\_b,0,Speed\_pos\_a,16#0000);****Transition X4 > X8**

!(\*Type 2 part and clamp closed\*)

**Not Sensor\_2 AND Sensor\_3****Step 8 : Action on activation**

!(\*Move to conveyor C\*)

**XMOVE INTERPO (5,90,9,0,X\_c,Y\_c,0,Speed\_pos\_c,16#0000);****Transition X5 > X6**

!(\*Moving part in position on conveyor B\*)

**AT\_POINT\_INT AND Next\_INT****Transition X8 > X6**

!(\*Moving part in position on conveyor C\*)

**AT\_POINT\_INT AND Next\_INT****Step 6 : Continuous action**

!(\*Open clamp\*)

**RESET Clamp;****Transition X6 > X2**

!(\*Clamp open\*)

**NOT Sensor\_3 AND Cycle**

### Pre-processing

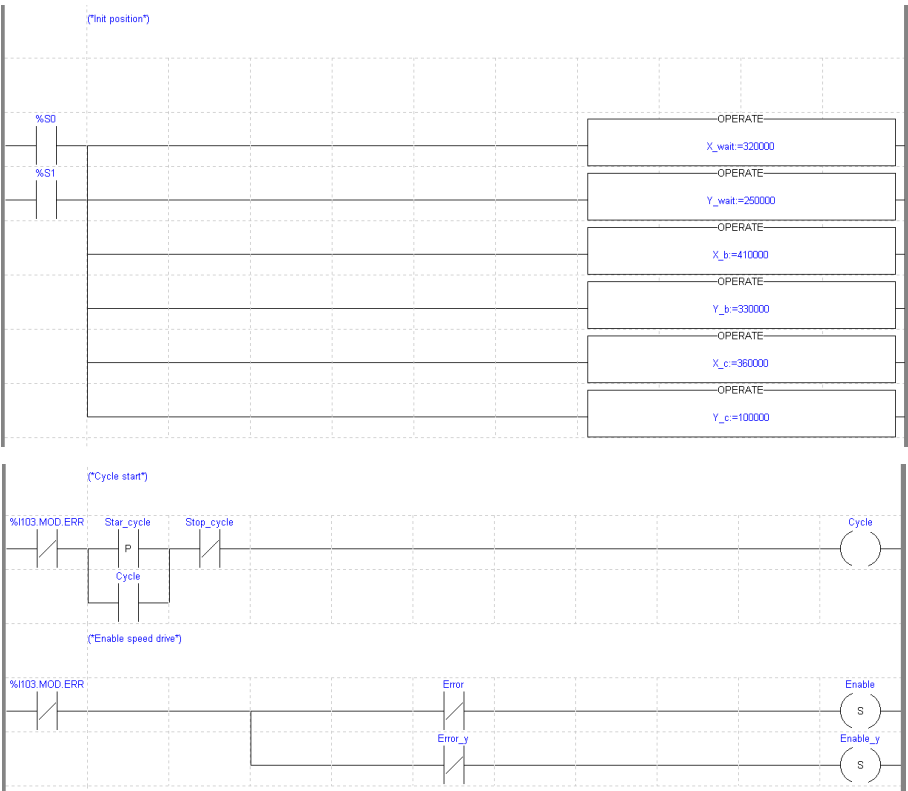
Pre-processing comprises the management of the operating modes.

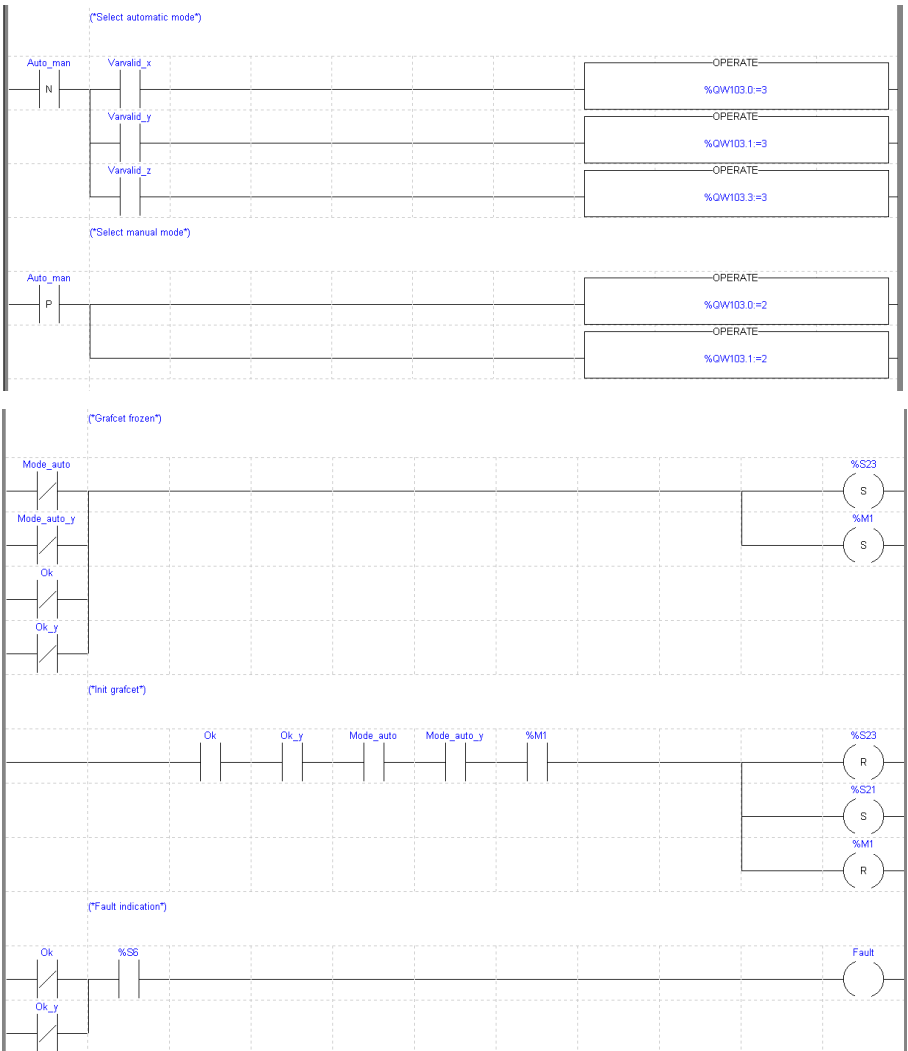
On a blocking fault :

- Freezing of the chart.
- The operator can then control his moving part in manual mode, and correct and acknowledge the fault from the front panel.
- Reinitialization of the chart when the fault has disappeared and been acknowledged.

On changing to manual mode :

- Freezing of the chart.
- Reinitialization of the chart when AUTOMATIC mode is selected again.





## Post-processing

Post-processing comprises management of the manual mode.

```
! (*Testing the selected mode*)
  IF Mode_auto AND Mode_auto_y AND Config AND Config_y
  THEN JUMP %L200;
  END_IF;
! (*Selecting the axis to drive*)
  %L100: IF NOT Selection_x_y
  THEN JUMP %L200;
  END_IF;
! (*Manual setpoint command of X axis*)
  IF RE Po_man
  THEN Posrp:=0; SET setrp; Fmanu_x:=1000; WRITE_PARAM Axis_x;
  END_IF;
  IF NOT Po_man
  THEN RESET Setrp;
  END_IF;
! (*Moving part in + direction of X axis*)
  Jog_p:=front;
! (*Moving part in - direction of X axis*)
  Jog_m:=rear;
! %L200: IF Selection_x_y
  THEN JUMP %L300;
  END_IF;
! (* Moving part in + direction of Y axis*)
  Jop_p_y:=front;
! (* Moving part in - direction of Y axis*)
  Jop_m_y:=rear;
! (*Opening the clamp*)
  %L300: IF Auto_man AND op_clamp
  THEN RESET Clamp;
  END_IF;
  (*Closing the clamp*)
  IF Auto_man AND cl_clamp
  THEN SET Clamp;
  END_IF;
! (*Defaults Acknowledgement*)
  Ack_def:=Ack_def_y:=Ack_defaults;
! %L999;
```

### 1.2-7 Transferring the program


Once the program has been entered, this operation consists of transferring the configuration and the program to the PLC processor memory :

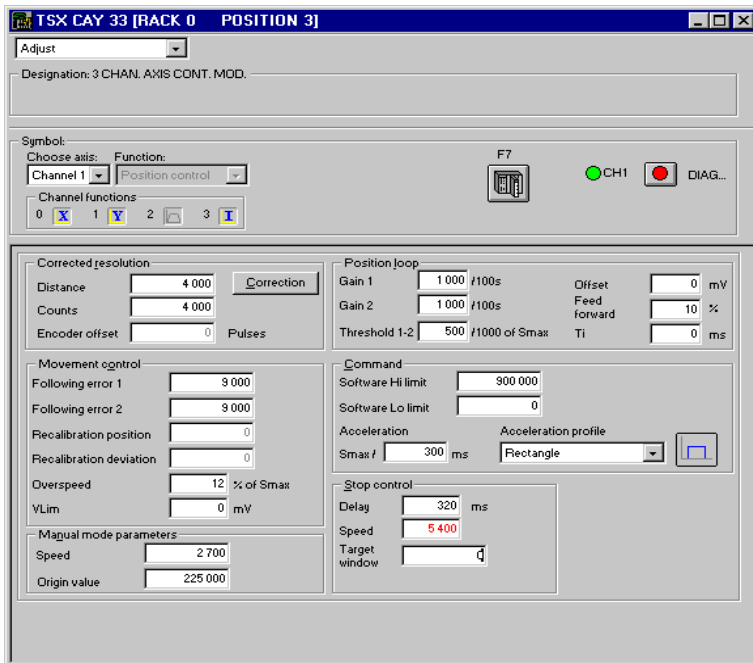
- Connect the terminal to the PLC using the **PLC/Connect** command.
- Launch the **PLC/Transfer** command, choose the "Terminal -> PLC" option then confirm.

## 1.2-8 Adjustment and debugging

### Adjusting the parameters

Firstly, for safety, perform the preliminary operations described in part B1, section 5.1. Then perform the following operations :


- Set the PLC to RUN.
- Select the **Application/Configuration** command or click on the  icon.
- Select position no. 3 in rack 1 and execute the **Edit/Open module** command (or double click on the selected module).
- Select the "**View/Adjust**" command.



The screenshot shows the 'Adjust' dialog box for 'TSX CAY 33 [RACK 0 POSITION 3]'. The 'Adjust' dropdown is set to 'Adjust'. The designation is '3 CHAN. AXIS CONT. MOD.'. The symbol section shows 'Channel 1' selected for 'Position control' with 'F7' and 'DIAG...' indicators. The 'Corrected resolution' section has 'Distance' and 'Counts' set to 4000. The 'Position loop' section has 'Gain 1' and 'Gain 2' set to 1000, 'Threshold 1-2' set to 500, and 'Offset' set to 0. The 'Movement control' section has 'Following error 1' and 'Following error 2' set to 9000, 'Recalibration position' set to 0, 'Recalibration deviation' set to 0, 'Overspeed' set to 12, and 'VLim' set to 0. The 'Magual mode parameters' section has 'Speed' set to 2700 and 'Origin value' set to 225000. The 'Command' section has 'Software Hi limit' set to 300000 and 'Software Lo limit' set to 0. The 'Acceleration' section has 'Smax' set to 300 and 'Acceleration profile' set to 'Rectangle'. The 'Stop control' section has 'Delay' set to 320 and 'Speed' set to 5400.

The following table summarizes the parameters modified in this example. The other parameters have retained their default values.

Parameter	Value
Target window	320 $\mu$ m
Speed (manual mode)	5400 mm/min
RP value	0 $\mu$ m


- Confirm the values entered with the **Edit/Confirm** command, or click on the .

- Select channel 1 in the channel zone.

The screenshot shows the 'TSX CAY 33 [RACK 0 POSITION 3]' configuration window. The 'Adjust' dropdown is set to 'Adjust'. The designation is '3 CHAN. AXIS CONT. MOD.'. The symbol section shows 'Channel 1' selected for 'Position control'. The 'Corrected resolution' section has 'Distance' and 'Counts' both set to 4000. The 'Position loop' section has 'Gain 1' and 'Gain 2' both at 1000 /100s, 'Threshold 1-2' at 8000 /1000 of Smax, and 'Offset' at 8388 mV. The 'Movement control' section has 'Following error 1' and 'Following error 2' both at 9000, 'Recalibration position' and 'Recalibration deviation' both at 0, 'Overspeed' at 12 % of Smax, and 'VLim' at 0 mV. The 'Manual mode parameters' section has 'Speed' at 2700 and 'Origin value' at 225000. The 'Command' section has 'Software Hi limit' at 900000 and 'Software Lo limit' at 0. The 'Acceleration' section has 'Smax t' at 300 ms and 'Acceleration profile' set to 'Rectangle'. The 'Stop control' section has 'Delay' at 500 ms, 'Speed' at 5400, and 'Target window' at 9000.

The following table summarizes the parameters modified in this example. The other parameters have retained their default values.

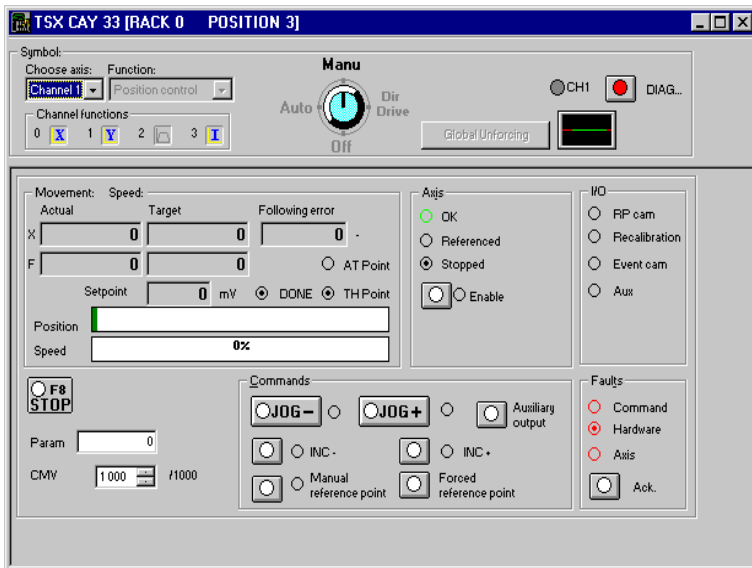
Parameter	Value
Encoder offset	8 388 607
Deviation 1 and 2	8000 $\mu$ m
Target window	8000 $\mu$ m
Speed (manual mode)	5400 mm/min

- Confirm the values entered with the **Edit/Confirm** command or click on the  icon.
- Save these values in the PLC processor by selecting the **Services/Save parameters** command.




## Control in manual mode

If the user wishes to move the moving part without first performing the programming stage, select Manual mode. To do this, access the debug screen, in online mode : activate the **Tool/Configuration** command then select the TSX CAY module to be opened and execute the **Services/Open module** command (or double click on the module to be opened). The debug screen is then displayed by default.



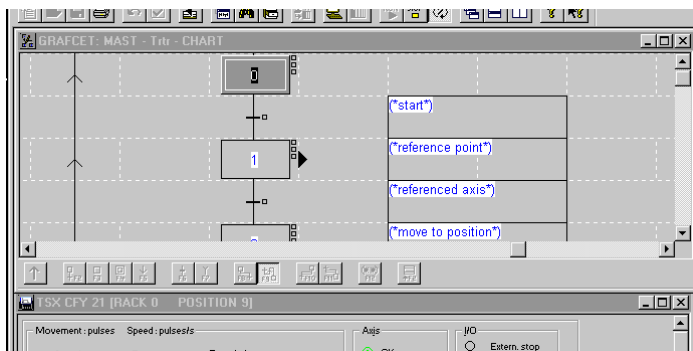
Perform the following operations in the debug screen :

- Set the PLC to RUN (**PLC/Run** command or click on the  icon).
- Select the axis to be controlled : channel 0 (axis X) or channel 1 (axis Y).
- Select manual mode by setting the mode switch to **Man**.
- Click on the **Enable** button in the **Axis** zone (enables the speed drive safety relay).
- Acknowledge any faults by clicking on the **Ack** button in the **Faults** field.
- Set a reference point :
  - either by selecting the **Set manual reference point** command
  - or by selecting the **Set forced reference point** command. In this case, first enter (in the **Param** field) the value of the position of the moving part in relation to the reference point.
- Perform movements in a positive direction using the **JOG+** command or in a negative direction using the **JOG-** command. The position of the moving part is then displayed in field **X** and the speed in field **F** in the **Movement / Speed** zone.

## Debugging

To debug the program :

- Set the PLC to RUN.
- Display the TSX CAY module debug screen.
- Display the Grafcet screen at the same time in order to follow the evolution of the sequential processing.
- Start the execution of the program by pressing the "Start\_cycle" button on the front panel.



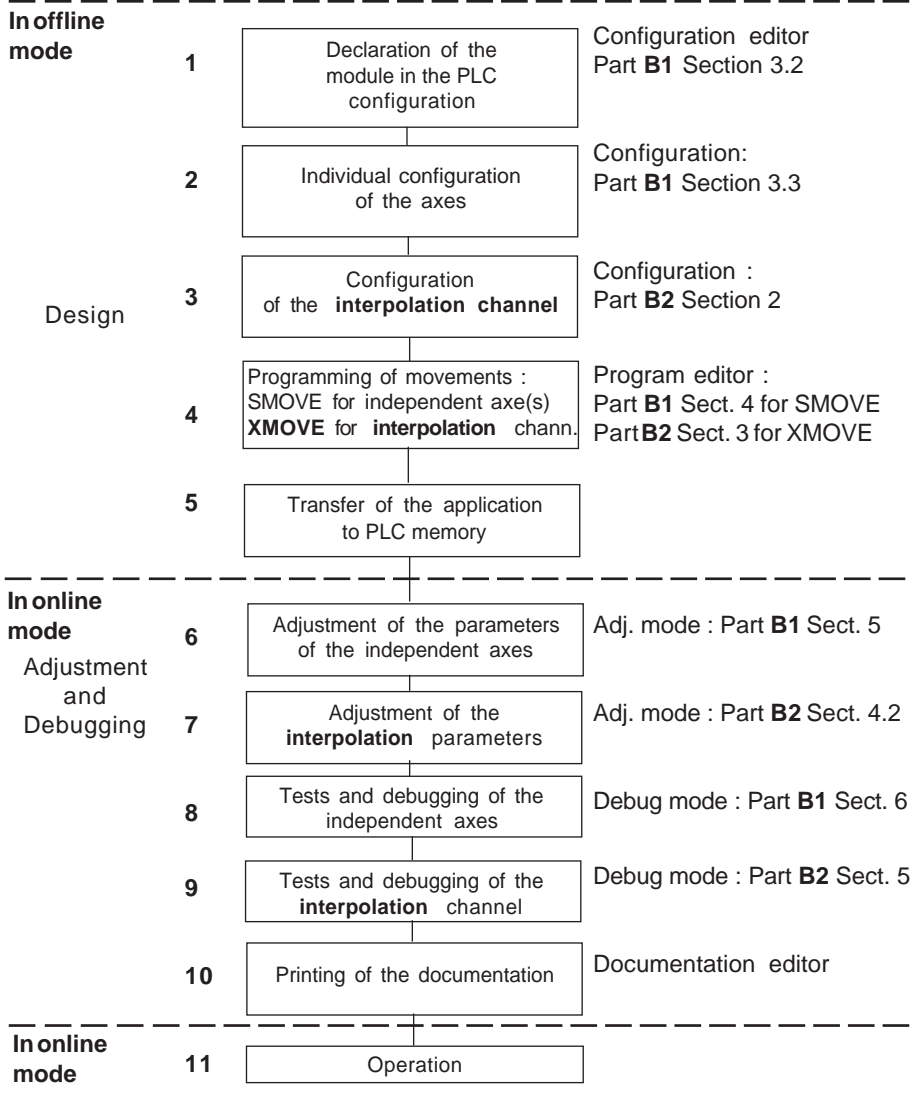
### 1.2-9 Archiving

Once debugging of the program has been completed :

- Save the parameters again if they have modified during debugging, by selecting the **Services/Save parameters** command.
- Transfer the application from the PLC processor to the disk for archiving : **PLC/ Transfer command**, "PLC --> terminal" option. Then execute the **File/Save As** command, give the application a name and confirm.

### 1.3 Methodology for setting up interpolation

The following flow chart summarizes the various stages.



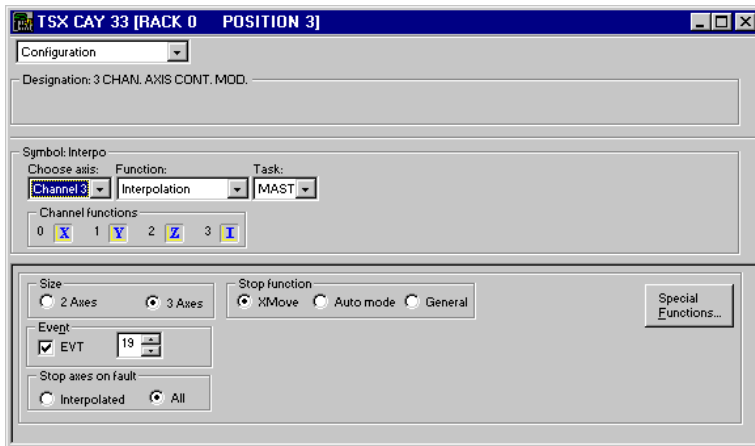


## 2.1 Accessing parameters in the configuration screen

### Prior action :

Channel no. 3 is dedicated to interpolation. It is essential to **configure the other channels** before configuring **channel 3**.

- Select the TSX CAY 33 module and confirm.



## 2.2 Entering parameters

### Select axis/Function/Task fields

Access Select Axis **channel 3** :

- Select the **Interpolation** function.
- Select the MAST or FAST task associated with channel no. 3.

The MAST or FAST task must be the same for this channel and the channels involved in interpolation. On confirmation a dialog box displays the number of any channel which does not have the same task as this channel.

The **type of axis** for the channels involved in **interpolation** must be **limited**. It is not possible to interpolate infinite axes.

On confirmation a dialog box displays the number of any channel which is not of the limited axis type.

## Channel functions field

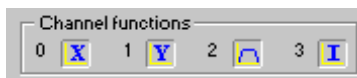
The **Channel functions** zone provides information on the axes involved in interpolation.

The diagram opposite shows that :

- **channel 0** is axis **X**
- **channel 1** is axis **Y**
- **channel 2** is axis **Z**
- **channel 3** is the **interpolation** ("I") channel for axes X, Y and Z



Channel 2 can be configured as an independent channel. The character "Z" does not appear in field 2, and a curve is displayed there.



In this case, **interpolation** is only performed on **channels 0 and 1**.

The functions are displayed in the interpolation debug screens.

**Dimension field** : is used to define the number of interpolated axes :

- 2 axes : channels 0 and 1
- 3 axes : channels 0, 1 and 2

**Stop function field** : defines the role of the STOP command for channel no. 3 (%Qxy.3.15):

- **XMOVE** : only affects an XMOVE command which is in progress
- **Automode** : STOP command active on all axes which can be interpolated, even when they are used independently, but only in AUTO mode
- **General** : STOP command active on all axes which can be interpolated, even when they are used independently, whatever the mode (AUTO, MAN, etc)

Recommendation : by default choose XMOVE.

**Event field** : is used to define the event-triggered task associated with channel 3.

**Stop axes on fault field** : effect of a blocking fault.

**Interpolated** : stops the axes involved in the current XMOVE command.

**All** : stops all axes which can be interpolated, even if they are being used independently at that time.

Recommendation : by default choose Interpolated.

**Special functions field** : reserved use.

---

## 2.3 Confirming the configuration parameters

---

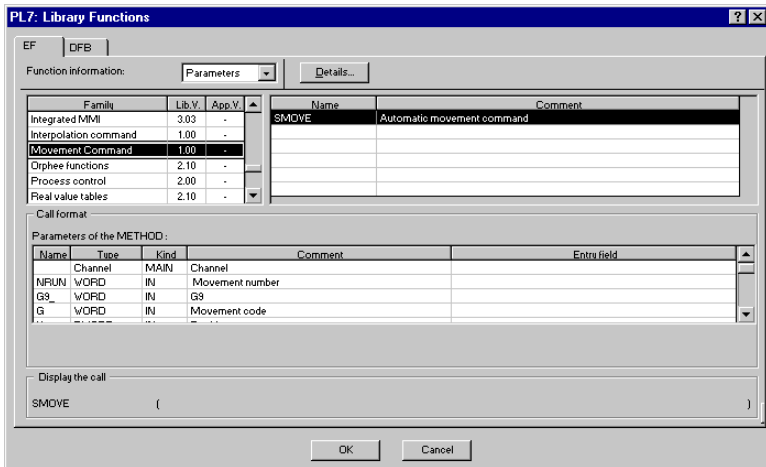
When all the parameters have been configured, confirm the configuration obtained using the **Edit/Confirm** command or select the  icon.

## 3.1 Programming interpolated movements : XMOVE function

### 3.1-1 Programming an XMOVE function

An interpolated movement is programmed using the XMOVE instruction which is systematically sent on channel 3 of the TSX CAY 33 module.

It is entered either directly, or via the "Function call" assisted entry screen.



### Assisted entry

In the program editor :

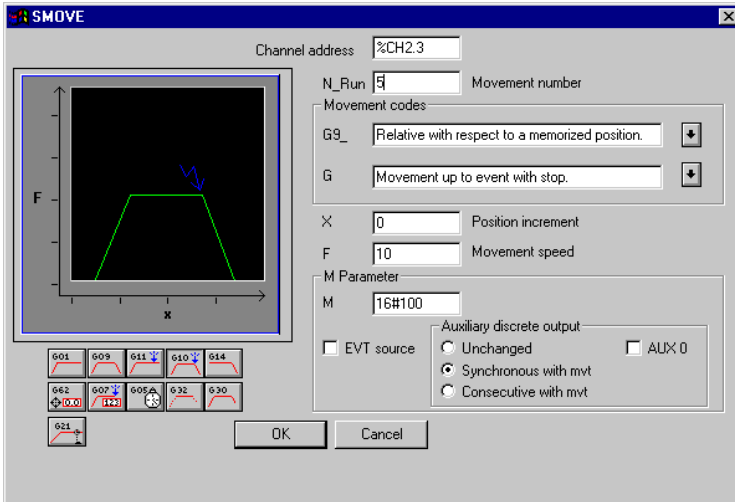
- 1 Press the **SHIFT+ F8** keys simultaneously or click on the **F(...)** icon. The **Library functions** window is then displayed.
- 2 Select the **Parameters** option.
- 3 Select the **Interpolation Command** family.
- 4 Select the **XMOVE** function.
- 5 Press the **Details...** button and define the various fields (see description in the following pages) or enter the variables of the function directly in the parameter entry zone.
- 6 Confirm with **OK** or **ENTER**.

### 3.1-2 Entering parameters for the XMOVE function

A movement command is programmed by an XMOVE function, with the following syntax :

**XMOVE%CHxy.3(N\_Run,G9\_,G\_,SPACE,X,Y,Z,F,M)**

The XMOVE function **Details** screen provides assisted entry for each of the fields,



with

**%CHxy.3** = **Channel address** of the axis control module in the PLC configuration.

**x** = rack number

**y** = position of the module in the rack

**3** = interpolation channel.

**N\_Run** = **Movement number** from 0 to 32767. Number identifying the movement performed by the XMOVE function. In debug mode it identifies the current movement.



## Movement codes

**G9\_** = type of movement

**90** Move to an **absolute** position

**91** Move to a **relative** position **with respect to the current position**.

**98** Move to a **relative** position **with respect to the memorized position PREF1**.

Select the type of movement using the scroll button to the right of field **G9\_** or enter the code directly during a direct entry (without using the details screen).

**G** = instruction code

**09** Move to a position and stop

**01** Move to a position without stopping

**10** Move until an event is detected and stop

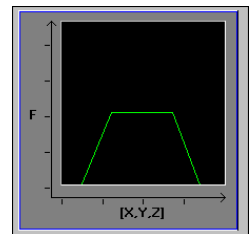
**05** Await an event

**92** Initialize the PRF1 registers of axes X, Y, Z



Select the instruction code using the scroll button to the right of field **G**, or press the corresponding icon, or enter the code directly during a direct entry (without using the details screen).

In the details screen : a graphic is displayed representing the selected movement (eg : code 09).



**SPACE** defines the number of the plane or space in which the movement is to be performed. It gives a list of the axes involved in the movement :

**0** movement in the XY plane

**1** movement in the XZ plane

**2** movement in the YZ plane

**3** movement in the XYZ space

### Special case of a 2-dimensional group

When the group of interpolated axes is 2-dimensional, the **SPACE** field must be zero. Field **Z** of the XMOVE function has no significance and is ignored.

**X, Y, Z** = coordinates of the position to be reached.

This position can be :

- immediate
- coded on internal double words %MDi or internal constants %KDi (these words can be indexed)

The unit in which these values are expressed is defined by the **Length Units** configuration parameter.

**F** = speed (1) of movement of the moving part. This speed can be :

- immediate
- coded in a double word %MDi or constant %KDi (this word can be indexed)

**M** = word coded on four-bit bytes (in hexadecimal) 16# 

3	2	1	0
---	---	---	---

- for optionally activating triggering of the application event processing for instructions 05 and 10 (4-bit byte no. 3 at 1 for activation)

- the list of events which can terminate the G05 or G10 instruction :

four-bit byte no. 1 :

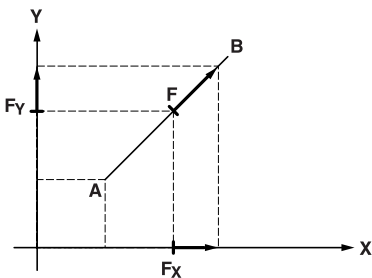
- bit 0 for the event input or the EXT\_EVT bit of axis X
- bit 1 for the event input or the EXT\_EVT bit of axis Y
- bit 2 for the event input or the EXT\_EVT bit of axis Z
- bit 3 for the EXT\_EVT bit of the group

If more than one bit is at 1, the event which ends the instruction is the first event in the list which occurs : an OR of the events is performed.

This is coded automatically in field **M** in the **Details** screen using the check boxes on this screen.

(1) Required speed of movement in the direction of the movement to be performed.

Example for a 2-axis system :



Based on this speed, the interpolator calculates projections Fx and Fy which will enable both axes to follow the trajectory.

Note : The actual speed of movement is equal to the required speed F multiplied by the Velocity Correction Factor, whose value can be adjusted within the range [0.001, 2.000]

---

### 3.1-3 Description of elementary movements

2 classes of movement can be programmed :

- move to a position (instruction codes 01 and 09)
- move until an event is detected (instruction code 10)

When programming these movements, the user determines the positions to be reached, the speed and the plane or space of the interpolation. The acceleration parameters are defined by adjustment.

The movements can be :

- absolute (movement with respect to the machine reference), code **90**

Example :    move without stopping in the plane (X, Y)  
                  to the position (50000,10000) at speed 1000  
                  XMOVE %CH102.3 (1,**90**,01,0,50000,10000,0,1000,0)

- relative to the current position, code **91**

Example :    move without stopping in the plane (X, Y)  
                  by one increment (+ 2000, -1000) with respect to the current position at speed  
                  500  
                  XMOVE %CH102.3 (1,**91**,01,0,1,2000,-1000,0,500,0)

- relative with respect to the memorized position PREF, code **98**

Example :    move without stopping in the plane (X, Y)  
                  by one increment (+ 5000, + 2000) with respect to the previously memorized  
                  position at speed 800  
                  XMOVE %CH102.3 (1,**98**,01,0,5000,2000,0,800,0)

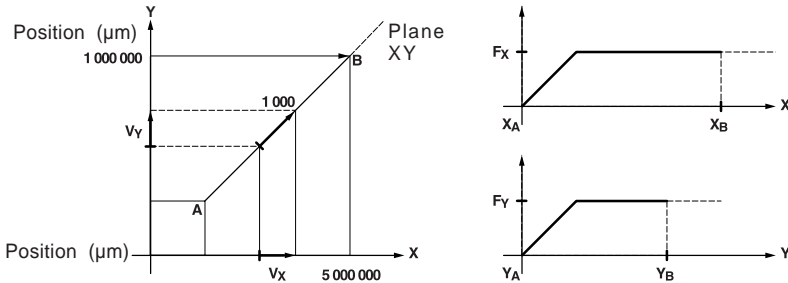
### 3.1-4 Description of the instructions

For simplicity, all the examples concern a 2-axis interpolated movement.

**Move to a position without stopping** : instruction code **01**



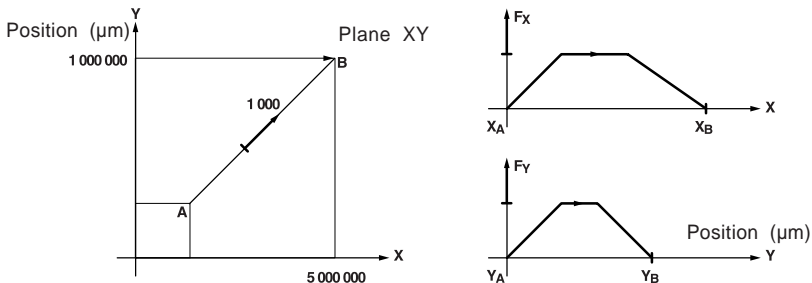
Example 1 : XMOVE %CH102.3 (1,90,01,0,5000000,1000000,0,1000,0)



**Move to a position and stop** : instruction code **09**



Example 2 : XMOVE %CH102.3 (1,90,09,0,5000000,1000000,0,1000,0)



**Move until an event is detected and stop : instruction code 10**

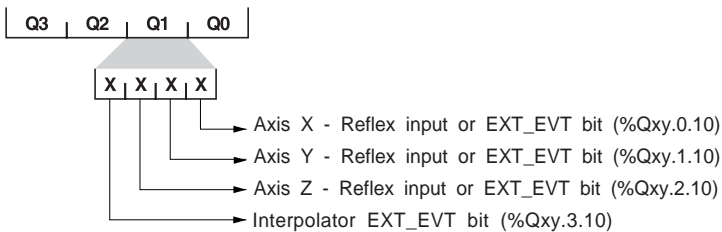


Instruction 10 triggers movement of the axes until an event is detected or up to the specified position if no event occurs :

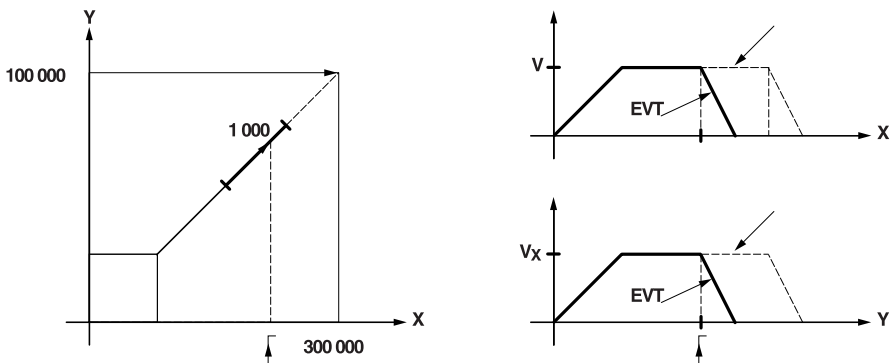
The event which is awaited can be :

- a rising or falling edge (depending on the selection made at configuration) on one of the reflex inputs of one of the interpolated axes
- a command from the application which may be :
  - a rising edge on the EXT\_EVT bit of one of the axes
  - a rising edge on the EXT\_EVT bit of the interpolator

Four-bit byte number 1 of code M is used to specify the axis (or axes) on which the event is awaited.



Example : Move in the plane (X, Y), until an EVT is detected on a reflex input on axis X, and stop the axes (if no EVT occurs) at (300000, 100000) at speed 1000 with activation of the event-triggered task when the EVT is detected.



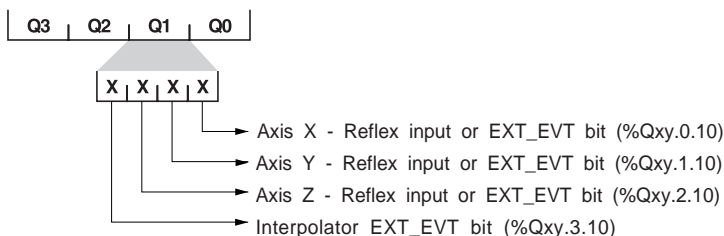
**Await an event** : instruction code **05**

This instruction is used to await an event with a time-out defined in parameter F in ms. If the event has not appeared within the time-out, the await event command is deactivated. If parameter F is defined as 0, the waiting period is not limited.

The event which is awaited can be :

- a change of state on a reflex input of one of the interpolated axes
- a command from the application

Four-bit byte number 1 of code M is used to define the axis (axes) on which the event is awaited.



This instruction can activate an event-triggered task on detection of the event if four-bit byte no. 3 of parameter M is set to 1.

Bit TO\_G05 (%lxy.i.49) is set to 1 when the time-out has elapsed with no event detected if activation of the event-triggered task has been requested.

Example : wait with a time-out of 1.5 s and activation of the event-triggered task.

```
XMOVE %CH102.3 (1,90,05,0,0,0,0,1500,16#1000)
```

**Load PRF1 register instruction** : instruction code **92**

Instruction G92 initializes the PRF1 registers of the various axes. These registers are used by the relative movement instructions : code G98. Four-bit byte number 1 of code M is used to select the list of axes affected by this initialization.

- Bit 0 for axis X
- Bit 1 for axis Y
- Bit 2 for axis Z

Example : Loading of the PRF1 registers of axes X and Y respectively at 2000 to 4000

```
XMOVE %Chxy.3 (1,90,92,2000,4000,0,0,16#0030)
```

### General acceptance conditions

The general acceptance conditions of the XMOVE function are as follows :

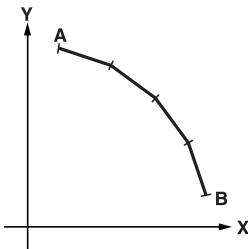
- There is no fault. Bit **GP\_OK = 1**
- The configuration is suitable. **CONF\_OK = 1**
- The axes are referenced. **REF\_OK = 1**
- The axes involved in the movement are in automatic mode with **DONE = 1**, **ENABLE = 1**, and are stopped.

### 3.1-5 Sequencing movement commands

The TSX CAY 33 module does not provide circular interpolation. It is however possible to approximate any trajectory using a series of segments.

Example :

In a 2-axis system



Each elementary segment has a corresponding XMOVE command.

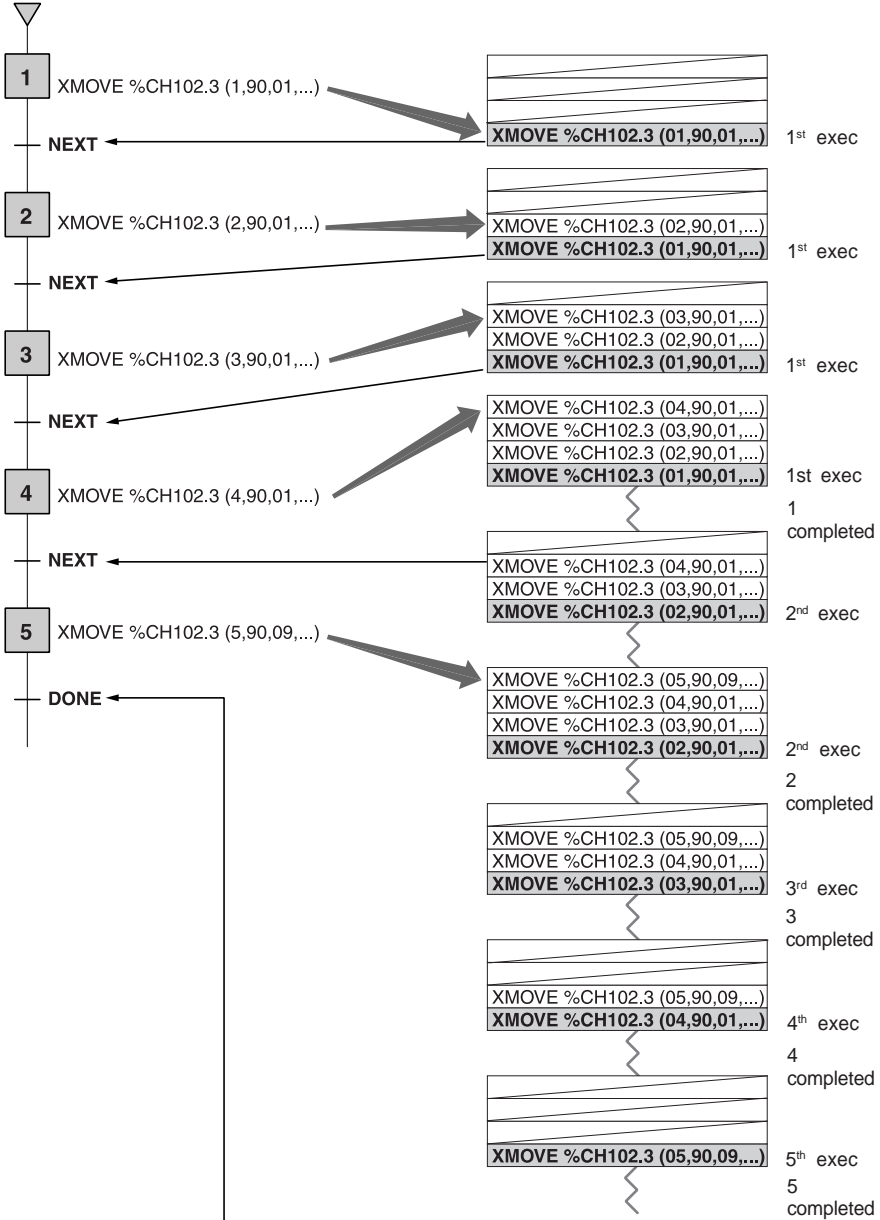
Each elementary XMOVE command is performed once only, so the execution must be programmed in :

- Grafset : in a step, on activation or deactivation of that step
- Structured Text or Ladder language, on the rising edge of a bit.

The execution report for the function is provided by the module, via the NEXT and DONE bits (see table on the next page).

The TSX CAY 33 module has a mechanism for sequencing movement commands.

The interpolator has a buffer memory or "stack" which can hold **3 movement commands** in addition to the command it is in the process of executing. Thus at the end of execution of the current command, it goes directly to the first command in the buffer memory.





---

When the stack is empty and a G1 type movement is requested, the movement will not start until the module has received the next movement.

2 movement commands can be sequenced as follows :

- immediately if the first movement does not include a stop
- as soon as the moving part is in the target window or at the end of the TSTOP delay defined in the stop control (parameter adjustment screen) if the first movement includes a stop.

For sequencing to occur immediately, the execution time of the current instruction must be greater than the period of the task in which the XMOVE commands are programmed.

The failure of an XMOVE command is indicated by the following bits :

- CMD\_NOK (%Ixy.i.3.6) indicates a failure
- CMD\_FLT (%MWxy.i.3.7) indicates the cause of the failure (requires a READ\_STS instruction).

### Restrictions on XMOVE movements which can be sequenced

All the examples below result in a command failure (CMD\_NOK), stopping of the moving part and **resetting of the buffer memory** :

- sequencing a G05 or G92 after a G1 instruction
- no instruction after a G1
- receipt of a movement with the **SPACE** parameter involving an axis which is not stopped while the preceding XMOVE does not involve that axis (example of an XMOVE instruction with an axis for which the last movement was an SMOVE G1 instruction).

### Bits associated with the sequencing mechanism

Addressing	Description
NEXT %lxy.3.0	Indicates to the user program that channel 3 is ready to receive the next XMOVE command.
DONE %lxy.3.1	Indicates the end of execution of the current command and that there is no new command in the buffer memory.
TH_PNT %lxy.3.10	Indicates that the setpoint value has been reached on the axes involved in the XMOVE.
AT_PNT %lxy.3.9	At the end of a movement with stop, indicates that for all the axes involved in the movement, the moving part is in the target window or the <b>TSTOP</b> period has elapsed.

### Notes

The NEXT or DONE bit should always be tested before executing an XMOVE command.

A new command can only be transmitted to the module if the buffer memory associated with the axis to be controlled is not full.

Word SYNC\_N\_RUN (%lWxy.3.8) periodically provides the current step number in order for movements to be sequenced.

**3.1-6 Using the XMOVE function and the SMOVE function together**

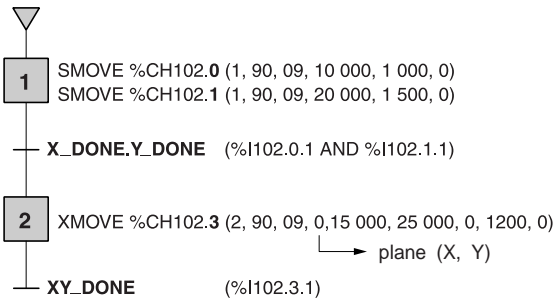
It is possible to mix movement instructions concerning a single axis (SMOVE) and instructions concerning several axes (XMOVE) in the same program. This makes it possible to alternate interpolated movements with non-interpolated movements.

When the user has configured axes which can be interpolated, he can still use the axes independently.

The program must refer to the objects of the axis concerned before sending an SMOVE instruction and refer to the channel 3 objects before sending an XMOVE instruction.

Example :

Independent movement of axes X and Y followed by an interpolated movement.



An interpolated movement concerning an axis i prevents all control of that axis via SMOVE : an XMOVE instruction which is in progress forces the **NEXT** and **DONE** bits of the axes concerned to 0.

In addition, a data bit **IN\_INTERPO** (%Ixy.i.32), which can be accessed from the application program, is set to 1 for all the axes executing movements linked with an XMOVE instruction. This assists with programming and supervision.

### 3.1-7 Interpolator channel automatic mode

**Automatic** mode is the active mode for interpolated axes. It is the only mode in which interpolated movements can be executed.

This mode is mainly used to send a movement command (code G), via the XMOVE function, for executing an interpolated movement by briefly creating a link between a number of axes.

Channel 3 changing to AUTO mode does not change the current mode or the current commands of the 2 (or 3) axes of the module. Thus movements/debugging operations which are executed axis by axis (independently) in Man, Dirdrive and even Automatic mode continue to be performed via the position control function of each of the module axes.

The actual change is indicated by bit **IN\_AUTO** (%Ixy.3.23).

In automatic mode, commands can be applied to the XMOVE function :

- **CMV** : velocity correction factor. This affects the current tangential speed reference in a ratio of 1/1000 to 2000/1000 (%QWxy.3.1).
- **CMV = 0** : immediate pause command which stops the moving part, while still ensuring that on the command to restart the movement (CMV # 0) the programmed trajectory is followed. The status is indicated in bit **IM\_PAUSE** (%Ixy.3.34).
- **Pause** : suspends sequencing of the XMOVE movements. The pause only becomes active when the moving part is stopped. The state is indicated by bit **ON\_PAUSE** (%Ixy.3.33).
- **MOD\_STEP** (%Qxy.3.19) : is used to execute a sequence of movements, stopping after each elementary instruction. The status is indicated in bit **IN\_STEP** (%Ixy.3.39).

Bit **NEXT\_STEP** (%Qxy.3.22) is used to execute the next step.

- **EXT\_EVT** (%Qxy.3.10) : stops a G05 or G10 instruction.

These commands are similar to those for the position control function in relation to SMOVE (see section 4 part B1).

Automatic mode also provides access to two other commands which are active during and beyond an XMOVE instruction :

- **STOP** : command to stop the various axes involved in the interpolation (depending on the Stop role defined at configuration).
- **ACK\_DEF** : a rising edge triggers fault acknowledgment on all the axes.

### 3.1-8 Event processing

Channel 3 of the TSX CAY 33 module can activate an event-triggered task. For this, the function must be enabled in the configuration screen and an event processing number associated with the channel must be defined.

#### Activation of an event-triggered task

The event-triggered task is activated by the appearance of the event awaited by G10 and G05 commands if 4-bit byte no. 3 of parameter M of the XMOVE function associated with the instruction is at 1.

#### Variables which can be used by the event-triggered task

- The following bits are used to determine what triggered the application event processing :
  - EVT\_G1 (%Ixy.3.50) event during a G10 instruction
  - EVT\_G05 (%Ixy.3.48) event during a G05 instruction
  - TO\_G05 (%Ixy.3.49) G05 time-out elapsed.
- Bit OVR\_EVT (%Ixy.3.46) detects a delay in transmission of the event or loss of the event.

#### Note

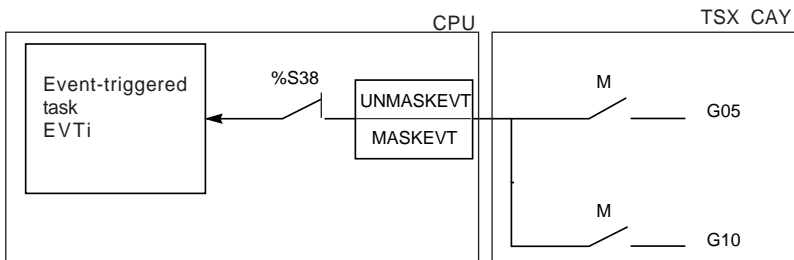
The bits and words described above are the only values which are updated when the event-triggered task is executed and they are only updated in the PLC if the event-triggered task is activated.

#### Event masking

PL7 language offers 2 ways of masking events :

- MASKEVT instruction for global masking of events (the UNMASKEVT instruction unmaskes them)
- bit %S38 : 0 = global inhibition of events (bit normally at 1).

#### Summary diagram



## 3.2 Fault management

---

### 3.2-1 Role

Checking for faults is of primary importance in the axis control function because of the inherent risks when moving parts are in motion.

The checks are performed internally and automatically by the module.

#### Monitoring interpolated axis faults

Channel 3 of the TSX CAY 33 module does not have any faults of its own.

It sends back fault data which is the OR of the faults of the configured axes (channels 0 and 1 if there is 2-axis interpolation, channels 0 to 2 if there is 3-axis interpolation).

This data is as follows :

• emergency stop	%MWxy.3.3: X5
• speed drive fault	X2
• encoder wiring break	X4
• analog output short-circuit	X0
• auxiliary output short-circuit	X1
• encoder supply	X3
• 24 V supply	X6
• absolute encoder frame	X7
• lower soft stop	X9
• upper soft stop	X8
• overspeed	X10
• blocking deviation	X11
• non-blocking deviation	X15
• recalibration on the fly deviation	X12

For the following faults :

• target window TW	X13
• stop speed TSTOP	X14

Only the axes which have been moved via XMOVE are taken into account in the generation of the fault data.

#### Note

See part B1 for the description of channel faults.

Reminder : The fault data is only updated when the READ\_STS %CHxy.3 instruction is executed.

---

### 3.2-2 Principle

Faults are classified into two levels of severity :

- **Critical or blocking faults** which cause the moving part to stop. The following occurs :
  - fault indication
  - deceleration of the moving part until the analog output is zero
  - deactivation of the speed drive enable relay
  - deletion of all the commands which have been memorized
  - wait for acknowledgment.

The fault must have disappeared and been acknowledged before the application can be restarted.

- **Non-critical faults** which cause a fault indication without stopping the moving part. The action to be carried out when such a fault occurs is left up to the user in the PL7 program.  
The fault indication disappears when the fault has disappeared and has been acknowledged.

### 3.2-3 Programming

Faults can be displayed, corrected and acknowledged from the debug screen, but it may be useful during operation to be able to control the moving part and correct faults from an operator panel.

#### Fault indication

The module provides a large amount of data via status bits and words which can be accessed via the PL7 program. These bits make it possible to deal with faults hierarchically :

- so that action can be taken on the main program
- simply to indicate the fault.

There are two levels of indication :

<b>• 1st level : general data</b>	
<b>%Ixy.3.ERR</b>	: channel fault
<b>AX_OK</b>	: (%Ixy.3.3) no blocking fault (with stop of moving part) is detected
<b>AX_FLT</b>	: (%Ixy.3.2) fault (covers all faults)
<b>HD_ERR</b>	: (%Ixy.3.4) external hardware fault
<b>AX_ERR</b>	: (%Ixy.3.5) application fault
<b>CMD_NOK</b>	: (%Ixy.3.6) command failure
<b>• 2nd level : detailed data</b>	
Channel fault status words %MWxy.3.3	

In general it is advisable to stop the evolution of the sequential processing associated with the axes when a blocking fault occurs and to correct the fault. Correction of the fault should be followed by an acknowledgment.

#### Acknowledging faults

When a fault appears on one of the interpolated axes :

- the fault bit associated with the axis changes to 1 : AX\_FLT (%Ixy.i.2), HD\_ERR (%Ixy.i.4), AX\_ERR (%Ixy.i.5), STATUS (%MWxy.i.3:Xj) as well as the fault bit corresponding to the interpolation channel AX\_FLT (%Ixy.3.2) HD\_ERR (%Ixy.3.4), AX\_ERR (%Ixy.3.5), STATUS (%MWxy.3.3:Xj).
- if it is a blocking fault, the OK bit changes to 0.

When the fault disappears, the state of all the fault bits remains unchanged. The fault is memorized until it is acknowledged by setting bit ACK\_FLT %Qxy.i.8 (where i is the number of the channel on which the fault is located) to 1 or by the ACK\_FLT (%Qxy.3.8) command of the interpolator channel (which triggers an acknowledgment on all the interpolated axes). The fault must be acknowledged after it has disappeared (except in the case of soft stop faults).

If several faults are detected, the acknowledgment command only applies to those faults which have totally disappeared. Faults which remain must be acknowledged again when they have disappeared.

#### Note

Channel 3 (interpolator) does not memorize faults.



**3.2-4 Description of command failure faults**

A command failure fault is generated every time a command cannot be executed, whether this is due to the fact that this command is not compatible with the status of the axis or the current mode, or because at least one of the parameters is missing from the validity area.

These faults are indicated by the Cmd Fail indicator lamp in the debug screens. The channel DIAG button is used to establish the source of the command failure. The source can also be accessed by the program via bit CMD\_NOK (%Ixy.3.6) and word CMD\_FLT (%MWxy.3.7).

**Fault : Command failure**

<b>Cause</b>	<ul style="list-style-type: none"> <li>• Unauthorized movement command</li> <li>• Incorrect configuration or parameter transfer</li> </ul>		
<b>Parameter</b>	–		
<b>Consequence</b>	<ul style="list-style-type: none"> <li>• Immediate stopping of the current movement</li> <li>• Resetting to 0 of the buffer memory receiving the movement commands in automatic mode</li> </ul>		
<b>Indication</b>	<ul style="list-style-type: none"> <li>• CMD_NOK bit (%Ixy.3.6) movement command failure</li> <li>• CMD_FLT word (%MWxy.3.7) word coding the type of fault detected CMD_FLT</li> </ul> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">configuration</td> <td style="text-align: center;">Commands</td> </tr> </table> <p style="text-align: center;">A more detailed description is given in section 7</p>	configuration	Commands
configuration	Commands		
<b>Remedy</b>	<ul style="list-style-type: none"> <li>• Implicit acknowledgment on receipt of a new accepted command</li> <li>• Acknowledgment is possible using the ACK_FLT command (%Qxy.3.8)</li> </ul>		

**Note**

When sequencing movements in automatic mode, it is advisable to make the execution of each movement conditional on the end of execution of the previous movement, using bit AX\_FLT (%Ixy.3.2). This ensures that a link is not made to the following command when the current command has failed.

---

### 3.3 Managing OFF mode

---

OFF mode is the passive mode for the interpolator (axes X, Y and Z are in independent axis status, and therefore it is possible to control them via their respective Dir\_Drv, Man and Auto modes).


In this mode no channel 3 command is accepted apart from the fault acknowledgment command.

Changing to OFF mode causes the current XMOVE to stop if one is in progress.

## 4.1 Accessing the adjustment parameters

The adjustment parameters are accessed using the **View/Adjust** command in the TSX CAY 33 module configuration screen (or by selecting **Adjust** in the module zone in the parameter configuration or debug screen).

Select the channel to be adjusted in the **Select axis** field : channel 3.

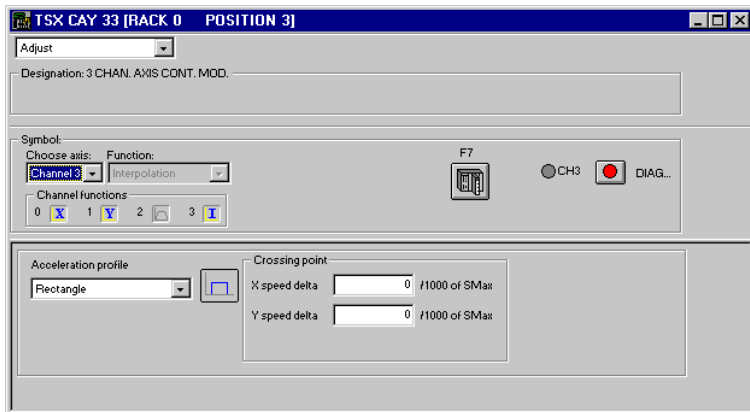
The  button is used to display either current or initial parameters.

The **initial parameters** are :

- Parameters entered (or defined by default when confirming the configuration) in offline mode and provided when transferring the program to the PLC
- Parameters taken into account at the last reconfiguration in online mode




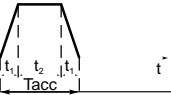

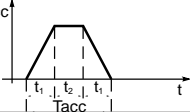

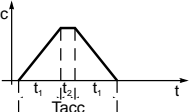

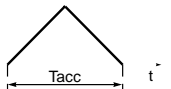
The **current parameters** are those modified and confirmed from the adjustment screen in online mode (or via the program, by explicit exchange). These parameters are replaced by the initial parameters on a cold restart.

The screen offers the following adjustment parameters :



## 4.2 Acceleration profile

**Acceleration profile** : the acceleration profile is common to all the interpolated axes.

Possible profile	Value	Icon	Description
Rectangle	0		$Acc^t$  $t_1 = 0$ $T_{acc} = T_{accrec}$
Trapezoid	1		$Acc^t$  $t_2 = 3t_1$ $T_{acc} = 1.25 T_{accrec}$
Trapezoid	2		$Acc^t$  $t_1 = t_2$ $T_{acc} = 1.5 T_{accrec}$
Trapezoid	3		$Acc^t$  $t_1 = 3t_2$ $T_{acc} = 1.75 T_{accrec}$
Triangle	4		$Acc^t$  $T_{acc} = 2 T_{accrec}$

The acceleration profile defines a common acceleration profile for all the axes involved in an interpolated movement (replaces the current parameter of the axis during the interpolated movement).

This field is used to enter the type of acceleration profile followed by all the axes involved in an interpolated movement.

---

### 4.3 Points through which the axes pass

---

The **Speed difference X, Speed difference Y, Speed difference Z parameters** (one parameter per interpolated axis) define the permitted speed variation for each of the axes at the points through which they pass.

They are used to adjust the speed of the moving part at the point through which the axis passes. This adjustment enables the moving part to pass as close as possible to the target point when a low value is used. This parameter is expressed in thousandths of VMax. The speed difference values vary from 0 to 500.

### Processing the points through which the axes pass

In a linear interpolation the passing point concept appears when a series of type G1 movements without stopping is performed.

For example, to obtain trajectory ABC. If the speed specified on segment AB is maintained up to position B, there is an overshoot (fig. a). If this speed is reduced before arriving at B, the actual trajectory remains within the angle ABC (fig. b).

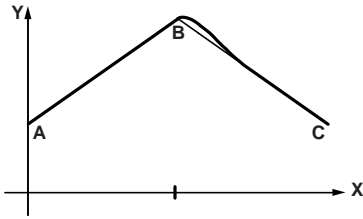


fig. a

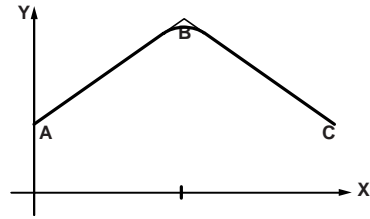
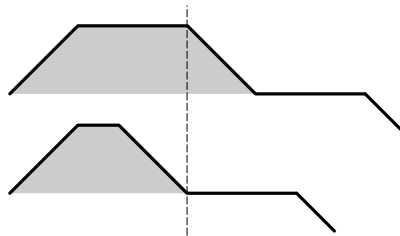


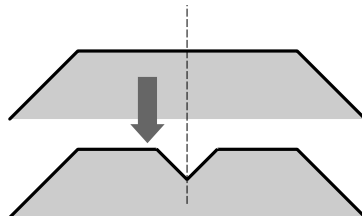
fig. b

To avoid overshooting :

- Count on the natural delay (deviation) of each of the axes. Consequently, it is advisable to limit adjustment of the velocity feedforward gain **KV** during interpolated movements.
- In a sequence (G1, X1, Y1, Z1, F1) followed by (G1, X2, Y2, Z2, F2), if F2 is smaller than F1 then the speed trajectory is modified so that the required speed at the break point is equal to F2 :

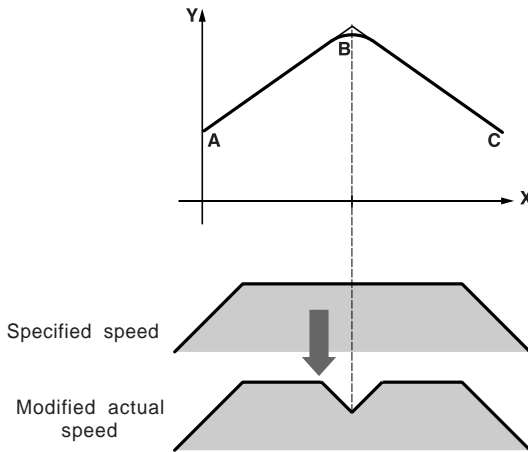


- The speed is reduced at the passing point when the axial speed difference is greater than can be accepted by one of the axes.



In order to be able to approach as close as possible to the passing point :

- The speed variation for each of the axes at the points through which they pass is made adjustable using the **Speed difference** parameter. Thus, by reducing this parameter, the user decreases the speed at the passing point, which enables the axis to pass as close as possible to the target point.







## 5.1 Principle of debugging a program with interpolation

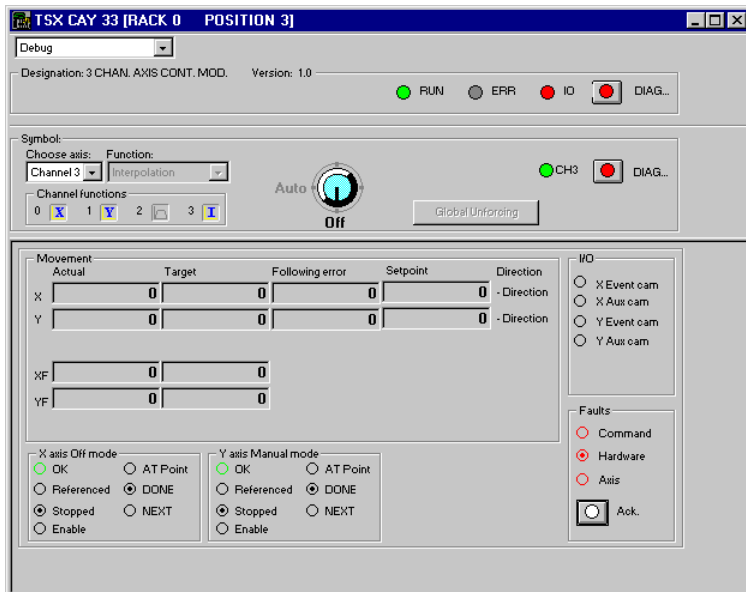
Axis control, which is integrated in the PL7 program, uses the PL7 debug functions.

### Reminder of the options offered by PL7

- Realtime display and animation of the program.  
In Grafcet : by programming each movement in a Grafcet step, it is easy to identify the current movement.
- Insertion of breakpoints and execution cycle by cycle, rung by rung or statement/sequence by statement/sequence.
- Access to animation tables. This makes it possible to display status bits and words and to control interpolation channel command bits. It also enables bit objects to be forced and the evolution of the Grafcet chart to be blocked.

PL7 offers a debug screen, specific to the TSX CAY 33 axis control module, which provides the user with all the necessary data and commands.

### TSX CAY 33 axis control debug screen



One of two screens is used for debugging the interpolation, depending on the operating mode chosen using the mode selector : Automatic (AUTO) or OFF.

---

## 5.2 Debug screen

---

### 5.2-1 Accessing the debug screen

The terminal must be in online mode (connected to the PLC).

To access the axis control module debug screen, select the **Configuration** editor then select and confirm the rack position containing the axis control module.

In online mode the debug screen is selected by default.

---

### 5.2-2 User interface

**Control buttons:** 

- For status commands :

Pressing then releasing a button activates the associated command. The indicator lamp in the button lights when this command is taken into account (the corresponding command bit %Q is set to 1).


Pressing then releasing the button a second time deactivates the command. The indicator lamp in the button goes off when this command is taken into account (the corresponding command bit %Q is set to 0).

- For commands on a rising or falling edge :

The command is activated as soon as the button is pressed then released, and the indicator lamp in the button switches on then off automatically.

The indicator lamp beside the button shows when the command is taken into account by the module.

#### Entry field

Any value entered in an entry field must be confirmed with the  key.

#### Keyboard

**Shift F2** : used for moving from one zone to another.

**Tab** : used for moving from one set of commands to another in the same zone.


**Arrow keys** : used for moving from one command to another in the same set of commands.


**Space key** : used for activating or deactivating a command.

#### Warning

There may be "conflicts" between the PL7 program which executes the commands or writes the variables and the commands executed from the debug screen. The last command taken into account takes priority.

**Note**

The **Services/stop animation** command or the  icon stops the animation in the display zones and inhibits the control buttons.

The **Services/animate** command or the  icon reactivates the animation.

**5.2-3 Description of the debug screens**

The debug screens have a common header, consisting of module and channel zones.




**Module zone**

Indicator lamps	Status	Meaning
<b>RUN</b>	On	Module operating
<b>ERR</b>	On Flashing	Module off Communication fault
<b>I/O</b>	On	External hardware fault (encoder, speed drive, outputs)
<b>DIAG</b>	On	Fault on the module. Pressing this button (a button is associated with the indicator lamp) displays a module diagnostics dialog box which gives the source of the fault (see sect. 6.3 Diagnostics).

## Channelzone

In addition to the **Select axis** and **Function** fields (common to all screens), this zone contains the following commands :

Command		Role
		<p>Operating mode selection button. To access another mode click on the text of the mode you wish to select (or click as many times as necessary on the button). Using the keyboard : select the button using the <b>Tab</b> key and press as many times as necessary on the <b>space</b> bar.</p> <p>Modes can also be accessed from the <b>View</b> menu.</p> <p>When the selected mode is accepted by the module, the movement control zone in the required mode is displayed.</p> <p><b>Warning</b> : although it has been selected, the chosen mode may not be taken into account by the module channel (for example if the PLC is stopped).</p>
<b>CH3</b>	On Flashing Off	Channel 3 configured and not faulty Fault on channel 3 Channel 3 not configured
<b>DIAG</b>	On	Channel fault. Pressing this button (a button is associated with the indicator lamp) displays a channel diagnostics dialog box which gives the source of the fault (see Diagnostics).

### 5.2-4 Debugging the interpolation of channels X, Y and Z

Debugging the linear interpolation function consists of two screens : an automatic mode screen and an OFF mode screen.

It is possible to change from one mode to the other using a 2-position switch.

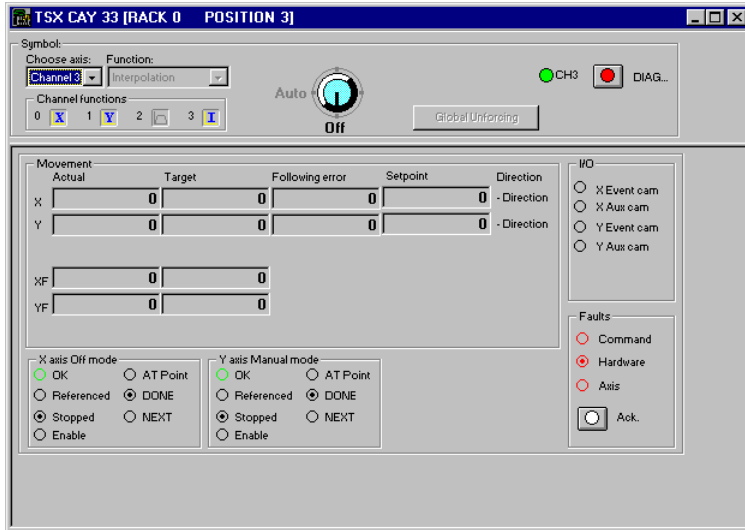
OFF mode displays the state of the various channels (either 2 or 3) of the group, without being able to send commands.

Automatic mode is used to :

- display the state of the module axes
- send commands to the interpolator

### 5.2-5 OFFmode

In this mode, the interpolation channel is used to display the 2 or 3 interpolated axes. This mode supervises the module axes.



#### Description of the information displayed

Movement : Current / Target

<b>Current X, Y, Z</b>	Displays the position of the moving part in the unit of measurement defined at configuration
<b>Target X, Y, Z</b>	Displays the target position for the moving part to reach
<b>X, Y, Z deviation</b>	Displays the difference between the calculated position and the actual position of the moving part
<b>X, Y, Z setpoint</b>	Displays the position to be reached
<b>+ direction</b>	Indicates a movement of the moving part in a positive direction
<b>- direction</b>	Indicates a movement of the moving part in a negative direction

Axes X, Y, Z

Indicator lamps	Status	Meaning
<b>OK</b>	on	Axis in operating state (no blocking fault)
<b>Referenced</b>	on	Axis referenced
<b>Stopped</b>	on	Moving part stopped
<b>Enable</b>	on	Speed drive enable relay active

Indicator	Status	Meaning
AT Point	on	Indicates that the current movement has been completed and that the moving part is in the target window
DONE	on	Indicates that the current movement has been completed
NEXT	on	Indicates that the next movement can be sent

Note : The Axis zone also contains the **Enable** command which is used to control the speed drive enable relay.

I/O

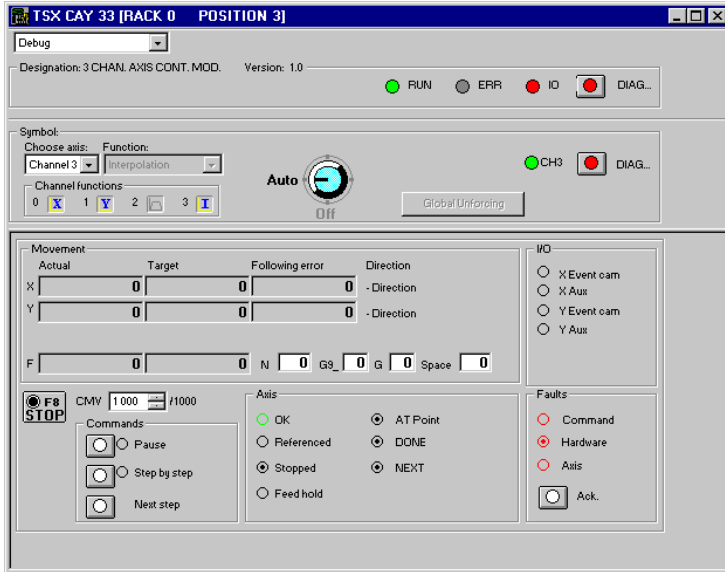
Indicator lamps	Meaning
EVT cam X,Y, Z	State of the signal (0 or 1) at the Event input
Aux cam X, Y, Z	State of the signal (0 or 1) at the auxiliary output

Faults

Indic. lamps Buttons	Status	Meaning
Cmd failure	on	Movement command failure (application)
Hardware	on	External hardware fault (encoder, speed drive, outputs, etc)
Axis	on	Application fault (deviation, soft stops, etc)
Ack		Fault acknowledgment button (all faults which have disappeared are acknowledged).

### 5.2-6 Automatic mode (Auto)

The debug screen displayed depends on the number of interpolated axes : there is a 2-axis debug screen and a 3-axis debug screen.



### Description of the commands and information displayed

Movement :

<b>Current X, Y, Z</b>	Displays the value of the current measurement of axis X, Y or Z
<b>Target X, Y, Z</b>	Displays the value of the position setpoint of axis X, Y or Z
<b>Deviation X, Y, Z</b>	Displays the difference between the calculated position setpoint and the actual position of the moving part (deviation) for axis X, Y or Z
<b>N</b>	Field containing the step number of the instruction being executed
<b>G9</b>	Field containing the type of movement of the instruction being executed
<b>G</b>	Field containing the code of the instruction being executed
<b>Space</b>	Field containing the space in which the current movement is executed (1)
<b>+ direction</b>	Indicates a movement of the moving part in a positive direction
<b>- direction</b>	Indicates a movement of the moving part in a negative direction
<b>CMV</b>	Velocity correction factor (0.001 to 2.000 in steps of 1/1000) for all the interpolated axes
<b>Stop F8</b>	Button for stopping all the interpolated axes

(1) 0 : (X,Y) ; 1 : (X,Z) ; 2 : (Y,Z) ; 3 : (X,Y,Z)

<b>Axis indicators</b>	<b>Status</b>	<b>Meaning</b>
<b>OK</b>	Activated	Indicates that the axes are operating
<b>Referenced</b>	Activated	Indicates that all the axes are referenced
<b>Stopped</b>	Activated	Indicates that the axes are stopped (no movement)
<b>Enable</b>	Activated	Indicates that all the axes have the speed drive enable signal
<b>AT Point</b>	Activated	Indicates that the current movement has been completed, and that the moving part is in the target window (for instructions with stop)
<b>DONE</b>	Activated	Indicates that the current movement(s) have been completed
<b>NEXT</b>	Activated	Indicates that the module is ready to receive a movement command

I/O

<b>Indicator lamps</b>	<b>Meaning</b>
<b>EVt cam X, Y or Z</b>	State of the signal (0 or 1) at the Event input of axes X, Y or Z
<b>Aux X, Y or Z</b>	State of the signal (0 or 1) at the auxiliary output of axes X, Y or Z

<b>Faults</b>	<b>Meaning</b>
<b>Failure</b>	The last XMOVE movement received has failed
<b>Hardware</b>	One of the interpolated axes has an external fault
<b>Axis</b>	One of the interpolated axes has a process fault

The **Ack** command button is used to acknowledge faults displayed in the fault zone.

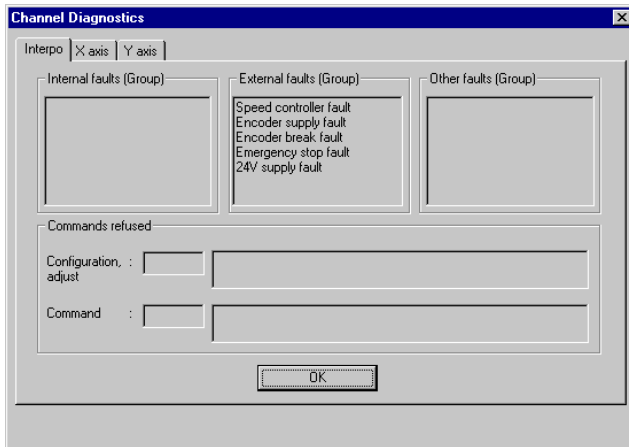


### 5.3 Diagnostics

In online mode the Debug, Adjustment and Configuration screens display the **DIAG** button, which gives details of faults detected by the module.

The channel 3 diagnostics window has 4 fault display tabs :

- Interpolation channel which shows all the faults for all of the interpolated axes
- Axis X which displays faults on channel 0
- Axis Y which displays faults on channel 1
- Axis Z which displays faults on channel 2



- **Internal (Group) faults** : internal module fault which generally requires the replacement of the module.
- **External (Group) faults** : fault originating in the operative part
- **Other (Group) faults** : application fault
- **Command failures** : gives, in the field concerned, the cause of the command failure and the message number.



## 6.1 Programming the XMOVE interpolation function

**XMOVE %CHxy.3 (N\_Run, G9\_, G\_-, Space, X, Y, Z, F, M)**

**%CHxy.3** = address of the TSX CAY 33 module in the PLC configuration

x = rack number

y = position of the module in the rack

3 = channel number of the TSX CAY 33 module

**N\_Run** = movement number (0 to 32767). Number identifying the movement executed by the XMOVE function. In debug mode it identifies the current movement.

**G9\_** = type of movement

**90** move to an absolute position value

**91** move to a relative value with respect to the current position

**98** move to a relative value with respect to the memorized position

**G\_ -** = instruction code

**09** move to a position and stop

**01** move to a position without stopping

**10** move until an event is detected and stop

**05** await an event

**92** load the PRF1 registers of axes X, Y, Z

**Space** = number of the plane or space in which the movement is to be performed. Lists the axes involved in the movement.

**0** movement in the XY plane

**1** movement in the XZ plane

**2** movement in the YZ plane

**3** movement in the XYZ space

**X, Y, Z** = coordinates of the position to be reached for channels 0,1 and 2, or position to which the moving part is to move.

The unit in which these values are expressed is defined in the **Length Units** configuration parameter of each of the axes.

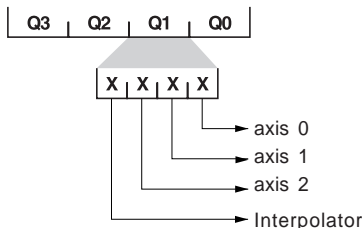
**F** = speed of movement of the moving part

**M** = word coded on two of the four 4-bit bytes 

Q3	Q2	Q1	Q0
----	----	----	----

Q3 : optional activation of the event processing associated with channel 3 for instructions 05 and 10

Q1 : a list of the axes involved in the instruction



Q2, Q0 : not used

## 6.2 Interpolator data

---

### Internal status data (implicit exchange)

%lxy.3.ERR	<b>ERROR</b>	Channel fault (Standard )
%lxy.3.0	<b>NEXT</b>	Ready to receive a new command
%lxy.3.1	<b>DONE</b>	All the instructions have been executed
%lxy.3.2	<b>AX_FLT</b>	Fault present on one of the axes
%lxy.3.3	<b>AX_OK</b>	Axes have no blocking faults
%lxy.3.4	<b>HD_ERR</b>	Hardware fault
%lxy.3.5	<b>AX_ERR</b>	Process fault
%lxy.3.6	<b>CMD_NOK</b>	Command failure
%lxy.3.8	<b>NOMOTION</b>	Moving part stopped
%lxy.3.9	<b>AT_PNT</b>	Moving part on target
%lxy.3.10	<b>TH_PNT</b>	Theoretical setpoint reached
%lxy.3.12	<b>CONF_OK</b>	Axes configured
%lxy.3.14	<b>REF_OK</b>	Axes referenced
%lxy.3.20	<b>IN_DROFF</b>	OFF mode selected
%lxy.3.23	<b>IN_AUTO</b>	AUTO mode selected
%lxy.3.33	<b>ON_PAUSE</b>	Movement sequencing suspended
%lxy.3.34	<b>IM_PAUSE</b>	Movement suspended (immediate PAUSE)
%lxy.3.39	<b>IN_STEP</b>	Step-by-step mode active
%lDxy.3.2	<b>X_SPEED</b>	Measured speed
%lDxy.3.4	<b>FOL_ERR</b>	Current position error in the space
%lWxy.3.7	<b>SYNC_N_RUN</b>	Current step number

### Internal command data (implicit exchange)

%Qxy.3.8	<b>ACQ_DEF</b>	Axis fault acknowledgment
%Qxy.3.10	<b>EXT_EVT</b>	Application program external event
%Qxy.3.15	<b>STOP</b>	STOP command
%Qxy.3.16	<b>PAUSE</b>	Pause command (end of current block)
%Qxy.3.19	<b>MODE_STEP</b>	Step-by-step selection mode
%Qxy.3.22	<b>NEXT_STEP</b>	Next step command
%Qxy.3.23	<b>PCQ23_FCTSPE</b>	<i>Reserved use</i>
%QWxy.3	<b>MOD_SELEC</b>	Mode selector
%QWxy.3.1	<b>CMV</b>	Velocity correction factor

---

### 6.3 Interpolation adjustment parameters

---

%MWxy.3.23	<b>SLOPE</b>	Acceleration profile
%MWxy.3.24	<b>TACC</b>	Acceleration time
%MWxy.3.25	<b>DELTASPEEDPATH_X</b>	Permitted speed threshold on X axis (as a % of VMAX)
%MWxy.3.26	<b>DELTASPEEDPATH_Y</b>	Permitted speed threshold on Y axis (as a % of VMAX)
%MWxy.3.27	<b>DELTASPEEDPATH_Z</b>	Permitted speed threshold on Z axis (as a % of VMAX)

## 6.4 Internal status data

### %MWxy.3.0 Exchange management (standard)

Bit X0	STATUS parameter exchange in progress
Bit X1	COMMAND parameter exchange in progress
Bit X2	ADJUSTMENT parameter exchange in progress
Bit X15	Reconfiguration in progress

### %MWxy.3.1 Exchange report (standard)

Bit X1	COMMAND exchange report
Bit X2	ADJUSTMENT exchange report
Bit X15	Configuration OK

### %MWxy.3.2 Channel status (standard)

Bit X0	External fault
Bit X1	External fault
Bit X2	Terminal block fault
Bit X4	Internal fault (Module missing, off or performing self-tests)
Bit X5	Hardware or software configuration fault (module reference <> physical module)
Bit X6	Communication fault (no communication with the processor)
Bit X7	Application fault (incorrect configuration, adjustment or command failure)
Bit X8	Channel indicator lamp
Bit X9	Channel indicator lamp

### %MWxy.3.3 Interpolation status (blocking faults)

Bit X0	<b>ANA_FLT</b>	Analog output short-circuit
Bit X1	<b>AUX_FLT</b>	Auxiliary output short-circuit
Bit X2	<b>DRV_FLT</b>	Speed drive fault
Bit X3	<b>ENC_SUP</b>	Encoder supply fault
Bit X4	<b>ENC_BRK</b>	Encoder wiring break fault
Bit X5	<b>EMG_STP</b>	Emergency stop fault
Bit X6	<b>AUX_SUP</b>	24 V supply fault
Bit X7	<b>ENC_FLT</b>	Serial absolute encoder parity fault
Bit X8	<b>SLMAX</b>	Upper soft stop fault
Bit X9	<b>SLMIN</b>	Lower soft stop fault
Bit X10	<b>SPD_FLT</b>	Overspeed fault
Bit X11	<b>FE1_FLT</b>	DMAX1 position error fault

### %MWxy.3.3 Interpolation status (non-blocking faults)

BitX12	<b>REC_FLT</b>	Recalibration fault
Bit X13	<b>TW_FLT</b>	Target window fault
Bit X14	<b>STP_FLT</b>	STOP fault
BitX15	<b>FE2_FLT</b>	DMAX2 position error fault

---

<b>%MWxy.3.4 N_RUN</b>	Current step number
<b>%MWxy.3.5 G9_COD</b>	Type of movement in progress (G9_)
<b>%MWxy.3.6 G_COD</b>	Code of the current instruction (G)
<b>%MWxy.3.7 CMD_FLT</b>	Failure report
<b>%MWxy.3.12G_SPACE</b>	List of the axes of the current XMOVE : 0 = X and Y 1 = X and Z 2 = Y and Z 3 = X, Y and Z
<b>%MDxy.3.13 T_XPOS</b>	Target position to be reached on axis X
<b>%MDxy.3.15 T_YPOS</b>	Target position to be reached on axis Y
<b>%MDxy.3.17 T_ZPOS</b>	Target position to be reached on axis Z
<b>%MDxy.3.19 T_SPEED</b>	Target speed





## 7.1 List of *CMD\_FLT* error codes

The following pages give a list of the messages explaining the *CMD\_FLT* (%MWxy.i.7) command failure word.

This word is read by explicit exchange.

Messages also appear in plain text in the Diagnostics dialog boxes which can be accessed using the **DIAG** button (see section 5.3 part B2).

The *CMD\_FLT* word is composed of two bytes which each correspond to a specific class of error.

%MWxy.i.7

Configuration and adjustment parameter	Movement command
---	------------------

High order byte

Low order byte

High order byte : error in the configuration and adjustment parameters (XX00)

Low order byte : failure to execute the movement command (00XX) :

Example : 0023

└── Low order byte : Stack full

---

## List of error codes linked to the interpolation channel

The values given are the hexadecimal code values :

- 0012** = the command cannot be executed for one of the following reasons :
  - there is another command which has not yet been completed
  - the channel is no longer in AUTO mode
  - there is currently a stop on the channel
  - the channel relay is open (position control only)
- 0013** = command G01 cannot be executed
- 0014** = command G09 cannot be executed
- 0015** = command G10 cannot be executed
- 001B** = command G07 cannot be executed (position control only)
- 001D** = code G\_ of the command is not recognized
- 0023** = since the stack is full, no further code G\_ can be memorized
- 0060** = code G\_ is not permitted after a code G01
- 0061** = code G01 is not executed unless it is followed by a movement code
- 0063** = the execution conditions for the interpolated movement have not been observed on axis X
- 0064** = the execution conditions for the interpolated movement have not been observed on axis Y
- 0065** = the execution conditions for the interpolated movement have not been observed on axis Z
- 0066** = a group action is requested on axis Z, but the axis is not part of the group
- 0067** = an axis stops during interpolation (change of mode, opening of the relay, etc)
- 0068** = the requested target positions of code G\_ are outside the soft stop limits
- 0069** = code G01 has failed because the next movement cannot be accepted
- 0040** = the distance for executing this movement is zero
- 0050** = the distance for executing this movement is zero
- 0080** = a reversal is required for this G9/G10 instruction
- 0081** = the distance of this G1 instruction is too short
- 0082** = the distance of the movement which follows this G1 instruction is too short
- 0083** = the current speed is too high and/or the distance of this G1 instruction is too short to reach Vthresh.

### Adjustment parameters command failure

- 0092** = acceleration profile failed
- 0093** = speed diff. X incompatible
- 0094** = speed diff. Y incompatible
- 0095** = speed diff. Z incompatible

**Symbols**

%CHxy.3 B2 3/2  
 %CHxy.i B1 10/1  
 %lxy.MOD.ERR B1 10/3  
 %MWxy.MOD B1 10/3  
 + direction B1 6/8, B1 6/11, B2 5/5, B2 5/7  
 - direction B1 6/8, B1 6/11, B2 5/5, B2 5/7  
 01 B1 4/10, B2 3/6  
 05 B1 4/15, B2 3/8  
 07 B1 4/16  
 09 B1 4/10, B2 3/6  
 10 B1 4/11, B2 3/7  
 11 B1 4/11  
 14 B1 4/13, B1 4/14  
 2 axes B2 4/2  
 24V supply B1 4/37  
 3 axes B2 4/2  
 62 B1 4/15  
 90 B1 4/7, B2 3/5  
 91 B1 4/7, B2 3/5  
 98 B1 4/7, B2 3/5

**A**

Absolute encoder B0 2/2, B1 3/6  
 Absolute encoder frame B1 4/37, B2 3/5  
 Absolute movement B1 4/7, B2 3/5  
 Acceleration B1 5/13  
 Acceleration profile B1 5/14  
 Adjustment parameters B15/4, B111/3, B24/1  
 Analog output short-circuit B1 4/36  
 Animate B1 6/3, B2 5/3  
 AT\_PNT B1 4/19, B1 6/8, B1 6/11,  
 B2 3/12, B2 5/6, B2 5/8  
 AUTO B1 4/1, B1 6/10, B2 5/7  
 Automatic B1 4/1  
 Automatic mode B1 6/10, B2 5/7  
 Aux B1 6/8, B1 6/11, B2 5/6  
 Auxiliary output short-circuit B1 4/37  
 Await an event B1 4/15  
 Axis operating status B1 10/5

**B**

Blocking faults B1 4/33, B2 3/17

**C**

Changing the configuration B1 4/32  
 Changing to STOP B1 4/32  
 Channel faults B1 4/35  
 Channel operating status B1 10/5  
 Clear reference B1 4/47  
 CMD\_FAIL B1 11/1, B2 7/1  
 CMD\_FLT B1 11/1, B2 7/1  
 CMV B1 5/9, B1 6/8, B1 6/11  
 Command B1 6/6  
 Command failure B1 4/41, B1 6/13, B1 11/1,  
 B23/19, B25/9  
 Command failure report B1 10/5  
 Configuration B1 11/2  
 Configuration parameters B1 3/18, B1 5/2,  
 B2 2/2, B2 4/1  
 Confirm adjustment parameters B1 5/18  
 Control parameters B1 5/12  
 Critical faults B1 4/33, B2 3/17  
 Current instruction code B1 10/5  
 Current limit as a function of the speed B1 5/1  
 Current loop B1 5/1  
 Current parameters B1 5/4, B2 4/1  
 Current position error B1 10/4  
 Current step number B1 10/4, B1 10/5

**D**

Debug screen B1 6/2, B2 5/2  
 Debugging B1 6/1, B2 5/2  
 Deferred "PAUSE" B1 4/27  
 Delay B1 5/15  
 Deviation B1 4/38  
 Deviation 1 B1 5/12  
 Deviation 2 B1 5/12  
 DIAG B1 5/15, B1 6/13, B2 5/9  
 Diagnostics B1 6/13, B2 5/9  
 Dim : B2 3/3  
 DIRDRIVE B1 4/1, B1 4/49, B1 6/6  
 Direct drive B1 4/1  
 Direct drive mode B1 4/48, B1 6/6  
 Documentation B1 6/14  
 DONE B1 4/19, B1 6/8, B1 6/11, B2 5/6

**E**

Emergency stop	B1 4/36
Encoder offset	B1 5/5
Encoder supply	B1 4/37
Encoder wiring break	B1 4/36
End G05	B1 6/12
End G10	B1 6/12
End G11	B1 6/12
Error codes	B1 11/1, B2 7/1
Event processing time	B0 2/1
Evt cam	B1 6/8, B1 6/11, B2 5/6, B2 5/8
EVT_G05	B1 4/31, B2 3/15
EVT_G07	B1 4/31
EVT_G1	B1 4/31, B2 3/15
Exchange management	B1 10/5
Exchange report	B1 10/5
Execution time	B0 2/1
Explicit exchanges	B1 10/5
External command	B1 6/12
External fault	B1 6/13, B2 5/9
External hardware faults	B1 4/36

**F**

F	B1 6/6
F Current	B1 6/7, B1 6/10
F Target	B1 6/7, B1 6/10
Fault acknowledgment	B1 4/34, B2 3/18
Fault indication	B1 4/34, B2 3/18
Fault monitoring	B1 8/1
Faults	B1 4/33
Feedforward	B1 5/8, B1 5/10
Follower movement of a periodic setpoint	B1 4/26
Follower movement of another axis	B1 4/22
Forced reference point	B1 4/46, B1 6/9
Forcing to 0	B1 6/4
Forcing to 1	B1 6/4

**G**

G	B1 6/10
G9	B1 6/10
Gain 1	B1 5/8
Gain 2	B1 5/8
Gain at high speed	B1 5/9
Gain at low speed	B1 5/10
Gain threshold	B1 5/8, B1 5/10
Global unforcing	B1 6/4

**H**

Hardware configuration	B1 3/1
------------------------	--------

**I**

Immediate "PAUSE"	B1 4/30, B1 6/11
Implicit exchanges	B1 10/3
INC+	B1 6/9
INC-	B1 6/9
INC_M	B1 4/45
INC_P	B1 4/45
Incremental movement	B1 4/45
Indicator lamps	B1 6/3, B2 5/3
Initial parameters	B1 5/4, B2 4/1
Inputs	B0 1/4
Instruction code	B1 4/4, B2 3/3
Internal fault	B1 6/13, B2 5/9
Interpolated axes	B2 1/2
Interpolated axis faults	B2 3/16
Interpolator status data	B2 6/2
Inversion	B1 5/2

**J**

JOG+	B1 6/9
JOG-	B1 6/9
JOG_M	B1 4/44, B1 6/9
JOG_P	B1 4/44, B1 6/9

**L**

Lower limit	B1 3/9
Lower soft stop	B1 5/13

**M**

MAN	B1 4/1, B1 6/7
Man-machine interface	B1 7/1
Manual	B1 4/1
Manual commands	B1 4/44
Manual mode	B1 4/43, B1 5/16, B1 6/7
Masking of events	B1 4/31
Maximum acceleration	B1 3/12
Maximum deceleration	B1 3/12
Maximum position error	B1 10/5
Maximum setpoint	B1 3/10
Measured position	B1 10/4
Measured speed	B1 10/4
Measurement	B1 4/1

Measurement mode B1 4/49, B1 6/5,  
B2 3/20, B2 5/5  
Measurement units B1 3/8  
Memorize current position  
when an event occurs B1 4/16  
Memory consumption B02/1  
Methodology B1 2/1, B2 1/17  
Mode selector B1 10/3  
Module status word B1 10/3  
Motion control B0 1/5  
Move to a position B1 4/10, B2 3/5  
Move to a position and stop B1 4/10  
Move to a position without stopping B1 4/10  
Move until an event is detected B1 4/7,  
B2 3/3, B2 3/7  
Movement codes B1 4/4  
Movement controls B1 4/40, B1 5/12

## N

N B16/10  
NEXT B1 4/19, B1 6/11, B2 3/12  
Next step B16/12  
Non critical faults B1 4/33, B2 3/17

## O

Observed distance B15/6  
OFF B1 4/1, B1 4/49, B1 6/5,  
B2 3/20, B2 5/5  
Offset B1 5/9, B1 5/10  
Operating modes B1 4/32  
Outputs B0 1/4  
Overspeed B1 4/38, B1 5/12

## P

Param B1 6/6, B1 6/8, B1 6/11  
Parameter configuration B1 3/4  
Pause B1 6/12  
PLC in RUN B1 4/32  
Position B1 6/6, B1 6/7, B1 6/10  
Position control loop B1 5/7  
Power outage and return B1 4/32  
PRef B1 6/12  
PRef1 B1 6/12  
PRef2 B1 6/12  
Presymbolization B0 1/12

## R

READ\_PARAM B1 10/7  
READ\_STS B1 10/7  
Recalibration B1 4/21, B1 4/39,  
B1 6/8, B1 6/11  
Recalibration deviation B1 5/12  
Recalibration position B1 5/12  
Reconfiguration in online mode B1 5/20  
Reference point B1 4/7, B1 4/13,  
B1 4/14, B1 4/46  
Referencing and offset calculation B1 4/47  
Reflex inputs B1 3/14  
Relative movement B1 4/7, B2 3/5  
Repetitive movements B1 4/17  
Resolution B1 3/8, B1 5/6  
RESTORE\_PARAM B1 10/7  
RP cam B1 6/8, B1 6/11  
RP\_HERE B1 4/46, B1 4/47

## S

Save adjustment parameters B1 5/18  
Save application B1 6/14  
SAVE\_PARAM B1 10/7  
Sequence check B1 3/12, B1 8/1  
Servo parameters B1 5/7, B1 5/8  
SET\_RP B1 4/46, B1 4/47  
Setpoint B1 6/6, B1 6/7  
Short cam B1 3/15  
Slave B1 6/12  
SMOVE B1 4/3, B1 10/1  
SMOVE function B1 4/3, B2 3/2  
Soft stops B1 4/38  
Software setup B0 1/7  
Special functions B0 1/5, B2 2/2  
Speed B1 5/16, B1 6/6, B1 6/7, B1 6/10  
Speed correction B1 10/3  
Speed drive B1 4/36, B1 5/1  
Speed loop B1 5/1  
Step by step B1 6/12  
Step by step mode B1 4/28  
STOP B1 6/6, B1 6/8, B1 6/11  
Stop B1 4/39  
Stop animation B1 6/3, B2 5/3  
Stop axes on fault B2 2/2  
Stop controls B1 5/15  
Stop speed B1 5/15

---

Switching on	B1 4/32
Synchro CPU	B1 6/12

**T**

Target speed	B1 10/5
Target window	B1 4/39, B1 5/15
Teaching the positions	B1 9/1
TH_PNT	B1 4/19, B1 6/8, B1 6/11, B2 3/12
Theoretical distance	B1 5/6
TO_G05	B1 4/31, B1 6/12, B2 3/15
Trajectory	B1 4/18, B2 3/9
TRL	B0 2/1
TSTOP	B1 4/19, B2 3/11
Type of movement	B1 4/4

**U**

Unforcing	B1 6/4
Upper limit	B1 3/9
Upper soft stop	B1 5/13

**V**

Value of register PREF1	B1 10/4
Value of register PREF2	B1 10/4
Value of the movement increment	B1 10/3
Visual movement	B1 4/44
VLIM	B1 5/12

**W**

WRITE_PARAM	B1 10/7
-------------	---------

**X**

X	B1 6/6
X Current	B1 6/7, B1 6/10, B2 5/7
X Deviation	B1 6/7, B1 6/10, B2 5/7
X Target	B1 6/7, B1 6/10, B2 5/7
XMOVE	B2 3/1
XMOVE function details	B2 3/2
XMOVE interpolation	B2 6/1
XMOVE movements	B2 3/12
X,Y,Z	B2 3/4

<b>Section</b>	<b>Page</b>
<b>1 Introduction to axis control</b>	<b>1/1</b>
1.1 Introduction	1/1
1.2 Functions	1/2
1.3 Stepper motor axis control	1/4
1.3-1 The control part	1/4
1.3-2 The translator	1/4
1.3-3 Stepper motors	1/4
1.3-4 SS_FREQ start/stop frequency	1/5
1.3-5 Boost	1/5
1.3-6 Brake output	1/5
1.4 Software setup	1/6
1.4-1 Programming movements	1/6
1.4-2 Configuring the axes	1/9
1.4-3 Adjusting the axes	1/10
1.4-4 Presymbolization	1/11
1.4-5 Debugging	1/12
1.4-6 Man-machine interface and control	1/13
<b>2 Tutorial</b>	<b>2/1</b>
2.1 Description of the example	2/1
2.2 Prerequisites	2/3
2.3 Application design	2/4
2.3-1 Software declaration of the PLC configuration used	2/4
2.3-2 Entering the configuration parameters for each axis	2/4
2.3-3 Entering the symbols for the application	2/7
2.3-4 Programming	2/9
2.3-5 Program transfer	2/13

<b>Section</b>	<b>Page</b>
2.4 Debugging	2/14
2.4-1 Preliminary measures	2/14
2.4-2 Using manual mode	2/14
2.4-3 Debugging	2/16
2.4-4 Archiving	2/16
<hr/> <b>3 Setup methodology</b>	<hr/> <b>3/1</b>
3.1 Setup methodology	3/1
<hr/> <b>4 Configuration</b>	<hr/> <b>4/1</b>
4.1 Configuring axis control modules	4/1
4.1-1 Introduction	4/1
4.1-2 The configuration editor	4/1
4.2 Declaring the axis control modules	4/2
4.3 Entering the configuration parameters	4/4
4.3-1 Access to the parameter configuration screen	4/4
4.3-2 User units	4/5
4.3-3 Control mode	4/6
4.3-4 Control parameters	4/7
4.3-5 Type of event edge	4/9
4.3-6 Translator reversal	4/10
4.3-7 Boost	4/11
4.3-8 Brake	4/11
4.3-9 Event	4/12
4.3-10 Reference point	4/12
4.4 Confirming the configuration parameters	4/17

---



<b>Section</b>	<b>Page</b>
<b>5 Programming</b>	<b>5/1</b>
5.1 Programming principle	5/1
5.2 Operating modes	5/1
5.3 Programming in automatic mode : SMOVE function	5/2
5.3-1 Programming an SMOVE function	5/2
5.3-2 Entering the parameters of the SMOVE function	5/3
5.3-3 Description of elementary movements	5/6
5.3-4 Description of instructions	5/7
5.3-5 Sequence of movement commands	5/13
5.4 Programming in automatic mode : other functions	5/16
5.4-1 Deferred "PAUSE" function	5/16
5.4-2 Immediate "PAUSE" function	5/17
5.4-3 Event processing	5/18
5.5 Managing the operating modes	5/19
5.6 Fault management	5/20
5.6-1 Role	5/20
5.6-2 Principle	5/20
5.6-3 Programming	5/21
5.6-4 Summary table	5/22
5.6-5 Description of module faults	5/22
5.6-6 Description of external hardware faults	5/22
5.6-7 Description of application faults	5/24
5.6-8 Description of command failure faults	5/25
5.7 Management of manual mode (MAN)	5/26
5.7-1 Selecting manual mode	5/26
5.7-2 Execution of manual commands	5/26
5.7-3 Detailed description of manual commands	5/27

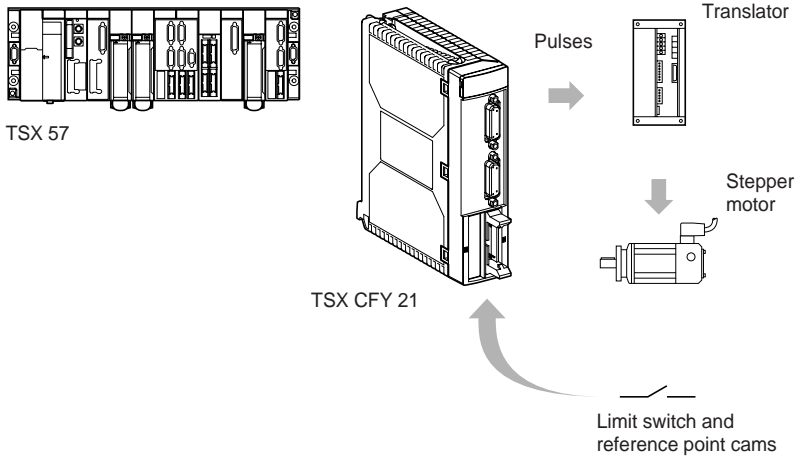
<b>Section</b>	<b>Page</b>
5.8 Managing direct drive mode (DIRDRIVE)	5/30
5.8-1 Selecting direct drive	5/30
5.8-2 Executing commands in direct drive mode	5/30
5.9 Management of stop mode (OFF)	5/31
5.10 Presymbolization	5/32
5.11 Transferring the program to the PLC	5/32
<b>6 Adjusting the axes</b>	<b>6/1</b>
6.1 Operations prior to adjustment	6/1
6.1-1 Preliminary conditions	6/1
6.1-2 Preliminary checks	6/1
6.1-3 Adjusting the translator	6/1
6.2 Adjusting the parameters	6/2
6.2-1 Access to the adjustment parameters	6/2
6.2-2 Trajectory	6/4
6.2-3 Brake output	6/5
6.2-4 Stop plateau	6/6
6.2-5 Manual mode parameters	6/7
6.3 Confirming and saving adjustment parameters	6/8
6.3.1 Confirming	6/8
6.3-2 Save	6/9
6.3-3 Restore	6/9
6.4 Reconfiguration in online mode	6/10

<b>Section</b>	<b>Page</b>
<b>7 Debugging an axis control program</b>	<b>7/1</b>
7.1 Principle of debugging an axis control program	7/1
7.2 Debug screens	7/2
7.2-1 Accessing the debug screens	7/2
7.2-2 User interface	7/2
7.2-3 Description of the debug screens	7/3
7.2-4 Stop mode (Off)	7/5
7.2-5 Direct mode (Dir Drive)	7/6
7.2-6 Manual mode (Man)	7/7
7.2-7 Automatic mode (Auto)	7/10
7.3 Diagnostics	7/13
7.4 Archiving	7/14
7.5 Documentation	7/14
7.6 Simulation	7/14
<b>8 Operation</b>	<b>8/1</b>
8.1 Designing a man-machine interface	8/1
8.1-1 Control station	8/1
8.1-2 Man-machine interface on CCX 17	8/1
<b>9 Diagnostics and maintenance</b>	<b>9/1</b>
9.1 Fault monitoring	9/1
9.2 Conditions for executing commands	9/1
9.3 Diagnostic help	9/2

<b>Section</b>	<b>Page</b>
<b>10 Performance and limitations</b>	<b>10/1</b>
10.1 Characteristics of the stepper motor control functions	10/1
10.2 Limitations of the TSX CFY module	10/2
10.2-1 Low amplitude movements	10/2
10.2-2 Maximum start/stop frequency	10/2
<b>11 Additional functions</b>	<b>11/1</b>
11.1 Teaching the positions	11/1
<b>12 Glossary</b>	<b>12/1</b>
12.1 Glossary	12/1
<b>13 Quick reference guide</b>	<b>13/1</b>
<b>14 List of CMD_FAIL error codes</b>	<b>14/1</b>
14.1 List of CMD_FLT error codes	14/1
<b>15 Index</b>	<b>15/1</b>

---

## 1.1 Introduction



The stepper motor axis control range for TSX 57 PLCs comprises 2 axis control modules :

- TSX CFY 11 : axis control module with 1 axis,
- TSX CFY 21 : axis control module with 2 independent axes.

PL7 software integrates stepper motor motion control functions as standard for programming these stepper motor axis control modules.

The TSX CFY 11/21 module manages limited and independent linear axes.

Elementary movements are controlled from the main sequential control program of the machine, but are performed and controlled by the TSX CFY 11/21 modules.

The TSX CFY 11/21 axis control module controls both the speed of rotation of a stepper motor and its acceleration and deceleration by sending a command expressed as a frequency to a translator ( $f_{max} = 187\text{KHz}$ ). The translator transforms each pulse into an elementary movement of the stepper motor.

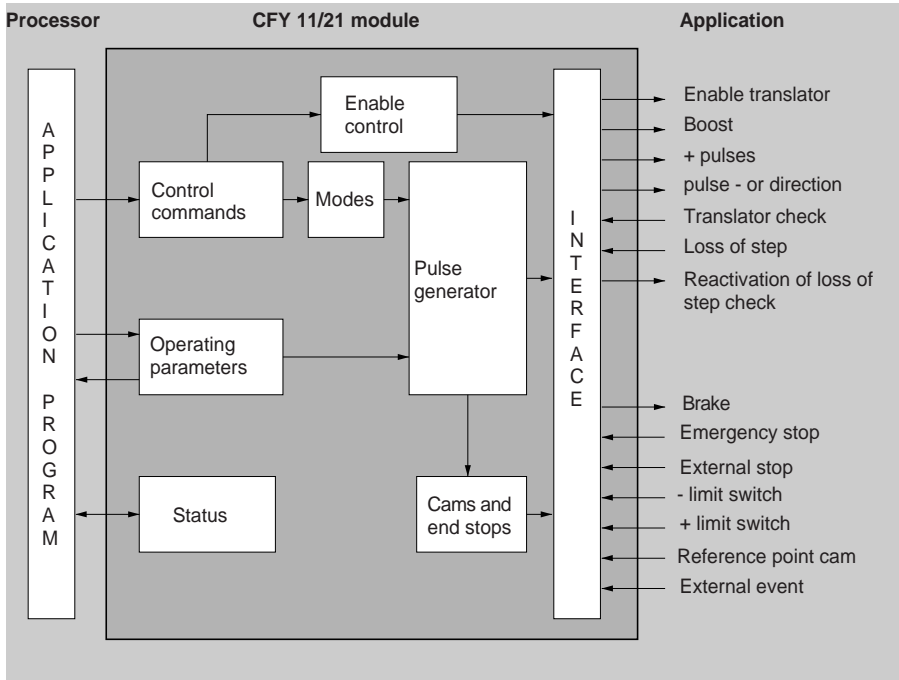
The stepper motor is controlled in open loop. Limit switch, reference point and event-triggered inputs enable the module to control the movements of the moving part on the axis.

Some translators have a built-in loss of step device : this information is available to the user program, which can then set a new reference point.

The stepper motor axis control range also includes the TSX CXP 611 cable for the direct connection of TSX CFY 11/21 modules to MSD and SP translators from Phytron Elektronik GmbH.

## 1.2 Functions

### Block diagram of a channel



### Application I/O

For each of the axes, stepper motor axis control modules provide :

For the auxiliary I/O :

- one reference point cam input,
- two limit switch inputs,
- one emergency stop input,
- one event input,
- one external stop input,
- one solid state output for the axis brake.

For the translator I/O :

- one translator check input,
- one input to check loss of step,
- one differential output to enable the translator,
- two differential pulse outputs : one positive and one negative,
- one differential output to boost the stepper motor,
- one differential output to reset the step loss control.

---

**Processing commands controlled from the PLC sequential program :**

Each movement is defined by an SMOVE motion control function in PL7 language. From this SMOVE command and the position of the moving part, the TSX CFY 11/21 module generates the position/speed setpoint and the movement pulses.

**Configuration and adjustment parameters :**

These parameters define the usage characteristics, limits, etc, of the axis.

**Special functions :**

- **Event-triggered processing** : events detected by the module can be used to activate an event-triggered task in the sequential program.
- **Boost command**: this function is used to boost the stepper motor during the acceleration and deceleration stages.
- **Brake command**: this function is used to control the stepper motor brake on start-up and on stopping.
- **Immediate pause** : this function is used to momentarily stop a movement which is in progress.
- **Deferred pause** : this function is used to momentarily stop a machine cycle without disturbing it.
- **Limit switches** : crossing these limits stops the movement. After a limit has been crossed, only movements to return within the limits are accepted.
- **External stop** : activation of the external stop input stops the movement.
- **Loss of step input and reset loss of step check output** : these functions are used by the application program to manage loss of step data from the translator. For the module, activation of the loss of step input does not constitute a stop condition, nor a fault condition.

---

## 1.3 Stepper motor axis control

---

The operational breakdown of a stepper motor axis control sequence generally includes three elements :

- a stepper motor,
- a translator,
- a control part.



---

### 1.3-1 The control part

In this diagram, the command function is carried by a channel of the TSX CFY 11/21 module. The main function of this channel is to supply a frequency pulse train controlled at every moment, in order to execute the required movements.

---

### 1.3-2 The translator

The main function of the translator is to transform each pulse received into a motor step (elementary rotation), by circulating the appropriate currents in the motor coils.

---

### 1.3-3 Stepper motors

Stepper motors are constructed using various technologies, for example permanent magnet motors, variable reluctance motors, and motors which include elements of both these techniques. Moreover, various coil solutions are available on the market : there are two, four and five phase motors.

In addition, stepper motors are associated with translators which have been designed and optimized for their particular architecture.



---

### 1.3-4 SS\_FREQ start/stop frequency

Control of the various stepper systems should generally conform to a common constraint, due to the reaction of the inertial system (motor + axis) to a pulsed command. This common constraint is the starting and stopping frequency.

The starting and stopping frequency is the frequency at which the motor can start and stop without ramp and without loss of step. Its maximum limit value depends on the external inertia on the motor axis. Its average value is 400Hz in 1/2 steps (1 revolution/s) and can be critical beyond 600/800Hz (1.5 to 2 revolution/s) (typical values for Phytron Elektronik 200 step/revolution translators/motors).

This constraint exists at both the stop and start of each movement, which is why it is called start/stop frequency : SS\_FREQ (Start/Stop Frequency). In the TSX CFY 11/21 module, this data can be adjusted.

Note: in this manual, the terms **frequency** and **speed** are used interchangeably. The **Hertz** and **pulses/s** speed units and the **Hertz/s** and **pulses/s<sup>2</sup>** acceleration units are also interchangeable.

---

### 1.3-5 Boost

Some translators have a boost input. This function increases the current in the motor coils.

The boost output of one TSX CFY 11/21 module channel is used to control this translator input. The intensity of the motor current can thus be synchronized with the movement. In particular, this output can be controlled in automatic mode for activation during the acceleration and deceleration stages.

---

### 1.3-6 Brake output

This solid state output is used to control a brake on the axis, synchronized with the movement, or at the request of the user.

This function is useful in applications with a driving load, in which the motor power supply needs to be interrupted.

Note that when the channel is in the safety position, this output sets the brake to active state (the brake is generally on when the power is off).

## 1.4 Software setup

The PL7 setup software provides :

- for configuration, adjustment and moving axes, screens which are available in the configuration editor.
- for programming movements, an SMOVE movement control function, which can be used in Ladder language, Instruction List language or Structured Text language.

### 1.4-1 Programming movements

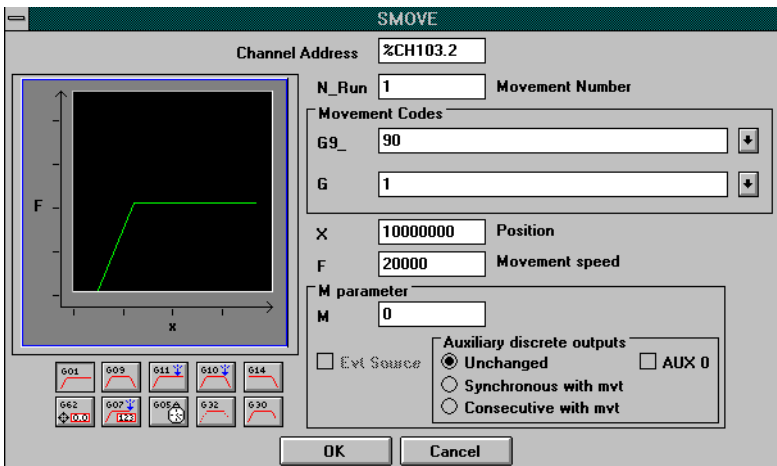
A movement is initiated by executing an SMOVE control function in the PL7 program.

#### Example 1 :

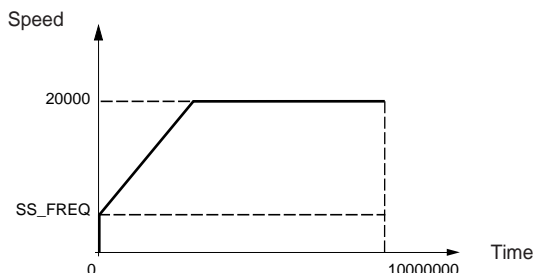
Go to the absolute position 10 000 000 pulses (increments), at a speed of 20 000 Hertz, without stopping.



A screen can be used for assisted entry of the parameters of the SMOVE function in the operation block (see section 5.3-1).



```
SMOVE%CH103.1 (1, 90, 1000000, 20000,0)
```



Significance of each parameter (see complete description in section 5) :

**SMOVE** : movement following an axis control function

**%CH103.1** : address of CFY module on the rack (channel 1, position 03, rack 1)

**01** : movement number 1

**90** : move to an absolute position

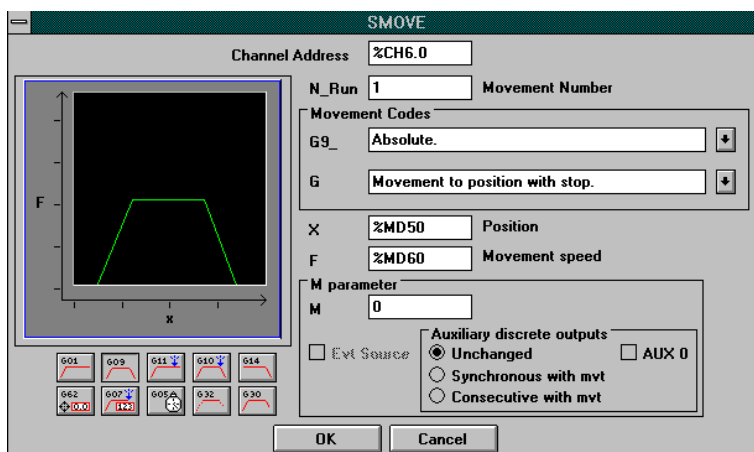
**01** : instruction code corresponding to movement to a position without stopping.

**10 000 000** : position to be reached by the moving part in number of pulses.

**20 000** : speed of the moving part in Hertz.

**0** : event not enabled.

**Example 2** : SMOVE %CH6.0 (1,90,09,%MD50,%MD60,16#0000)



In this example, the position to be reached is contained in double word %MD50 and the speed in double word %MD60. **These words can be symbolized and indexed.**

---

## Instruction codes

The characteristics of movements are described using a syntax similar to that for a numerical control program block written in ISO language.

TSX CFY 11/21 axis control provides the following instructions :

- 09** : move to the position and stop,
- 01** : move to the position without stopping,
- 10** : move until an event is detected and stop,
- 11** : move until an event is detected without stopping,
- 14** : reference point,
- 05** : await an event,
- 07** : memorize the current position when an event occurs,
- 62** : forced reference point.

These instructions can be represented as symbols by the user in G code (for example : 09 can be represented by G09).

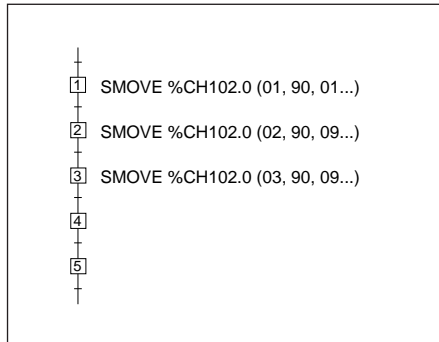
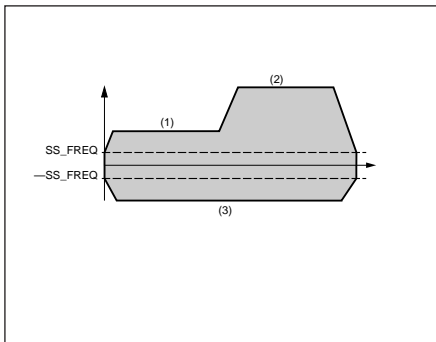
The instruction codes are preceded by another code if :

- 90** : the target position is absolute,
- 91** : the target position is relative with respect to the current position,
- 98** : the target position is relative with respect to a PREF memorized position (index).

## Programming a trajectory

A complete trajectory can be programmed by means of a series of SMOVE elementary motion control functions.

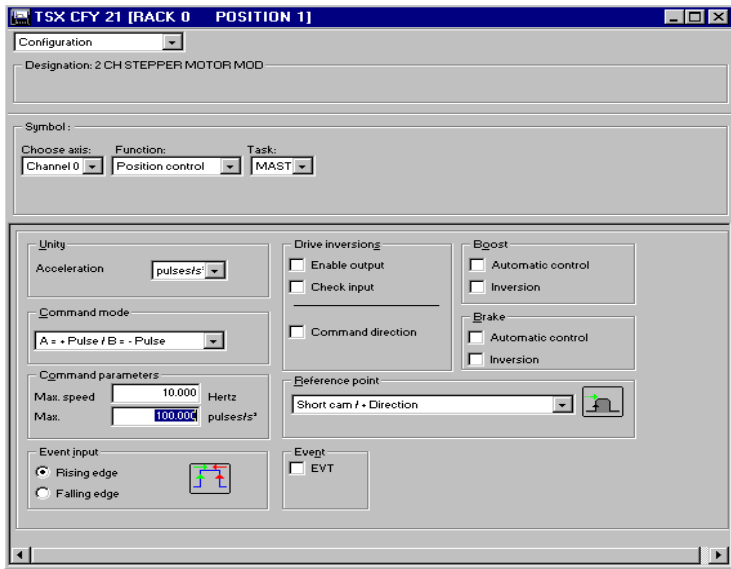
Grafcet language is ideal for this type of programming. An elementary movement is associated with each step.



## 1.4-2 Configuring the axes

The configuration editor provides assistance with entering and modifying the values of the various axis configuration parameters. These parameters enable the operation of the axis control module to be adapted to the machine which is to be controlled.

Axis configuration parameters : units of acceleration, command mode, translation reversal, boost, brake, etc are linked to the machine and cannot be modified by the program.



Configuration parameters must be entered (no default configuration).

### 1.4-3 Adjusting the axes

These parameters are linked to operation of the axes. They generally require the operations on and movements of the moving part to be known. These parameters are adjusted in online mode (they are initialized in offline mode).

Axis operating parameters :

- start/stop frequency, acceleration, etc,
- duration of stop plateau,
- manual mode parameters.

These parameters can be modified by the program.

The screenshot shows a software window titled "TSX CFY 21 [RACK 0 POSITION 1]". The window contains several sections for configuring axis parameters:

- Adjust**: A dropdown menu.
- Designation**: 2 CH STEPPER MOTOR MOD.
- Symbol**: A section with three dropdown menus: "Choose axis:" (Channel 0), "Function:" (Position control), and "Task:" (MAST).
- Trajectory**: A section with four input fields:
  - Start/Stop frequency: 200 Hertz
  - Acceleration: 50.000 pulses/s<sup>2</sup>
  - Software hi limit: 500.000 pulses
  - Software lo limit: -1.000.000 pulses
- Stop level**: A section with one input field:
  - Period: 50 ms
- Manual mode parameter**: A section with two input fields:
  - Speed: 5.000 Hertz
  - Origin value: 0 pulses
- Brake output**: A section with two input fields:
  - Timeout on deactivation: 0 ms
  - Timeout on activation: 0 ms

### 1.4-4 Presymbolization

Application-specific modules provide a way of allocating symbols automatically to objects which are associated with them. The user gives the generic symbol for channel %CHxy.i of the module, and all the symbols for the objects associated with this channel can then be generated automatically on request.

This presymbolization operation carried out in the variable editor makes programming easier as mnemonics are used rather than addresses which are more difficult to handle.

These objects are symbolized using the following syntax :

#### User\_prefix\_Manufacturer\_suffix

where

The **User\_prefix** is the generic symbol given by the user to channel %CHxy.i (12 characters maximum).

The **Manufacturer\_suffix** is the part of the symbol which corresponds to the channel bit or word (20 characters maximum) given by the system.

In addition to the symbol, a manufacturer comment is generated automatically which gives a brief description of the role of the object.

**Example :** Axis\_0 is the user prefix for channel 0.

Address	Type	Symbol	Comment
%I1.0	EBOOL	Axis_0_next	Ready for next command block
%I1.0.1	EBOOL	Axis_0_done	All instructions have been completed
%I1.0.2	EBOOL	Axis_0_at_fit	Fault on the axis %<VOIE>
%I1.0.3	EBOOL	Axis_0_at_ok	No fault on the axis %<VOIE>
%I1.0.4	EBOOL	Axis_0_hd_err	Hardware error on the axis %<VOIE>
%I1.0.5	EBOOL	Axis_0_at_err	Presence of an error on the axis %<VOIE>
%I1.0.6	EBOOL	Axis_0_cmd_nok	Command refused
%I1.0.7	EBOOL	Axis_0_nomotion	No motion on the axis %<VOIE>
%I1.0.8	EBOOL	Axis_0_at_pnt	Axis %<VOIE> is in position
%I1.0.9	EBOOL		
%I1.0.10	EBOOL	Axis_0_sys_err	System error on the axis %<VOIE>
%I1.0.11	EBOOL	Axis_0_conf_ok	Axis %<VOIE> has been configured
%I1.0.12	EBOOL	Axis_0_ref_ok	Axis %<VOIE> has been calibrated
%I1.0.13	EBOOL	Axis_0_at_evt	Image of the physical event input
%I1.0.14	EBOOL	Axis_0_home	Image of the physical home switch input
%I1.0.15	EBOOL	Axis_0_direct	Displacement in plus (=1), in minus (=0) direction
%I1.0.16	EBOOL	Axis_0_in_off	In Drive Off mode
%I1.0.17	EBOOL	Axis_0_in_dirdr	In Direct Drive mode
%I1.0.18	EBOOL	Axis_0_in_manu	In Manual mode
%I1.0.19	EBOOL	Axis_0_in_auto	In Automatic mode
%I1.0.20	EBOOL	Axis_0_st_dirdr	Motion in Direct Drive mode
%I1.0.21	EBOOL	Axis_0_st_jog_p	Motion in Jog plus (+) mode
%I1.0.22	EBOOL	Axis_0_st_jog_m	Motion in Jog minus (-) mode
%I1.0.23	EBOOL	Axis_0_st_inc_p	Motion in Incremental plus (+) mode
%I1.0.24	EBOOL	Axis_0_st_inc_m	Motion in Incremental minus (-) mode
%I1.0.25	EBOOL	Axis_0_st_setp	Motion in Manual Calibration mode
%I1.0.26	EBOOL	Axis_0_on_pause	Differed Pause is activated (Motions train is suspended)
%I1.0.27	EBOOL	Axis_0_im_pause	Immediate Pause is activated (Current motion is suspended)
%I1.0.28	EBOOL	Axis_0_stepflt	...
%I1.0.29	EBOOL	Axis_0_emgstop	Locking on fault: Emergency stop fault
%I1.0.30	EBOOL	Axis_0_st_stop	Unlocking on fault: Stop fault

## 1.4-5 Debugging

In online mode, the configuration editor also provides the user with a control panel screen, giving him a quick visual display which he can use to control and observe the behavior of the axis.

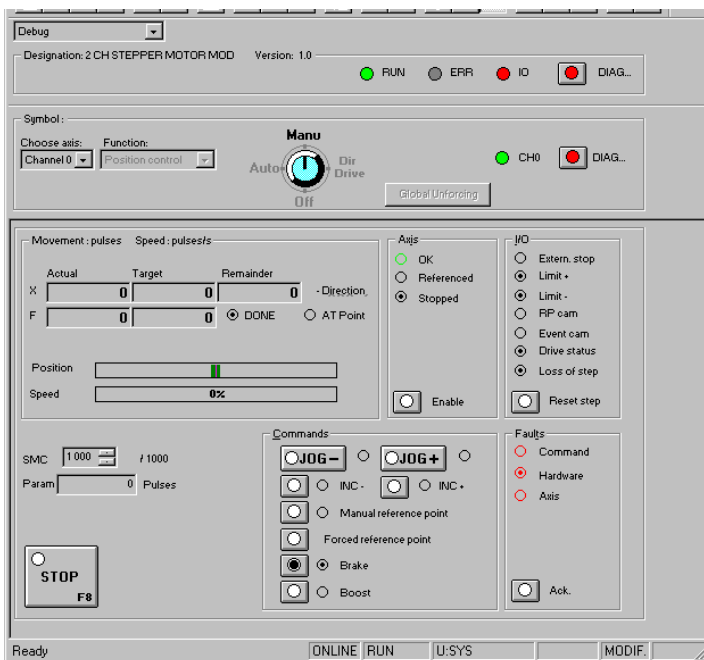
The control panel provides different information and commands according to the selected operating mode :

- automatic mode (AUTO),
- manual mode (MANU),
- direct (DIRDRV) mode,
- stop (OFF) mode (no movement is possible)

The upper part of the debug screen is identical in all modes. It gives information on the operating state of the module and diagnostic information, and is used to choose the module channel and select the mode.

The lower part gives information and commands specific to the selected operating mode :

- information on the movement,
  - information on the state of the axis and the I/O,
  - manual movement commands (when this mode is selected),
  - information on faults
- etc.





---

#### **1.4-6 Man-machine interface and control**

The user can make use of all the commands and all the axis parameters and measurements in the processor in the form of language objects. He can thus design the control interface for his machine and include in it all or part of the axis control data.

This man-machine interface can be supported by CCX17 terminals.

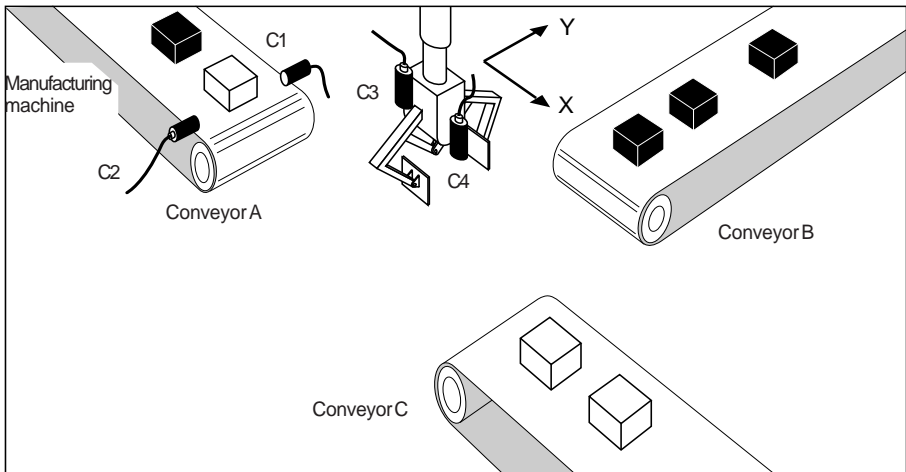


## 2.1 Description of the example

This example is given for information and learning purposes. It will enable you to follow all the stages involved in setting up a TSX CFY axis control system without having to read all the documentation.

A transfer device removes all the items as they leave the manufacturing process. This device consists of a clamp which can move spatially on a plane (X and Y axes) parallel to the ground.

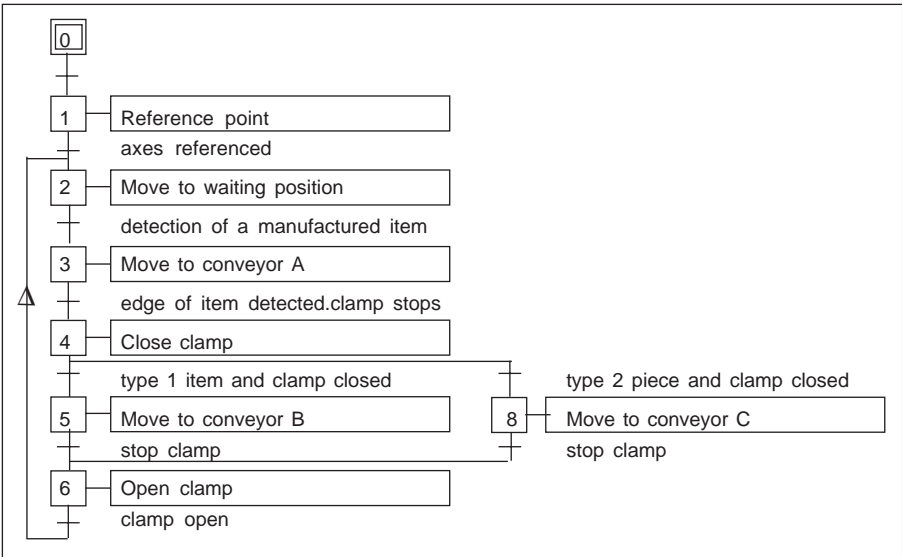
As soon as an item appears on exit conveyor A, the clamp will automatically pick it up and transfer it to conveyor B or conveyor C, depending on the type of item. The clamp then returns to waiting position ready to pick up another manufactured item as soon as one is detected.



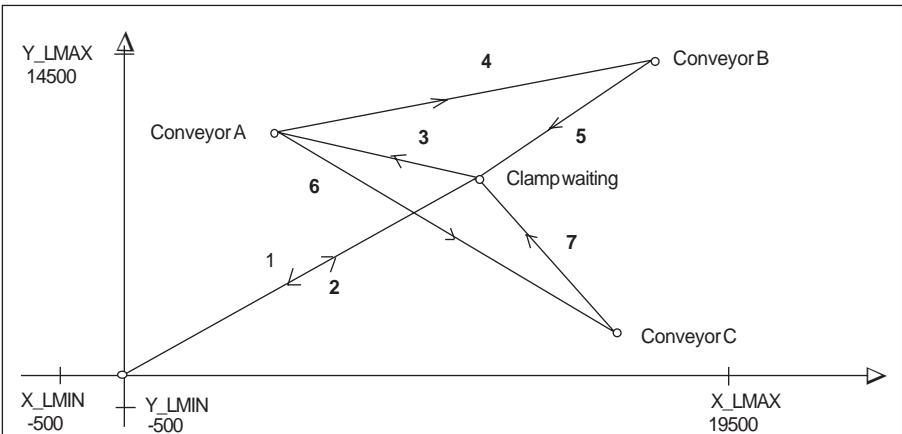
### I/O :

- C1 : cell for detecting the presence of a manufactured item,
- C2 : sensor for identifying the type of item,
- C3 : sensor for detecting clamp open/clamp closed,
- C4 : sensor for detecting the edge of an item (in the clamp), connected to module event input,
- O/F clamp : open/close clamp command.

## Grafcet chart for the application



## Description of the trajectory



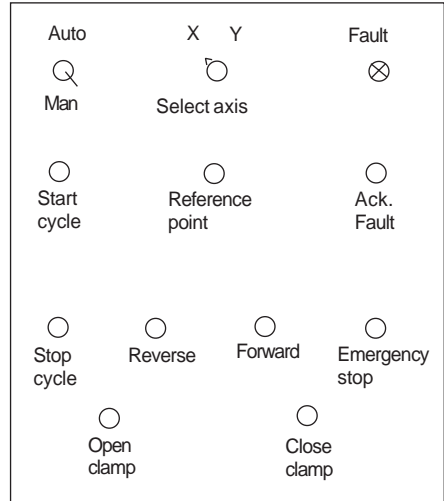
- 1 Reference point at speed  $V_{rp}$
- 2 Move at speed  $V_{ret}$  to waiting position  $(X_{wait}, Y_{wait})$  and stop
- 3 Move to conveyor A  $(X_A, Y_A)$  at speed  $V_A$  until the manufactured item is detected
- 4 Move at speed  $V_B$  to conveyor B  $(X_B, Y_B)$  and stop
- 6 Move at speed  $V_C$  to conveyor C  $(X_C, Y_C)$  and stop
- 5 and 7 Move at speed  $V_{ret}$  to waiting position  $(X_{wait}, Y_{wait})$  and stop

## Man-machine interface

The following commands are all on the front panel, and are used to control the moving part manually when there is a fault in the installation. The commands and the indicator lamps are controlled by an input module and a discrete output module.

Description of the commands :

- **Auto/Man** : switch for selecting the operating mode,
- **Start Cycle** : automatic execution of the cycle,
- **Stop Cycle** : automatic cycle stop,
- **Select axis X/Y** : selects the axis to be controlled in manual mode,
- **Reference point** : manual reference point for the selected axis,
- **Forward/Reverse** : manual move command in positive or negative direction, for the selected axis.
- **Fault** : indicator lamp signaling any hardware or application fault,
- **Ack. Fault** : fault acknowledgment command,
- **Emergency stop** : immediate stop of the moving part whatever mode is selected.
- **Open clamp** : open clamp command,
- **Close clamp** : close clamp command.



## 2.2 Prerequisites

Only functions which are specific to axis control will be described here. It is therefore assumed that the following operations have been performed :

- PL7 software has been installed,
- the hardware has been installed : module and translators controlling the 2 axes have been wired.

---

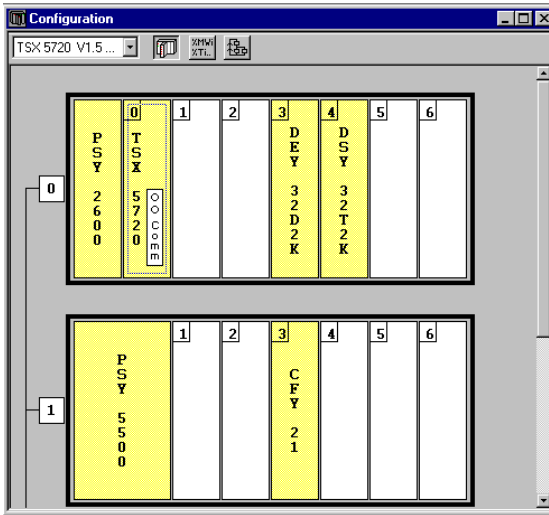
## 2.3 Application design

---

### 2.3-1 Software declaration of the PLC configuration used

Launch PL7 software, select the **File/New** command and select a TSX 57 20 processor.

Using the Application Browser, double-click on the Station, Configuration and then Hardware configuration icons.



Then select each element of the PLC configuration. The following selections have been made in this application :

- rack 0 and rack 1 : TSX RKY 8E
- 32-input module : TSX DEY 32D2K in position no.3 of rack 0
- 32-output module : TSX DSY 32T2K in position no.4 of rack 0
- 2-axis control module : TSX CFY 21 in position no.3 of rack 1

---

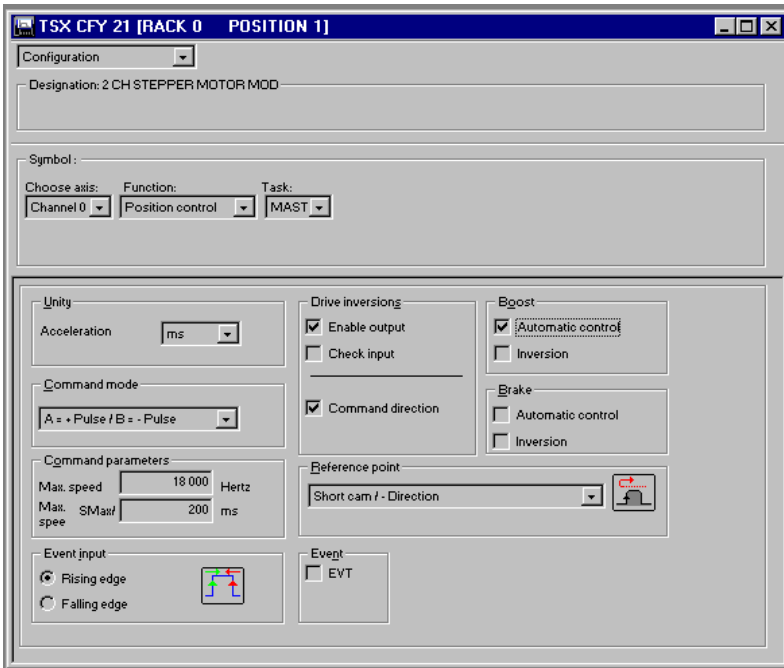
### 2.3-2 Entering the configuration parameters for each axis

Select position no.3 of rack 1 and execute the **Edit/Open Module** command (or double-click on the selected module).

#### Configuration of channel 0

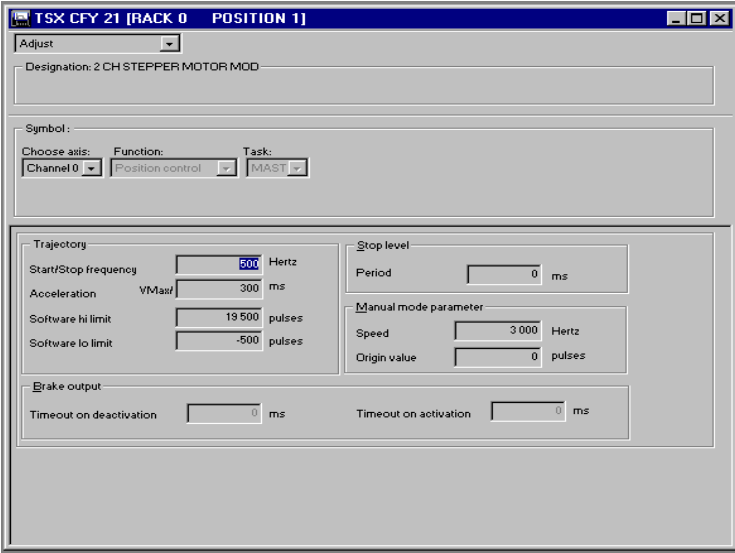
For channel 0, select the position control function and the MAST task.

Enter the configuration parameter values.





Parameter	Description	Value
Units	Acceleration	ms
Control mode		A=pulse/B=Direction
Control parameters	Max speed Max acc.	18 000 Hz 200 ms
Translator reversals	Enable Output Check input Direction of the command	Reversal No reversal Reversal
Boost	Automatic management Reversal	Selected Without
Brake	Automatic management Reversal	not selected Without
Reference point		Short cam / Negative direction
Reflex input		Rising edge
Event		not used

Click on the right arrow in the Configuration field and click on Adjust.  
Enter the configuration parameter values as shown in the table



Parameter	Description	Value	Note
Trajectory	Start/stop frequency	500 Hertz	
	Acceleration	VMax/300 ms	Acceleration/deceleration phase duration
	Upper soft stop	19 500 pulses	Axis length =
	Lower soft stop	-500 pulses	20 000 pulses
Stop plateau	Duration	0 ms	Not used
Manual mode parameters	Speed	3000 Hz	
	RP value	0 pulse	Reference position
Brake output	Automatic management	not selected	
	Reversal	Without	
Reference point		Short cam / Negative direction	

Confirm all parameters (Configuration + Control) using the **Edit/Confirm** command or the  icon.

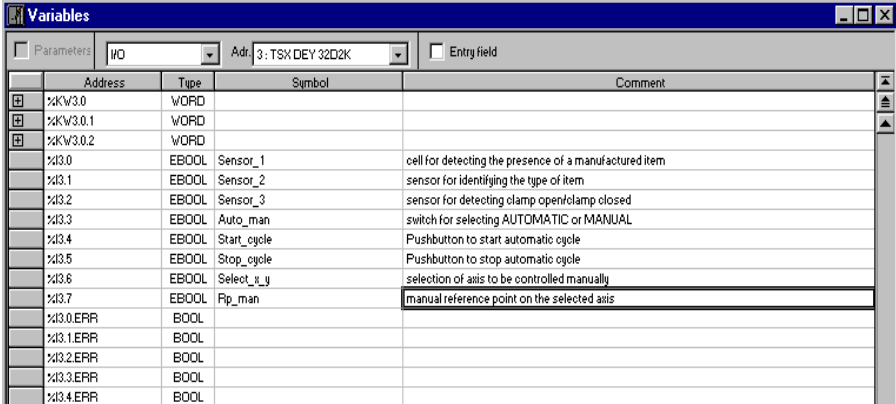
In the basic configuration editor screen, confirm the configuration using the **Edit/Confirm** command or the  icon.

Proceed in the same way for channel 0 and give the same values for the parameters.



### 2.3-3 Entering the symbols for the application

This is performed by double-clicking on the Variables icon and then the I/O icon in the Application Browser.



Symbol	Object	Role
Sensor_1	%I3.0	Cell for detecting the presence of a manufactured item
Sensor_2	%I3.1	Sensor for identifying the type of item (0=type 2, 1=type 1)
Sensor_3	%I3.2	Sensor for detecting clamp open/clamp closed
Auto_man	%I3.3	Switch for selecting AUTOMATIC (=0) or MANUAL (=1) mode
Start_cycle	%I3.4	Pushbutton to start automatic cycle
Stop_cycle	%I3.5	Pushbutton to stop automatic cycle
Select_x_y	%I3.6	Selection of axis to be controlled manually (1=X, 0=Y)
Rp_man	%I3.7	Manual reference point on the selected axis
Forward	%I3.8	Move moving part in positive direction on the selected axis
Reverse	%I3.9	Move moving part in negative direction
Ackflt	%I3.10	Fault acknowledgment
Emg_stop	%I3.12	Emergency stop
O_clamp	%I3.13	Pushbutton to open the clamp
C_clamp	%I3.14	Pushbutton to close the clamp
Clamp	%Q4.0	Open/close clamp actuating command (o=open, 1=close)
Fault	%Q4.1	Fault indication
X_wait	%MD50	Waiting position (X axis)
Y_wait	%MD52	Waiting position (Y axis)
X_b	%MD54	Position of conveyor B (X axis)
Y_b	%MD56	Position of conveyor B (Y axis)
X_c	%MD58	Position of conveyor C (X axis)
Y_c	%MD60	Position of conveyor C (Y axis)

Symbol	Object	Value	Role
Cycle	%M0		Condition of the machine in work mode
Speed_r_p	%KD0	5000	Reference point speed on X and Y axes
Speed_x_wait	%KD4	10000	Speed towards waiting position, X axis
Speed_y_wait	%KD6	10000	Speed towards waiting position, Y axis
Speed_pos_a_x	%KD8	15000	Speed towards conveyor A position, X axis
Speed_pos_a_y	%KD10	15000	Speed towards conveyor A position, Y axis
Speed_pos_b_x	%KD12	15000	Speed towards conveyor B position, X axis
Speed_pos_b_y	%KD14	15000	Speed towards conveyor B position, Y axis
Speed_pos_c_x	%KD16	12000	Speed towards conveyor C position, X axis
Speed_pos_c_y	%KD18	12000	Speed towards conveyor C position, Y axis

### Entering symbols for the axis control module

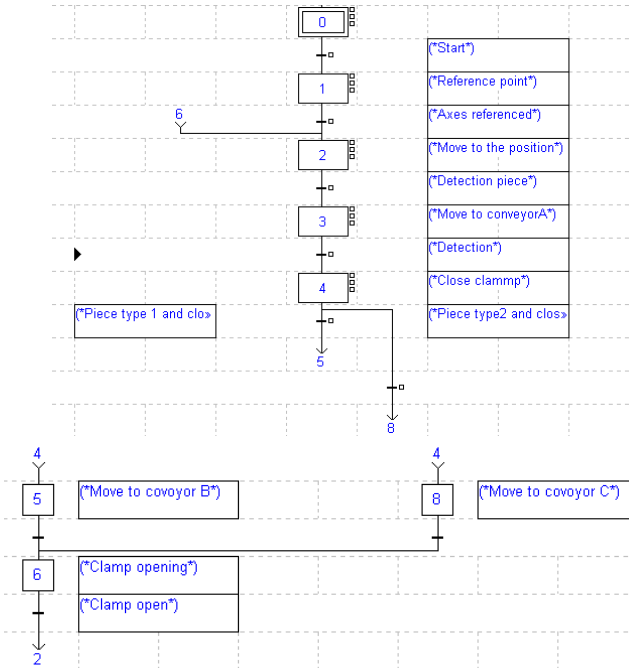
Symbol	Object	Symbol	Object
Axis_x	%CH103.0	Axis_y	%CH103.1
Next	%I103.0	Next_y	%I103.1
Done	%I103.0.1	Done_y	%I103.1.1
Error	%I103.0.2	Error_y	%I103.1.2
Ok	%I103.0.3	Ok_y	%I103.1.3
Hard_err_x	%I103.0.4	Hard_err_y	%I103.1.4
Axis_error_x	%I103.0.5	Axis_error_y	%I103.1.5
Ref_cmd_x	%I103.0.6	Ref_cmd_y	%I103.1.6
At_point	%I103.0.8	At_point_y	%I103.1.9
Conf_x	%I103.0.11	Conf_y	%I103.1.11
Calib	%I103.0.12	Calib_y	%I103.1.12
Mode_drive_off	%I103.0.16	Mode_drive_off_y	%I103.1.16
Mode_dir_drive	%I103.0.17	Mode_dir_drive_y	%I103.1.17
Mode_manual	%I103.0.18	Mode_manual_y	%I103.1.18
Mode_auto	%I103.0.19	Mode_auto_y	%I103.1.19
Trans_x	%I103.0.35	Varvalid_y	%I103.1.35
Dirdrive	%Q103.0	Dirdrive_y	%Q103.1
Jog_p	%Q103.0.1	Jog_p_y	%Q103.1.1
Jog_m	%Q103.0.2	Jog_m_y	%Q103.1.2
Inc_p	%Q103.0.3	Inc_p_y	%Q103.1.3
Inc_m	%Q103.0.4	Inc_m_y	%Q103.1.4
Setrp	%Q103.0.5	Setrp_y	%Q103.1.5
Rp_here	%Q103.0.6	Rp_here_y	%Q103.1.6
Ackflt	%Q103.0.9	Ackflt_y	%Q103.1.9
Enable	%Q103.0.10	Enable_y	%Q103.1.10
Event_uc	%Q103.0.11	Event_uc_y	%Q103.1.11
Posrp	%MD103.0.22	Posrp_y	%MD103.1.22

### 2.3-4 Programming

The programming in this example uses Grafcet structure :

- the sequential processing for the sequential description of the application : processing of the automatic cycle,
- the preprocessing for managing the operating modes,
- the post-processing for the execution of manual mode.

#### Sequential processing



Step 0 :

Transition X0 > X1

!(\*Channel X not faulty, clamp open, switch Auto\_man to Auto, start cycle, channel Y not faulty and automatic mode active\*)

```
NOT Error AND Sensor_3 AND NOT Auto_man AND Cycle
AND NOT Error_y AND Mode_auto
```

Step 1 : Action on activation

!(\*Reference point on X and Y axes\*)

```
SMOVE Axis_x(1,90,14,0,Speed_r_p,0);
```

```
SMOVE Axis_y(1,90,14,0,Speed_r_p,0);
```

Transition X1 > X2

!(\*Test : axes ready and referenced\*)

```
Done AND Calib AND DONE_Y AND CALIB_Y
```

---

```
Step 2 : Action on activation
!(*Move to waiting position (Xwait, Ywait)*)
SMOVE Axis_x(2,90,9,X_wait,Speed_x_wait,0);
SMOVE Axis_y(2,90,9,y_wait,Speed_y_wait,0);

Transition X2 > X3
!(*Mobile in waiting position and item detected on conveyor A*)
sensor_1 AND Next AND Cycle AND Next_y

Step 3 : Action on activation
!(*Move to conveyor A*)
SMOVE Axis_x(3,90,10,19500,Speed_pos_a_x,0);
SMOVE Axis_y(3,90,10,19500,Speed_pos_a_y,0);

Transition X3 > X4
!(*Moving part in position to pick up item detected on conveyor A*)
At_point AND Next AND Next_y AND At_point_y

Step 4 : Continuous action
!(*Close clamp*)
SET Clamp;

Transition X4 > X5
!(*Type 1 item and clamp closed*)
sensor_2 AND sensor_3

Step 5 : Action on activation
!(*Move to conveyor B*)
SMOVE Axis_x(4,90,9,X_b,Speed_pos_b_x,0);
SMOVE Axis_y(4,90,9,Y_b,Speed_pos_b_y,0);

Transition X4 > X8
!(*Type 2 item and clamp closed*)
Not sensor_2 AND sensor_3

Step 8 : Action on activation
!(*Move to conveyor C*)
SMOVE Axis_x(5,90,9,X_c,Speed_pos_c_x,0);
SMOVE Axis_y(5,90,9,Y_c,Speed_pos_c_y,0);

Transition X5 > X6
!(*Moving part in position on conveyor B*)
At_point AND Next AND Next_y AND At_point_y

Transition X8 > X6
!(*Moving part in position on conveyor C*)
At_point AND Next AND Next_y AND At_point_y

Step 6 : Continuous action
!(*Clamp opening*)
RESET Clamp;

Transition X6 > X2
!(*Clamp open*)
NOT sensor_3 AND Cycle
```

---

### Preprocessing

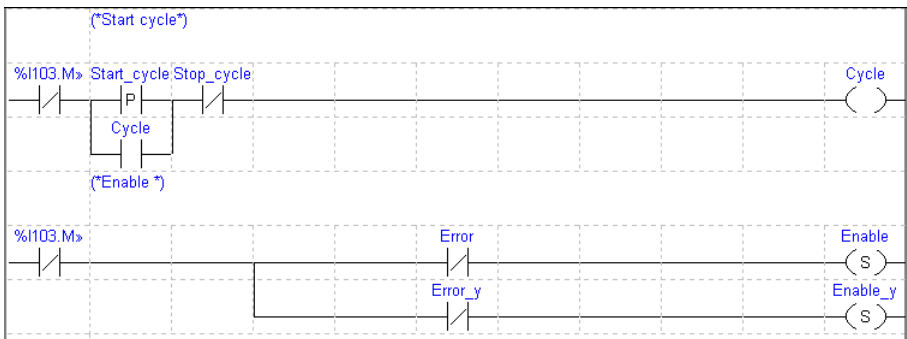
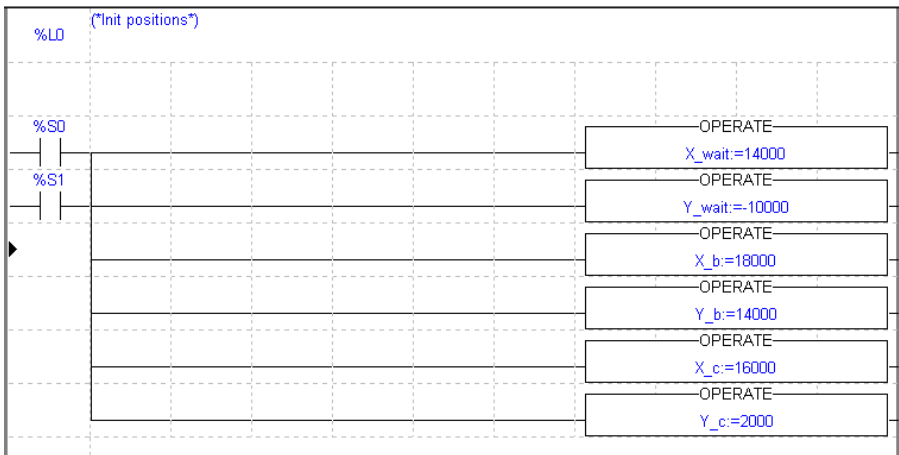
Preprocessing includes the management of the operating modes.

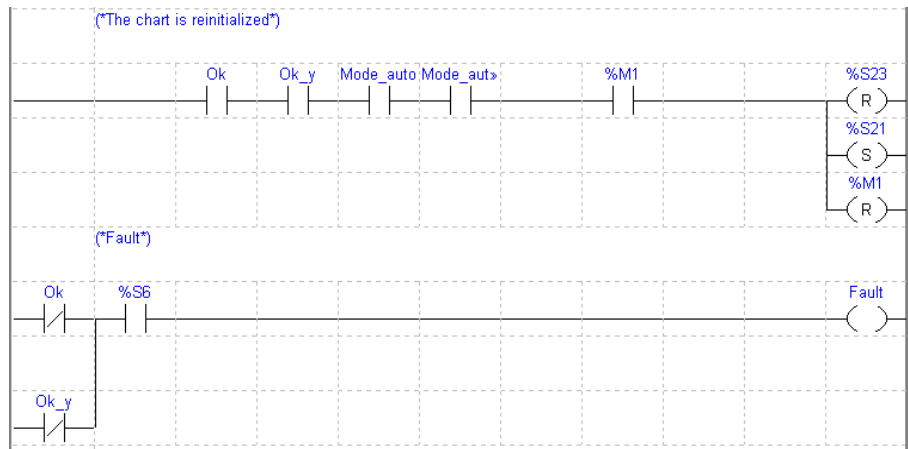
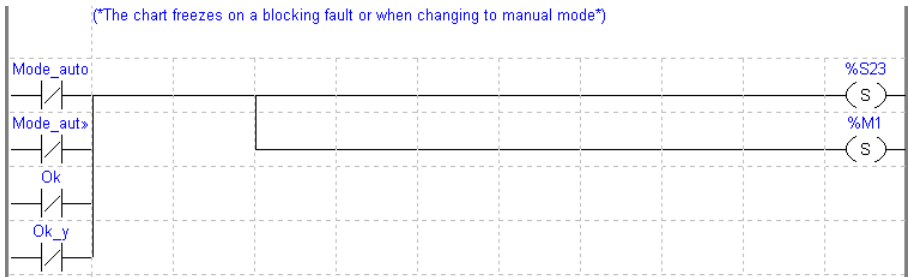
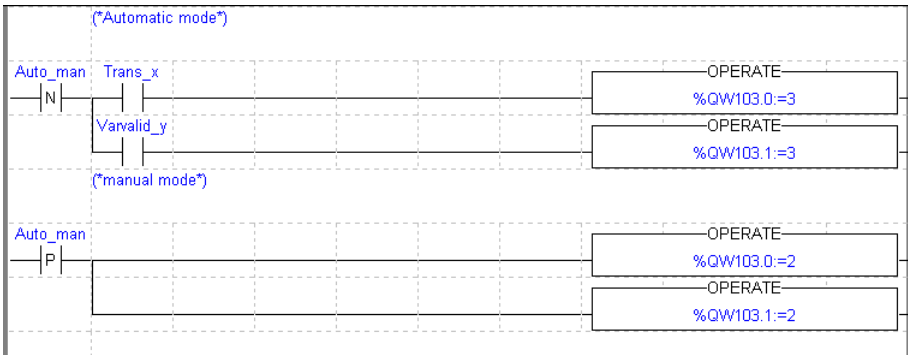
On a blocking fault :

- the chart freezes,
- the operator can then control the moving part in manual mode and correct and acknowledge the fault from the front panel.
- the chart is reinitialized when the fault has disappeared and has been acknowledged.

When changing to manual mode :

- the chart freezes,
- the chart is reinitialized when AUTOMATIC mode is reselected.





%M1 = Grafcet fixed

## Post-processing

Manual mode is managed in post-processing.

```

! (*Testing the selected mode*)
  IF Mode_auto AND Mode_auto_y AND Config_x AND Config_y
  THEN JUMP %L200;
  END_IF;
! (*Selecting the axis to drive*)
  %L100: IF NOT Selection_x_y
  THEN JUMP %L200;
  END_IF;
! (*Manual setpoint command of X axis*)
  IF RE Po_man
  THEN SET setrp;
  END_IF;
  IF NOT Po_man
  THEN RESET Setrp;
  END_IF;
! (*Moving part in + direction of X axis*)
  Jog_p:=front;
! (*Moving part in - direction of X axis*)
  Jog_m:=rear;
! %L200: IF Selection_x_y
  THEN JUMP %L300;
  END_IF;
! (*Manual setpoint command of Y axis*)
  IF RE Po_man
  THEN SET setrp_y;
  END_IF;
  IF NOT Po_man
  THEN RESET Setrp_y;
  END_IF;
! (* Moving part in + direction of Y axis*)
  Jop_p_y:=front;
! (* Moving part in - direction of Y axis*)
  Jog_m_y:=rear;
! (*Opening the clamp*)
  %L300: IF Auto_man AND op_clamp
  THEN RESET Clamp;
  END_IF;
  (*Closing the clamp*)
  IF Auto_man AND cl_clamp
  THEN SET Clamp;
  END_IF;
! (*Defaults Acknowledgement*)
  Ack_def:=Ack_def_y:=Ack_defaults;
! %L999;

```

### 2.3-5 Program transfer

Once the program has been entered, this operation consists of transferring the configuration and the program to the PLC processor memory :

- connect the terminal to the PLC using the **PLC/Connect** command,
- launch the **PLC/Transfer** command, select the "Terminal -> PLC" option, then confirm.

---

---

## 2.4 Debugging

---

### 2.4-1 Preliminary measures

As a safety measure, first perform the preliminary operations described in section 6.1. Then perform the following operations :

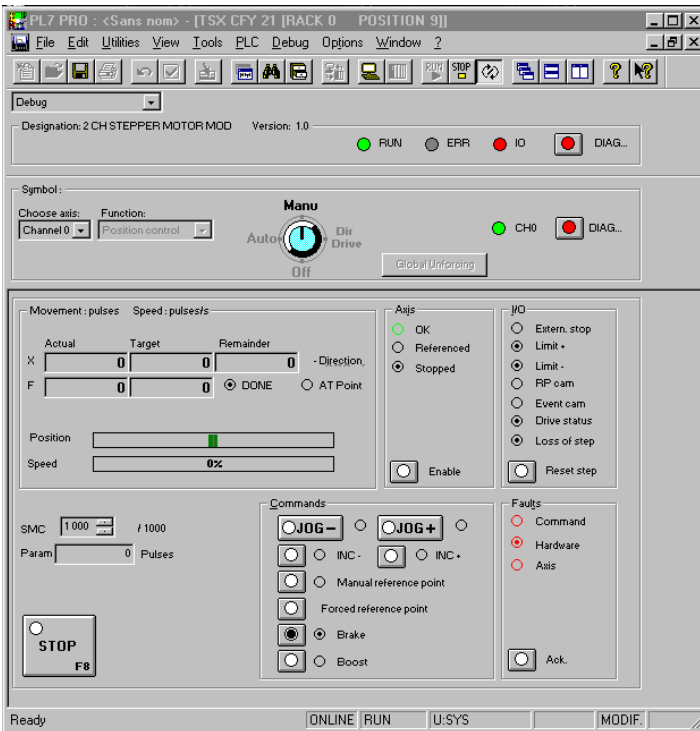
- transfer the application in the PLC if this has not been done and change to online mode,
- set the PLC to RUN mode,

---

### 2.4-2 Using manual mode

If a user wishes to move a moving part without performing the programming phase, select Manual mode. Using the Application Browser, double-click on the Configuration icon and then the Hardware configuration icon.


Select position no.3 of rack 1 and execute the **Service/Open the Module** command (or double-click on the module to be opened). The debug screen is selected by default.





---

Perform the following operations using the debug screen

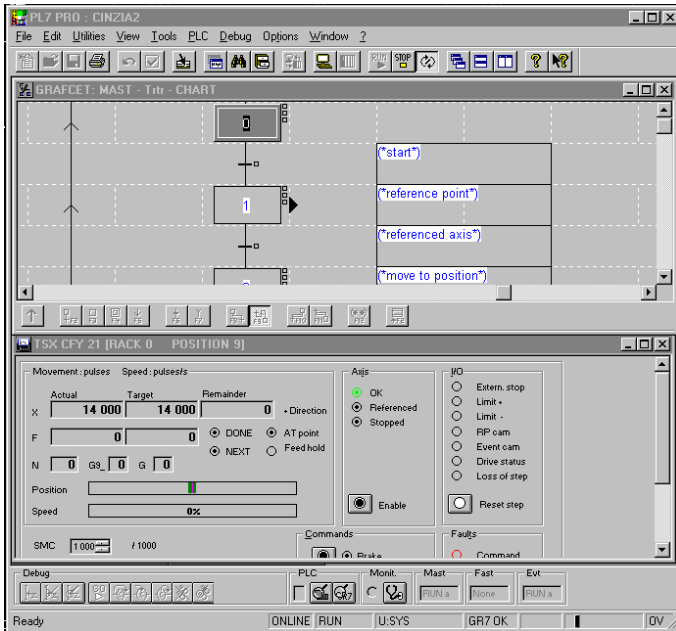
- set the PLC to RUN (**PLC/Run** command or click on the  icon),
- select the axis to be controlled : channel 0 (X axis) or channel 1 (Y axis),
- select manual mode with the mode selector in the **Man** position,
- click on the speed drive safety relay **Enable** button
- acknowledge any faults by clicking on the **Ack** button,
- set a reference point :
  - either by selecting the **Manual Reference Point** command,
  - or by selecting the **Forced Reference Point** command. In this case, first enter the value of the position of the moving part in relation to the reference point in the **Param** field,
- perform the positive direction movements using the **JOG+** command or the negative direction movements using the **JOG-** command. The position of the moving part is displayed in the X field and the speed in the **F** field.

---

## 2.4-3 Debugging

To debug the program :

- set the PLC to RUN mode,
- display the TSX CFY module debug screen,
- at the same time display the Grafcet chart screen to follow the progress of the sequential processing,
- start the program by pressing the "Start\_cycle" button on the front panel.



---

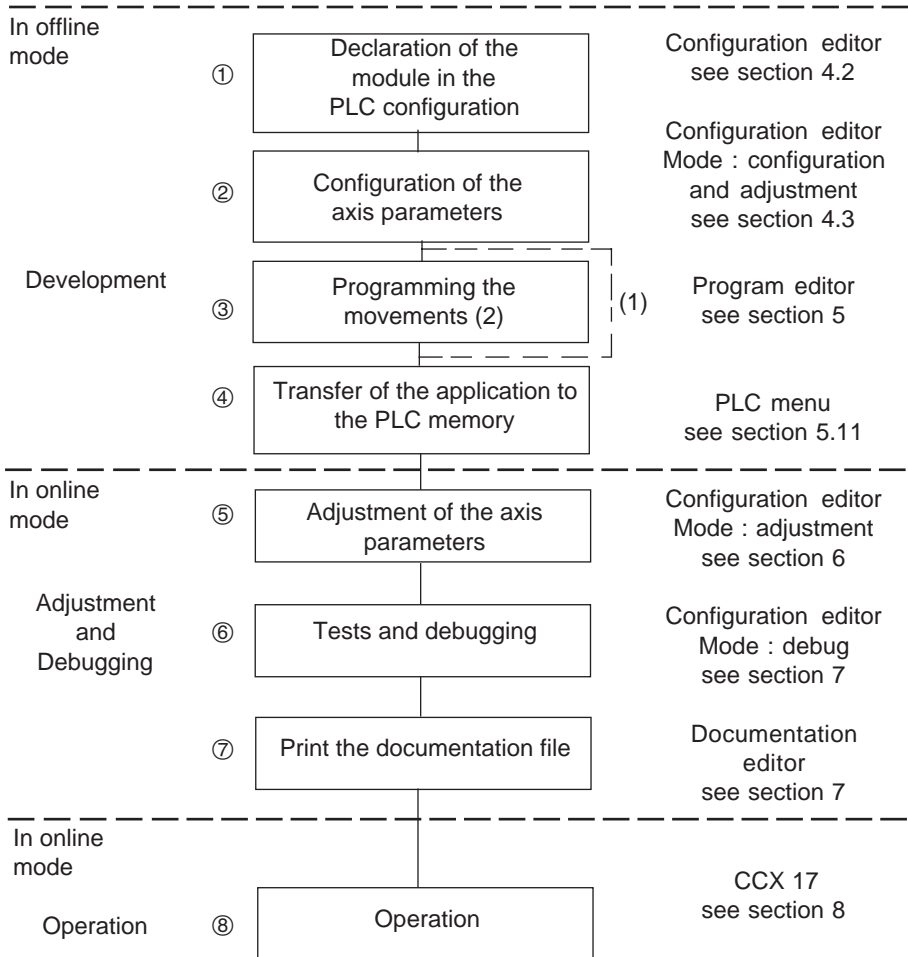
## 2.4-4 Archiving

When debugging of the program is completed :

- save the adjustment parameters by selecting the **Utilities/Save Parameters** command,
- transfer the application from the PLC processor to the hard disk to archive it, using the **PLC/Transfer** command, "PLC--> Terminal" option. Then execute the **File/Save As** command, give the application a name and confirm.

### 3.1 Setup methodology

The tutorial has shown the various phases in setting up an axis control application. The flowchart below summarizes these phases.



(1) If the user wishes, before programming, to move the moving part on the various axes in Manual mode, he can leave out operation 3. However, operations 1, 2, 4, 5 and 6 are compulsory.

(2) The programming operation may be preceded by symbolization of the variables which may be performed with the help of the variables editor.

The variables editor offers the Presymbolization function which is used to automatically generate symbols for the axis command module (see sections 1.4-4 and 5.10).

---

C

## 4.1 Configuring axis control modules

### 4.1-1 Introduction

Before creating an application program, the physical and software operating context in which it will be executed must be defined : type of TSX 57 processor, I/O modules used.

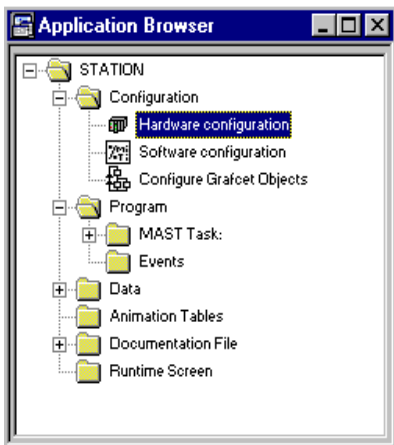
Programming axis control modules also requires the configuration parameters of the axes used to be defined.

PL7 software provides the configuration editor to perform these operations easily.

This editor also provides access to the adjustment parameters of the axes, and during operation online to the application, it is used to access the debug functions.

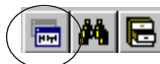
### 4.1-2 The configuration editor

Use the Application Browser to select the Station folder and then the Configuration folder, then double-click on the "Hardware configuration" icon.



If the Application Browser is not displayed :

- click on the Application Browser icon

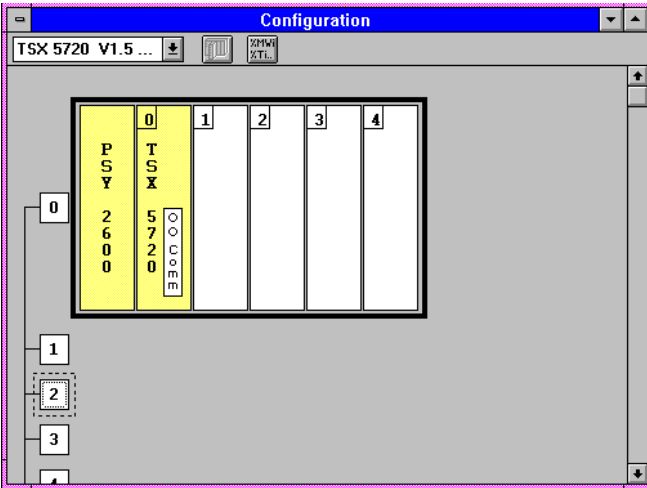


- or select **Tools/Application Browser**

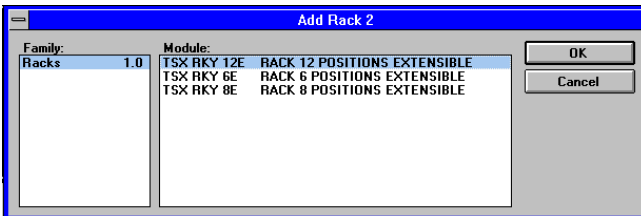
## 4.2 Declaring the axis control modules

This operation consists of declaring the positions which the TSX CFY axis control modules occupy in the PLC I/O configuration and determining their use in the master or fast task.

- Access the configuration editor.
- Select and confirm the rack, from 0 to 2 (for TSX 57 10) or 0 to 7 (for TSX 57 20), where the TSX CFY axis control module is to be installed.

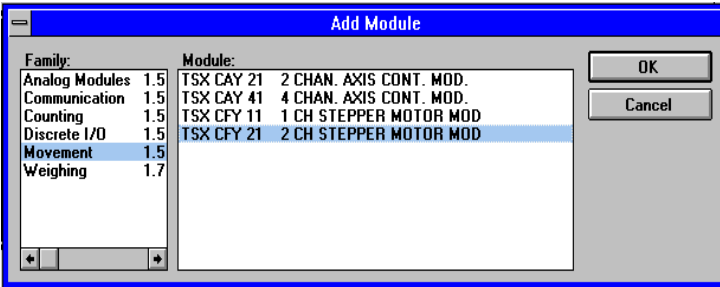


- Select the type of rack and confirm with **OK**

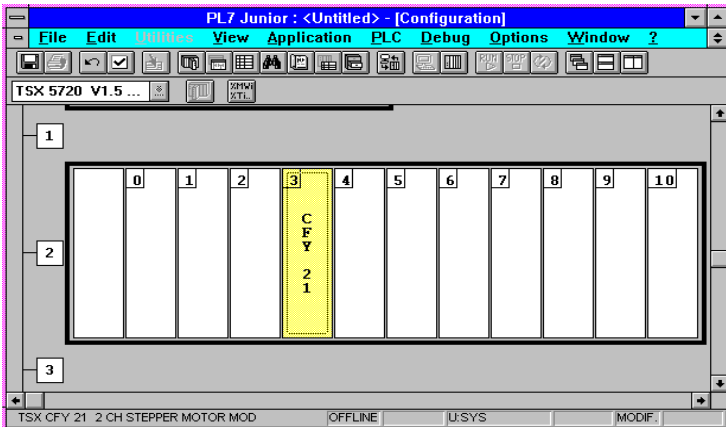


- Select and confirm the position in the rack where the TSX CFY axis control module is to be installed.

- Select the Motion Control family, then in this family select the TSX CFY axis control module and confirm with **OK**.



- After confirming, the module is declared in its position (the position contains the module reference).

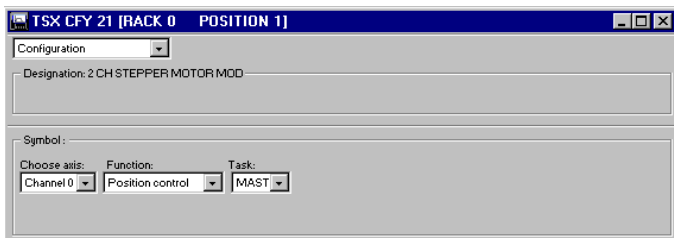


To move a module from one position to another, select the module and the command to **Edit/Move** a module, then set the target position (or simply select the module using the mouse and move it, holding the left mouse button down, to the target position).

## 4.3 Entering the configuration parameters

### 4.3-1 Access to the parameter configuration screen

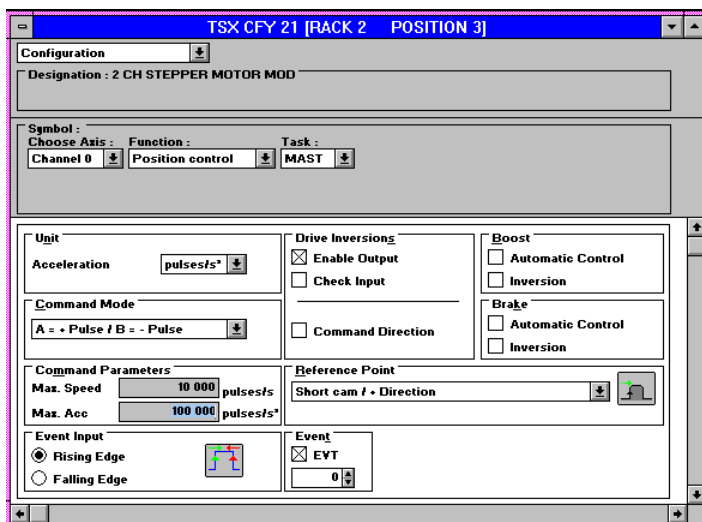
Select and confirm (or double-click) the position containing the declared axis control module.



Using the 3 pull-down lists, select :

- the axis to be controlled : channel 0 for TSX CFY 11 or channel 0 to 1 for TSX CFY 21,
- the position control function,
- the MAST or FAST task (if created) in which the channel is used.

The lower part of the screen then displays the configuration parameters.



To display the whole configuration parameters zone select the **View/Module Zone** and **View/Channel Zone** commands (to restore these zones, use the same commands).

#### Notes:

- the limits for each parameter are shown in the status bar.
- configuration of fields : Control mode A=Pulse/B=Direction, Translator reversal : Reversed enable output, (box checked) corresponds to typical configuration for controlling a Phytron MSD or SP translator.



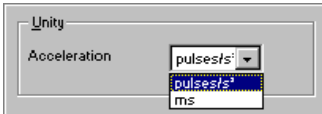


For the nine groups in the configuration screen, four contain data which is inter-linked :

- Acceleration units, maximum speed, maximum acceleration, start stop frequency, acceleration, manual mode speed,
- Automatic brake control, brake off and on delay,
- Event and event number,
- Reference point, upper and lower soft stops.

### 4.3-2 User units

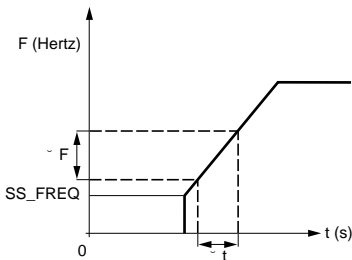
Movements and positions are always expressed in the number of pulses or increments. Speeds are always expressed in pulses per second (Hertz).



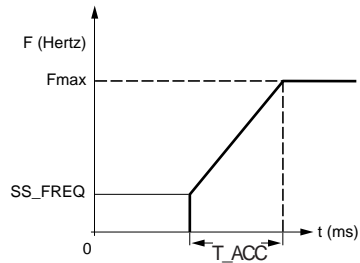
This scrolling list is used to select the physical units in which the acceleration values will be expressed.

The list offers the following units :

- **Hertz/s:** curve of acceleration and deceleration of the moving part,
- **ms:** duration of acceleration and deceleration in milliseconds.



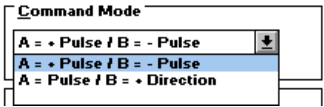
Acceleration in Hertz/s: is equal to the speed curve: relation  $\Delta F/\Delta t$



Acceleration in ms: is equal to the acceleration time, so that the speed increases from SS\_FREQ to the maximum speed.

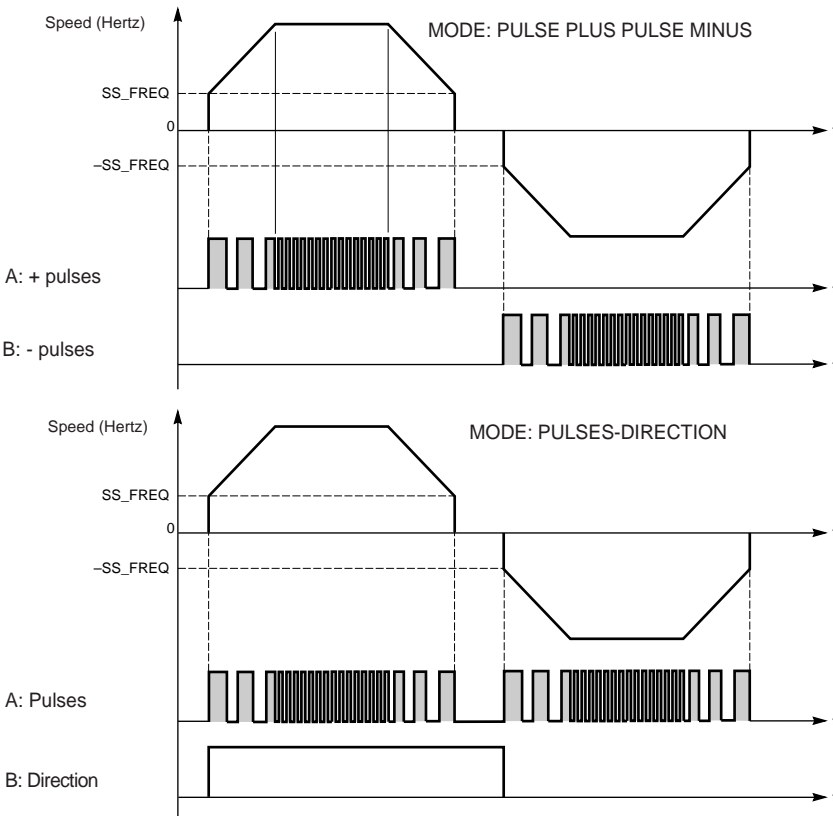
### 4.3-3 Control mode

This scrolling list is used to select the mode for applying the speed setpoint to the translator.



This setpoint is applied by two signals, **A** and **B**:

- **A = pulse + / B = pulse -** : a pulse on **A** is a command to move one step in a positive direction along the axis, a pulse on **B** is a command to move in a negative direction along the axis.
- **A = pulse / B = direction** : in this mode, **A** is a command to move one step, with the direction of movement along the axis indicated by **B**: if **B** is at state 1, movement is in a positive direction. If **B** is at state 0, movement is in the negative direction of the axis.



### 4.3-4 Control parameters

These fields are used to define the maximum speed and maximum acceleration of the axis control.

Note: in this manual, the terms speed and frequency are used interchangeably.

- **Maximum speed:** the maximum speed (frequency) depends on the combination translator - motor - moving part.

The pulse generation circuit has a resolution of 1024 points on the frequency dynamic (zero frequency included).

Depending on the **maximum speed** selected, the module channel automatically selects one of seven frequency ranges. The available frequency ranges are as follows :

Fmax from...	to...	range	Fmin
1 Hertz	936 Hertz	1	0.92 Hertz
937 Hertz	1873 Hertz	2	1.83 Hertz
1874 Hertz	4682 Hertz	3	4.58 Hertz
4683 Hertz	9365 Hertz	4	9.16 Hertz
9366 Hertz	46829 Hertz	5	45.78 Hertz
46830 Hertz	93658 Hertz	6	91.55 Hertz
93659 Hertz	187316 Hertz	7	183.11 Hertz

Note: the "range" information is internal data. It does not appear in the channel data.

#### Example:

The motor-translator combination supports speeds of up to 20 KHz: the **maximum speed** which can be declared in configuration is 20000 Hertz. The module channel selects the range immediately above, ie. 45 Hertz to 46829 Hertz; the lowest speed which the channel generates is 45.78 Hertz (frequency resolution), and the highest speed generated is 19960 Hertz (multiple of the resolution immediately below 20000 Hertz).

Data shown on the configuration screen :

Maximum speed : [ 1 , 187 316 ]

**Command Parameters**

Max. Speed  pulses/s

Max. Acc  pulses/s<sup>2</sup>

- **Maximum acceleration** : the effective acceleration of the axis in motion, which is defined during **adjustment**, must always be less than or equal to the **maximum acceleration** defined in configuration.

The TSX CFY 11/21 module channel generates acceleration / deceleration curves internally every 5 ms. The resolution during acceleration is 63 points. This means that, depending on the **adjusted** acceleration, the speed increases or decreases by 1, 2, up to 63 speed quanta, every 5 milliseconds.

#### - Case of a user acceleration unit in hertz/s :

In this case the effective acceleration in movement defined during **adjustment** is less than the value of the **maximum acceleration**.

Unit	
Acceleration	pulses/s² ↓

Unit	
Acceleration	ms ↓

Range for the above example :

Max. acceleration : [366, 23 071]

366 is the lower limit, 23071 the upper limit. 0 is a special value which corresponds to movements without acceleration curves.

The table below gives the 7 **maximum speed** ranges for each upper limit, the lower and upper limits of the **Max. acc.** (which is approximately equal to 63 times the lower limit):

Range	Max. speed	Acc. lower limit	Acc. upper limit
1	936 Hertz	183 Hertz/s	11535 Hertz/s
2	1873 Hertz	366 Hertz/s	23071 Hertz/s
3	4682 Hertz	916 Hertz/s	57678 Hertz/s
4	9365 Hertz	1831 Hertz/s	115356 Hertz/s
5	46829 Hertz	9155 Hertz/s	576782 Hertz/s
6	93658 Hertz	18311 Hertz/s	1153564 Hertz/s
7	187316 Hertz	36621 Hertz/s	2307128 Hertz/s

Note: the range information is unavailable internal data.

**- Case of a user acceleration unit in ms:**

In this case, the **maximum acceleration** corresponds to the minimum acceleration time required to reach **maximum speed** from the start stop frequency (SS\_FREQ).

<b>Unit</b>	
<b>Acceleration</b>	ms 

<b>Command Parameters</b>	
<b>Max. Speed</b>	1 000 pulses/s
<b>Max. Acc</b> <b>SMazl</b>	400 ms


Range for the above example :

Max. acceleration : [5, 5 000]
--------------------------------

The maximum limit for this time is **5000 ms**, whatever the **maximum speed** declared in configuration, and the minimum limit is 0.  
Enter this **minimum time** in the **Max. acc.** field.

**4.3-5 Type of event edge**

This field defines the type of edge (**rising edge** or **falling edge**) of the **event-specific physical input**, associated with the TSX CFY module channel. This input is also called "event input". The appearance of an edge conditions the execution of stepper motor axis control event instructions (**SMOVE**: instructions **11**, **10**, **07**, and **05**, see section 5.3).

<b>Event Input</b> <input checked="" type="radio"/> <b>Rising Edge</b> <input type="radio"/> <b>Falling Edge</b>		<b>Event</b> <input type="checkbox"/> <b>EVT</b>
--	--	---

### 4.3-6 Translator reversal

These fields are used to define the logic (positive or negative logic) of the translator **enabling output**, of the translator **control input**, as well as the movement **control direction** of signals **A** and **B**.

**Drive Inversions**

**Enable Output**

**Check Input**

---

**Command Direction**

- **Enabling output** : is used to select the logic of the translator enabling output :
  - box not checked : the enabling output is at state 1 when the translator is enabled, otherwise it is at state 0. This is for translators with an "enabling" input.
  - box checked : the enabling output is at state 0 when the translator is enabled, otherwise it is at state 1. This is for translators with an "inhibiting" input (1).
- **Control input** : is used to select the logic of the translator control input :
  - box not checked : if the control input is at state 1 : the translator is unavailable, otherwise it is available.
  - box checked : if the control input is at state 1 : the translator is available, otherwise it is unavailable.

Note: when the SUB-D translator connector is disconnected, the translator control and loss of step inputs are not activated and are seen at state 1. See the TSX CFY 11/21 modules setup manual for detailed connection instructions.

- **Control direction** : is used to select the logic of translator signals **A** and **B** (see section 4.3-3) :
  - box not checked : the direction of signals **A** and **B** is specified in section 4-3.3.
  - box checked : the logic of signals **A** and **B** is reversed :
    - when the control mode selected is A=pulse + / B=pulse - : a pulse on A is a command to move in a negative direction, a pulse on B is a command to move in a positive direction,
    - when the control mode selected is A=pulse / B=direction : a pulse on A is a command to move : if B is at state 1, the direction of movement along the axis is negative : if B is at state 0, the direction is positive.

(1) configuration for Phytron MSD/SD translator.

### 4.3-7 Boost

These fields are used to define control of the translator **boost** output.

Boost	
<input checked="" type="checkbox"/>	Automatic Control
<input type="checkbox"/>	Inversion

- **Automatic control :**

- box checked : the translator boost is activated automatically in the acceleration and deceleration stages of the moving part.
- box not checked : the translator boost is controlled by the object %Qxy.i.14 BOOST (1).

- **Inversion :**

- box not checked : the **translator boost output** is at state 1 when the boost is active, otherwise it is at state 0.
- box checked : the **translator boost output** is at state 0 when the boost is active, otherwise it is at state 1.

(1) The BOOST command remains active in automatic control. If Automatic Control is selected, this command must not be used in order to avoid any clashes.

### 4.3-8 Brake

These fields are used to define control of the stepper motor **brake output**.

Boost	
<input checked="" type="checkbox"/>	Automatic Control
<input type="checkbox"/>	Inversion

- **Automatic control :**

- box checked : the stepper motor brake control is automatically deactivated when the moving part starts and activated when it stops.
- box not checked : the brake is activated or deactivated only by acting on object %Qxy.i.13 BRAKE.

- **Inversion :**

- box not checked : the stepper motor **brake output** is at state 0 when the brake control is active, otherwise it is at state 1 (+24 V) in order to deactivate the brake.
- box checked : the stepper motor **brake output** is at state 1 when the brake control is active, otherwise it is at state 0.

(1) The BOOST command remains active in automatic control. If Automatic Control is selected, this command must not be used in order to avoid any clashes.

### 4.3-9 Event

This field is used to define (by checking the box) the number of the event-triggered task associated with the TSX CFY module (0 to 31 for a TSX 57-1• and 0 to 63 for a TSX 57-2•/57-3•/57-4•).

<b>Event</b>	
<input checked="" type="checkbox"/>	EVT
	2

Events are dealt with in section 5.

An event-triggered task is only necessary if additional processing using the reflex input is required.

### 4.3-10 Reference point

This field defines the type and direction of the reference point.

<b>Reference Point</b>		
Short cam / - Direction	↓	
Short cam / + Direction	↑	
Long cam / - Direction	↓	
Long cam / + Direction	↑	
At end limit / - Direction	↓	

The definition of "short cam" and "long cam" types depends on the connection of a reference point detector on the input (reference point cam). "On limit switch end stop" types assume that limit switch detectors are actually connected.

Possibility	Approach (1) speed	Reference point speed	Icon
• short cam / + direction	F	F	
• short cam / - direction	F	SS_FREQ	
• long cam / + direction	F	SS_FREQ	
• long cam / - direction	F	SS_FREQ	
• limit switch end stop / + direction	F	SS_FREQ	
• limit switch end stop / - direction	F	SS_FREQ	

(1) F is the speed programmed in the instruction in automatic mode or speed MAN\_SPD (defined in the adjustment screen) in manual mode. This speed can be corrected by the CMV coefficient.



---

The reference point command is implemented by :

- instruction code **14**, reference point in automatic mode,
- the manual reference point command SET\_RP.

**Note 1** : 2 other forced reference point commands (instruction code **62** and RP\_HERE command) are possible, but these do not initiate a movement and thus are irrelevant to this selection.

**Note 2** : if SS\_FREQ is zero, and if the reference point speed is SS\_FREQ, the actual reference point speed will be the lowest the module can generate within the range selected.

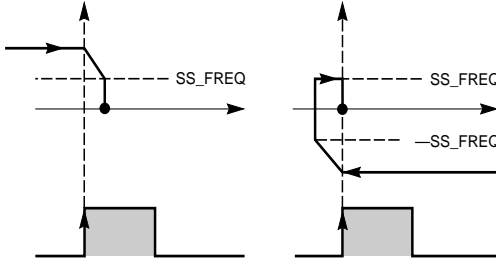
**Note** : SS\_FREQ = start stop frequency

Detailed description of each reference point set up

Type : Short cam + direction    Short cam - direction



Movement

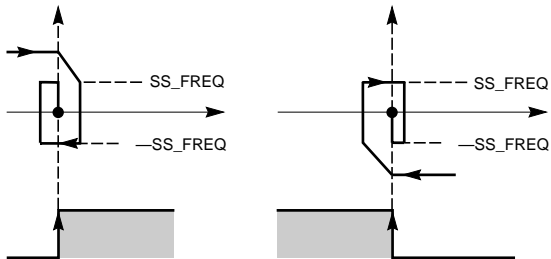


cam

Type : Long cam + direction start off cam    Long cam - direction start off cam



Movement

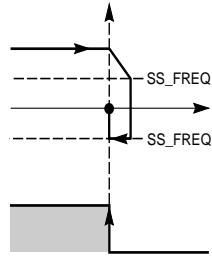
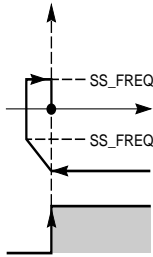


cam

<b>Type :</b>	Long cam + direction start on cam	Long cam - direction start on cam
---------------	--------------------------------------	--------------------------------------



Movement



cam

C

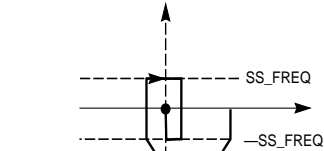
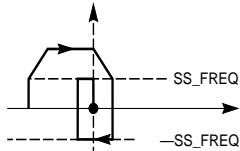
**Type :**

Limit switch + direction  
start off cam

Limit switch - direction  
start off cam



Movement



limit switch



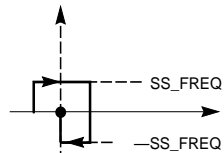
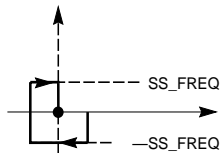
**Type :**

Limit switch + direction  
start on cam

Limit switch - direction  
start on cam



Movement



limit switch



## 4.4 Confirming the configuration parameters

When all the configuration parameters have been entered, confirm the configuration obtained using the **Edit/Confirm** command or select the  icon

If one or more of the parameter values are not within permitted limits, an error message appears indicating the parameter concerned.

Example :



Correct the parameter then confirm.

**Note** : incorrect parameters are displayed in red on the configuration screen. Grayed out parameters cannot be entered because they depend on incorrect parameters

### Important :

- The adjustment parameters are initialized as soon as there is a first request to confirm the configuration.

It is therefore possible that following modifications to the configuration values, the adjustment parameters will no longer be correct. In this case a message indicates the parameter involved :

Example :



Access the adjustment parameters screen (see section 6), correct the parameter, then confirm.

- The configuration parameters are taken into account :
  - when each of the configuration and adjustment parameters is correct,
  - when they are confirmed in the basic screen of the configuration editor.



## 5.1 Programming principle

Each channel of the axis control module is programmed in the following way :

- using the **SMOVE function** for movements in automatic mode.
- using **bit** (%I and %Q), **word** (%IW, %QW and %MW) and **double word** (%ID, %QD and %MD) **objects** associated with the module to :
  - select the operating modes,
  - control the movements (except for automatic mode),
  - check the operating status of the module and the axis.

For more information on bit and word objects see section 13.

**Note** : bit and word objects can be accessed via their address or their symbol. These symbols must be entered in the variables editor. The presymbolization function enables symbols to be automatically generated (see section 5.10).

## 5.2 Operating modes

Each TSX CFY module channel can be used in 4 modes :

- **Automatic**(AUTO) : movement commands controlled by the SMOVE.... functions are executed in this mode,
- **Manual** (MAN) : this mode enables the user to visually control the moving part from the front panel or from a man-machine interface terminal. The commands can be accessed via the output bits %Q.
- **Direct** (DIRDRIVE) : the output acts as a digital/frequency converter. This mode controls the movement of the moving part following the movement setpoint defined in the PARAM variable.
- **Stop**(OFF) : in this mode the channel only feeds back position and current speed data. This mode is forced at start-up as soon as the axis is configured.

The mode is selected using word MOD\_SEL %QWxy.i.0 :

Value

0	<b>OFF</b>	stop movement,
1	<b>DIRDRIVE</b>	command to move in direct mode,
2	<b>MANU</b>	command to move in manual mode,
3	<b>AUTO</b>	command to move in automatic mode.

**Note** : for any other value of %QWxy.i.0, OFF mode is selected.

Changing mode while a movement is in progress (bit DONE %Ix.y.i.1 at 0) stops the moving part. When the moving part is completely stationary (bit NOMOTION %Ix.y.i.8 at 1), the new mode is activated.

**Note** : only commands concerning the current mode are examined. Other commands are ignored : for example, if the channel is in MAN mode (IN\_MANU has the value 1) and the DIRDRV command is activated, it is ignored. It is necessary to first switch to DIRDRIVE mode.

---

## 5.3 Programming in automatic mode : SMOVE function

---

C

### 5.3-1 Programming an SMOVE function

SMOVE functions can be programmed in any program module in Ladder language (using an operation block), in Instruction List language (in square brackets) or in Structured Text language. The syntax is the same in all cases.

The function can be entered directly, or via the "Function Call" assisted entry screen.

**Function call**

Function Information: Parameters

Family	Lib.V	App.V	Name	Comment
Explicit exchanges	1.00	-	SMOVE	Automatic movement command
Integer tables	1.01	-		
Integrated MMI	1.05	-		
Movement Command	1.09	-		
Onphee functions	1.02	-		
Process control	1.08	-		

**Call Format**

Parameters of the METHOD :

Name	Type	Kind	Comment	Entry field
Channel	MAIN	Channel		
NPUN	vOPD	IN	Movement number	
G9	vOPD	IN	G9	
G	vOPD	IN	Movement code	

**Display the Call**

SMOVE ( )

OK Cancel Details...

### Assisted entry

In the selected program editor :

- 1 Press keys **SHIFT+ F8** simultaneously or click on the **F(...)** icon. The **Function Call** window appears.
- 2 Select the **Motion Control** family in the library.
- 3 Select the **SMOVE** function.
- 4 Press the **List...** button and fill in the various fields provided (see description on the next page) or enter the variables of the function directly in the parameter entry zone (the "parameters" option must be selected).
- 5 Confirm with **OK** or **ENTER**. The function appears.

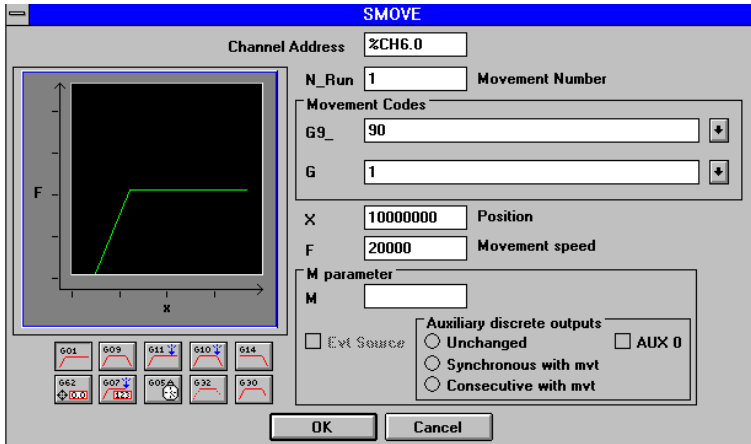


### 5.3-2 Entering the parameters of the SMOVE function

A movement command is programmed by an SMOVE function, with the following syntax :

```
SMOVE %CHxy.i(N_Run,G9_,G,X,F,M)
```

The SMOVE function **List** screen provides assisted entry for each of the fields.



where

**Channel address** Channel address of the axis control module in the PLC configuration, in the form %CHxy.i.  
**x** = rack no.  
**y** = position of the module in the rack  
**i** = channel no. (0 for the TSX CFY 11 or 0 to 1 for the TSX CFY 21)

**N\_Run** **Movement identifier** from 0 to 32767. Number identifying the movement performed by the SMOVE function. In debug mode it identifies the current movement.

## Movement codes

**G9\_** = type of movement

**90** move to an **absolute** position value,

**91** move to a **relative value with respect to the current position**.

**98** move to a **relative value with respect to the memorized position PREF**  
(position PREF is memorized using instruction code G07).

Select the type of movement using the scroll button to the right of field **G9\_** or enter the code directly during a direct entry operation (without going to the List screen).

**G** = instruction code,

**09** : Move to position and stop



**01** : Mve to position without stopping



**10** : Move until an event is detected and stop



**11** : Move until an event is detected without stopping



**14** : Reference point



**62** : Forced reference point



**05** : Await an event

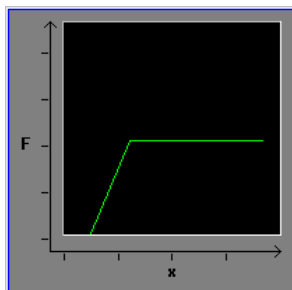


**07** : Memorize the position when an event occurs



Select the instruction code using the scroll button to the right of field **G**, or press the corresponding icon, or enter the code directly during a direct entry operation (without going to the List screen).

In the List screen : a graphic representing the selected movement is displayed (eg : code 09).





**X** = coordinate of the position to be reached (in number of pulses) or to which the moving part is to move (in the case of moving without stop).

This position can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this double word can be indexed).

**Note** : In the case of instructions G14 and G62 this value represents the reference point value. For instructions G07 and G05 see the detailed description.

**F** = speed of movement of the moving part. This speed can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this double word can be indexed).

The unit of speed is Hertz.

3 2 1 0

**Parameter M** = 16# 

--	--	--	--

Word coding optional activation of the triggering of the application event processing for instructions : 10, 11, 05 and 07 (Four-bit byte no. 3 at 1 for activation)

M= 16#0000 = no activation of event-triggered task when the SMOVE command is executed.

M= 16#1000 = activation of associated event-triggered task

This is coded automatically in field **M** in the **List** screen.

**Note :**

Speed can be corrected during movement using the CMV parameter (speed correction coefficient).

$$F_{real} = F_{programmed} \times CMV / 1000$$

This parameter, which is set by default at 1000, can be modified in the interval [1.2000], yet the resulting speed cannot be lower than SS\_FREQ.

The 0 value has a particular meaning (stop movement request).

### 5.3-3 Description of elementary movements

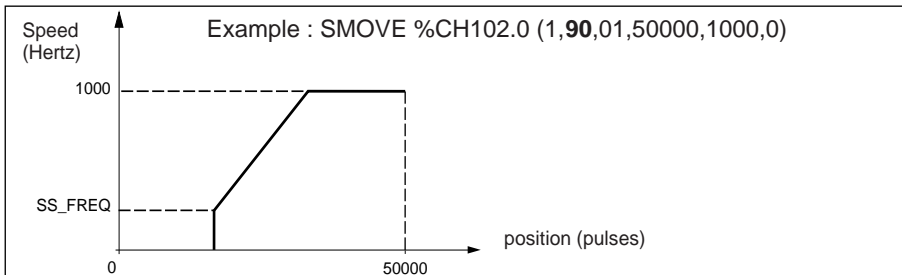
3 classes of movement can be programmed :

- move to a position (instruction codes 01 and 09),
- move until an event is detected (instruction codes 11 and 10),
- reference points (instruction code 14).

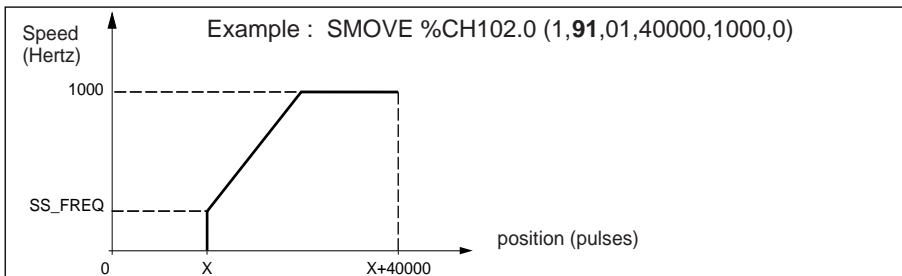
When programming these movements the user defines the position to be reached and the speed. The acceleration parameter (constant, trapezoid speed profile) is defined by an adjustable parameter.

Movements can be :

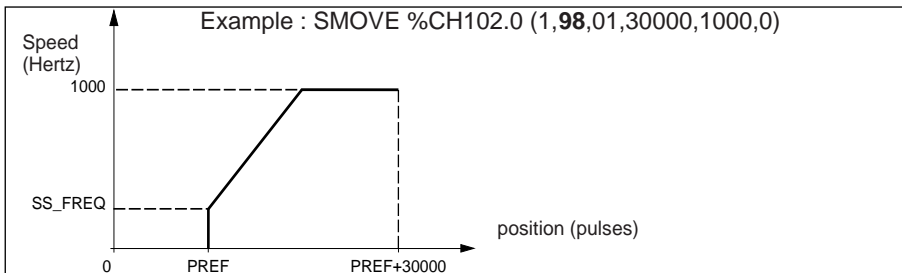
- absolute with respect to the machine reference **90**,



- relative with respect to the current position **91**,

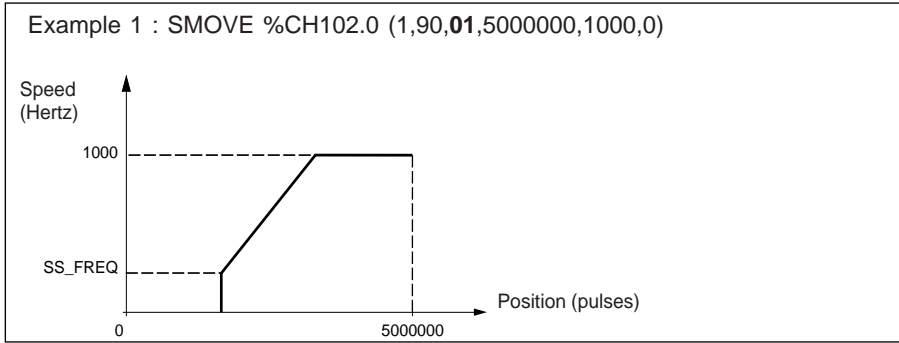


- relative with respect to the memorized position PREF **98**



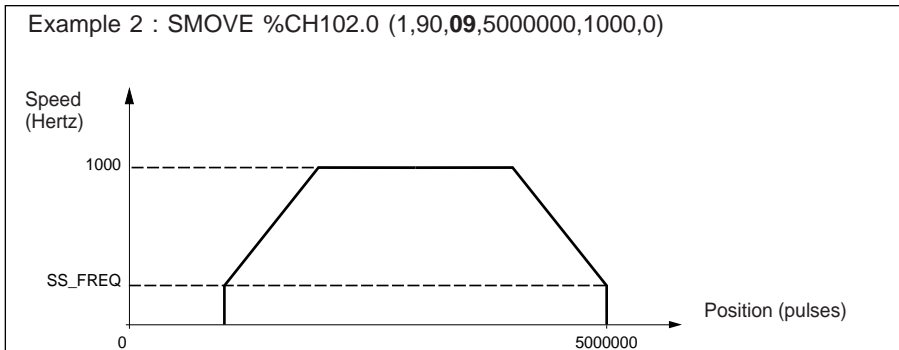
5.3-4 Description of instructions

**Move to a position without stopping : instruction code 01** 



**Note** : if instruction 01 is not followed by a movement instruction, the moving part continues its movement until it reaches the soft stops (once the target position has been passed, the velocity correction factor CMV is no longer interpreted).

**Move to a position and stop : instruction code 09** 



**Execution conditions for instructions 01 and 09** : see general conditions for execution (section 9-2).

**Move until an event is detected  
without stopping : instruction code 11  
and stop : instruction code 10**



Instructions 11 and 10 are similar to 01 and 09 in that the command ends when the event is detected (or at the entered position if the event is not detected).

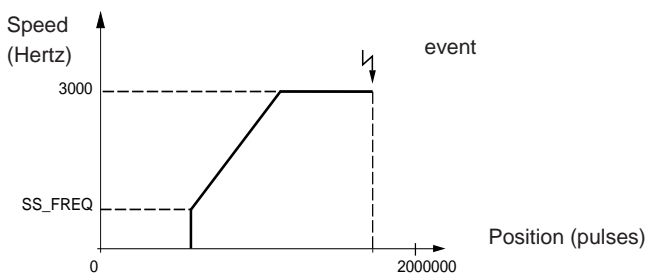
The event which is awaited can be :

- a rising or falling edge (depending on the selection made in the event edge type field in the configuration screen) on the dedicated event input associated with the channel controlling the axis,
- a rising edge on bit EXT\_EVT (%Qxy.i.11) generated via the program.

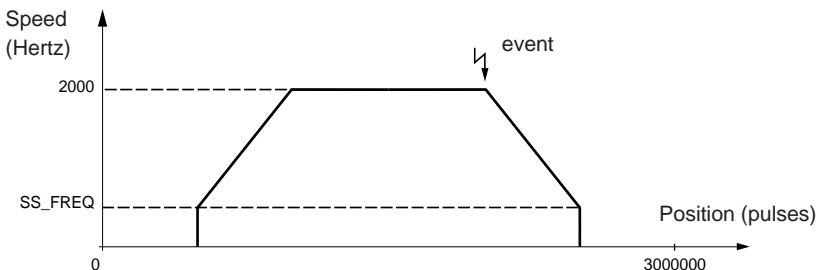
The position parameter must be defined. If the event is not detected, the instruction ends when the requested target position is reached.

These instructions may activate an event-triggered task when an event is detected if four-bit byte no. 3 of parameter M is set to 1.

Example 1 : SMOVE %CH102.0 (1,90,11,2000000,3000,0)



Example 2 : SMOVE %CH102.0 (1,90,10,3000000,2000,0)



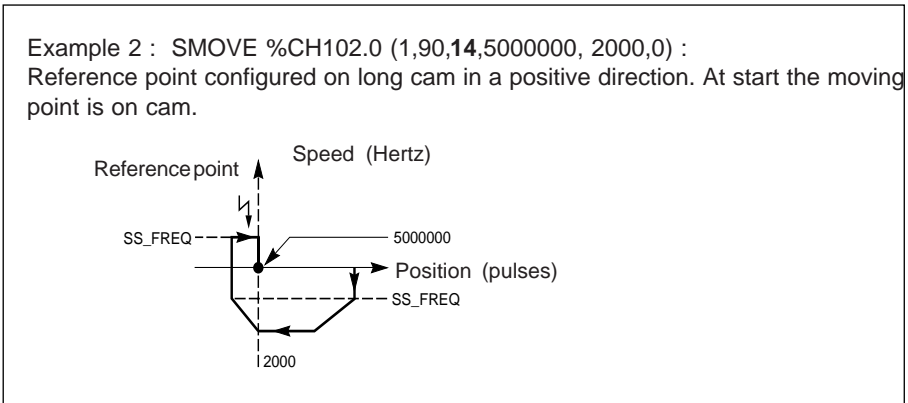
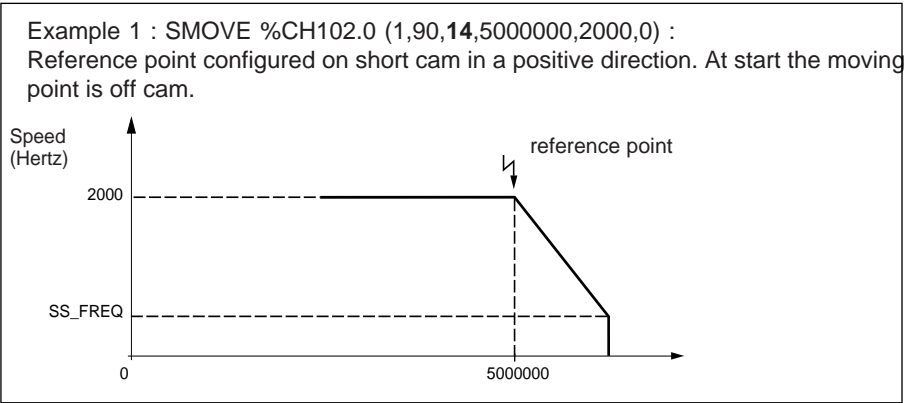
**Execution conditions for instructions 11 and 10 :**

See general conditions for execution (section 9-2).

**Reference point : instruction code 14** 

This instruction initiates a reference point sequence depending on the selection in configuration.

The value displayed in the X parameter corresponds to the coordinate to be loaded in the current value when the reference is detected.



**Execution conditions :**

This command is only accepted if the moving part is stopped : bit NOMOTION=1 (%Ixy.i.7=1).

See general conditions for execution (section 9.2)

**Forced reference point : instruction code 62**

This command sets a forced reference point (without moving the moving part). The current position value is forced to the value entered in position parameter : X.

Example :       SMOVE %CH102.0 (1,90,**62**,100000,0,0)

When this instruction is executed, the current position is forced to 100000.

**Note :**

Whatever the state of the axis : referenced or not referenced, this command is accepted and references the axis at the end of execution.

This command is only accepted if the moving part is stationary : NOMOTION bit =1 (%Ixy.i.8=1).

**Await event : instruction code 05**

This instruction makes the channel await an event (for example to generate movement on detection of an object or an instruction).

The event could be :

- a change in the state of the reflex input (a rising or falling edge depending on the selection made in configuration)
- a rising edge on bit EVT\_EXT (%Qxy.11).

The parameter F specifies the time period (resolution = 10ms).

If the event has not appeared within the time period, the command is deactivated.

If F=0, the waiting period is not limited.

Example :   SMOVE %CH102.0 (1,90,**05**,500,100,0)

It is also possible to associate event-triggered processing (in other words, activate the event-triggered task associated with the channel) when the awaited event appears. This is done by inserting the #1000 value in the parameter M of the SMOVE function.

It is advisable to systematically associate an event-triggered processing with this command as the bit TO\_G05 (%Ixy.i.39), which enables the application to distinguish whether the command has stopped on detection of an event or once the time period has elapsed), is only updated if this processing is active.

**Note :**

1-) during execution of the instruction, object %MDxy.i.10 T\_SPEED (target speed) does not contain the F waiting period parameter.

2-) for more details on event processing, see section 5.



**Memorize current position  
when an event occurs : instruction code 07**



After the execution of this instruction, when the event defined at configuration appears on the reflex input, the current position is memorized in the PREF register.

The parameter of position X should equal 1.

Example : `SMOVE %CH102.0 (1,90,07,1,0,0)`

Type of event on event input	Timing diagram	Choice at configuration
Rising edge		Rising edge of the event input
Falling edge		Falling edge of the event input

Event processing is activated when parameter M equals 16#1000, otherwise it is not activated. The event generated can be used in an event-triggered task.

**This instruction is not blocking** : the program carries on immediately to the next instruction.

The value of the current memorized position can only be accessed in the PREF register (%IDxy.i.7) if activation of the event-triggered task has been requested.

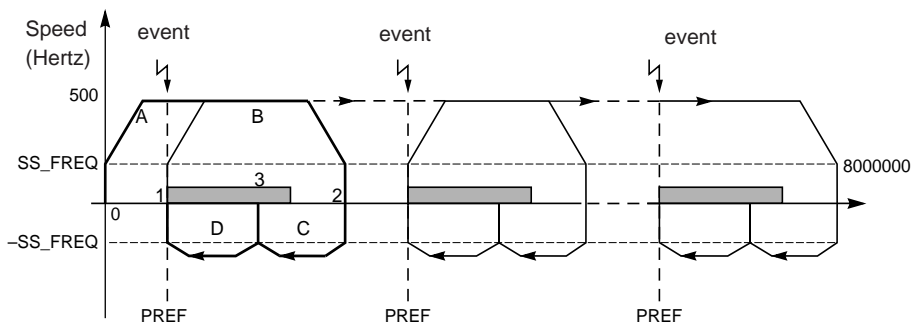
Note : during execution of this instruction, object %MDxy.i.8 T\_XPOS (target position) does not contain the parameter X=1.

### Example of using an indexed position (repetitive movements) :

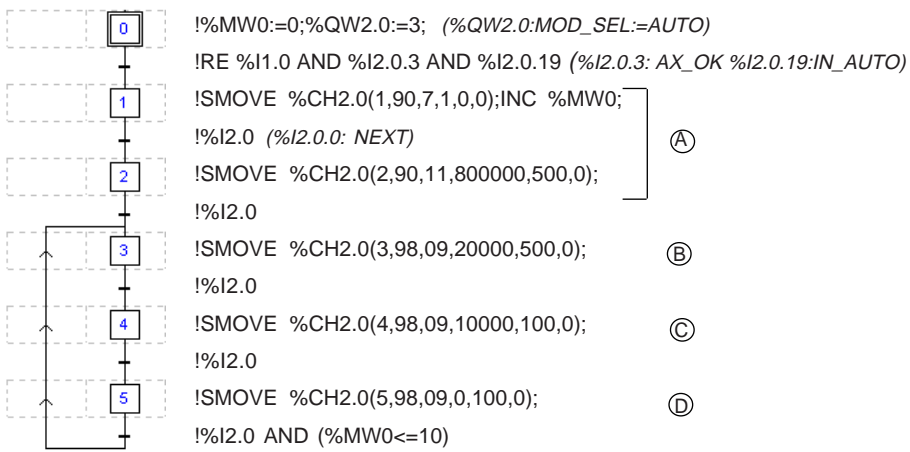
The example below gives a sequence of elementary movements to be performed 9 times :

- move **A** until the edge of the item is detected **1**,
- move **B** to position **2** = +20000 with respect to the edge of the item **1**,
- move **C** to position **3** = +10000 with respect to the edge of the item **1**,
- move **D** to the edge of the item **1**,

In this example it is assumed that the reference point has already been set and the moving part is in the start position.



**Note** : the sequence of elementary movements is represented in bold on the above chart. The numbers shown correspond to the program step numbers in the SMOVE function.



**Note** : All the actions must be programmed on activation.



### 5.3-5 Sequence of movement commands

A trajectory is created by programming a series of elementary movement instructions using the SMOVE function.

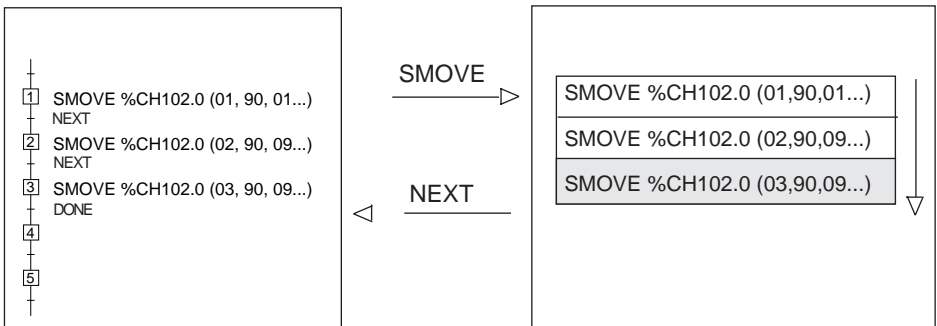
Each elementary command to execute an SMOVE function is performed once only, so the execution is programmed in :

- either Grafcet : in a step on activation or on deactivation of that step,
- or Structured Text or Ladder language on the rising edge of a bit.

The execution report for the function is provided by the module using the NEXT and DONE bits (see the table on the next page).

The TSX CFY module has a mechanism for linking movement commands together into sequences.

Each axis of the TSX CFY module has a buffer memory which can receive **two movement commands ahead** of that which it is currently executing. Thus, when the current command has been executed, it goes directly to the first command in the buffer memory.



The link between 2 movement commands is established in the following way :

- immediately if the first movement does not include a stop,
- as soon as the moving part stops, if the first movement includes a stop.


The execution time of the current instruction must be greater than the master task period so that the move from one command to the next is immediate.

**Note** : a new command must only be transmitted to the module if the buffer memory associated with the axis to be controlled is not full.

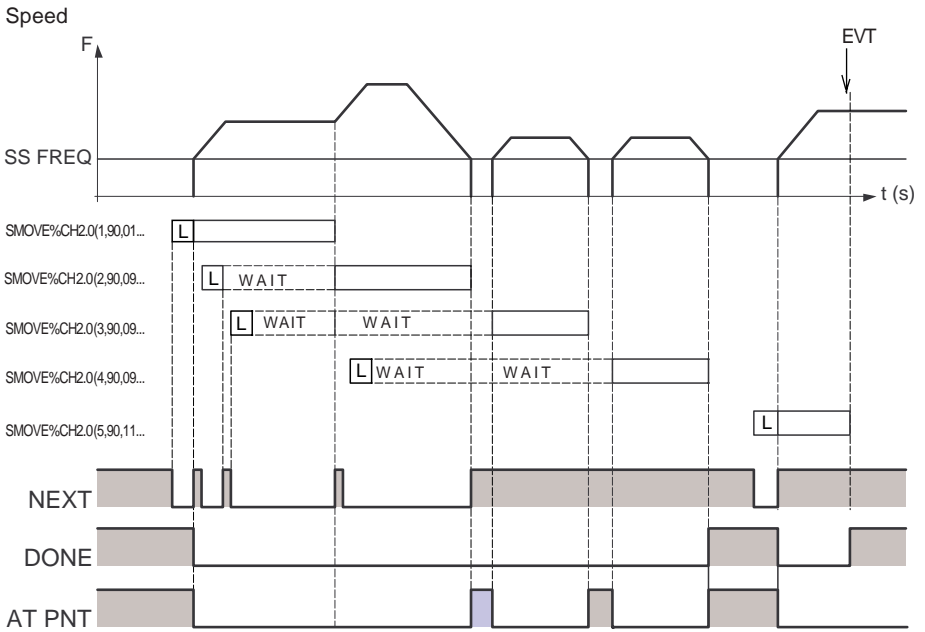
### Bits associated with the sequencing mechanism

Addressing	Description
NEXT %lxy.i.0	indicates to the user program that the module is ready to receive the next movement command.
DONE %lxy.i.1	indicates the end of execution of the current command and that there is no new command in the buffer memory.
AT_PNT %lxy.i.8	signals that the moving part has reached the intended point : <ul style="list-style-type: none"> <li>• for a movement without stop, at 0,</li> <li>• for a movement with stop, equal to NOMOTION if the remainder is zero.</li> </ul>

**Example** : to be sure that a command has been executed and that the moving part has reached the intended point.


 SMOVE %CH6.0(2,90,9,100000,2000,0)  
 %l6.0.1 AND %l6.0.8

**Example :**



Note: L means LAUNCH.

For a movement with stop : DONE changes to 1 when NOMOTION changes to 1 and when the buffer memory is empty.

For a movement without stop : DONE changes to 1 when the target position is exceeded and the buffer memory is empty.

---

## 5.4 Programming in automatic mode : other functions

---

C

### 5.4-1 Deferred "PAUSE" function

The PAUSE (%Qxy.i.12) command suspends the sequence of movements. It only becomes active when the moving part is stationary, say at the end of a G09 or G10 instruction.

The next movement starts as soon as the PAUSE command is reset to 0.

Bit ON\_PAUSE (%Ixy.i.26) signals, when it is at 1, that the axis is in "PAUSE" state.

This function has 2 possible uses :

- block by block execution of the movement program,
- synchronization of the axes on a stepper motor axis control module.

#### Block by block execution of the movement program

Activating the **Pause** command in the debug screen in automatic mode, or setting bit PAUSE (Qxy.i.12) to 1, causes the axis to change to waiting status after execution of the current instruction : the sequence of movements stops.

It is therefore possible to execute movements block by block for debugging purposes by successively activating and deactivating the Pause command.

#### Synchronization of several axes

Setting bit PAUSE (%Qxy.i.12) to 1 via the program for each axis causes the axis to change to waiting status after execution of the current instruction.

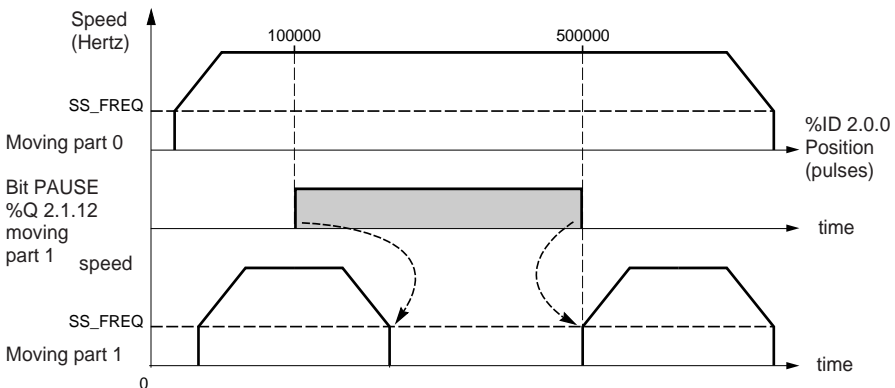
When the PAUSE bit is reset to 0, the module continues to execute the instructions.

**Example:** Execution of the movement of moving part 1 (CFY 21 module slot 2, channel 1) is stopped when moving part 0 (CFY 21 module slot 2, channel 0) reaches position 100000. It is restarted as soon as moving part 0 reaches position 500000.

```
IF (%ID2.0>=100000) THEN SET %Q2.1.12;
```

```
.. (%ID2.0: variable POS channel 0, %Q2.1.12: variable PAUSE channel 1) . .
```

```
IF (%ID2.0>=500000) THEN RESET %Q2.1.12;
```



**Note :** The PAUSE command is only processed when AUTO mode is active.

---



**5.4-2 Immediate "PAUSE" function**

This function is used to stop the moving part in automatic mode, while still ensuring that on the command to restart the movement the programmed trajectory is followed (with no risk of the command being refused).

This command is activated :

- via the program : by assigning the value 0 to the speed correction coefficient word CMV (%QWxy.i.1),
- via the debug screen : by assigning the value 0 to the speed correction coefficient parameter CMV.

It causes the moving part to stop according to the programmed deceleration. The pause status report is signaled by bit IM\_PAUSE (%Ixy.i.27).

This command is deactivated :

- via the program : by reassigning the initial value (>0) to the speed correction coefficient word CMV,
- via the debug screen : by reassigning the initial value (>0) to the speed correction coefficient parameter CMV.

It causes the interrupted movement to restart at the speed corresponding to :  $F \cdot CMV / 1000$

Example 1 : (TSX CFY 11 module slot 2 channel 0)

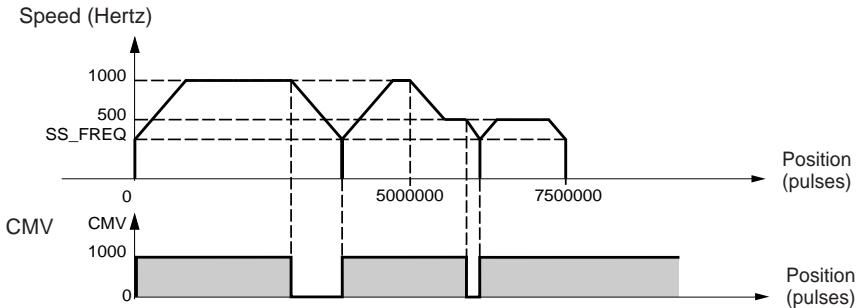
SMOVE %CH2.0 (1,90,10,5000000,1000,0);

SMOVE %CH2.0 (2,90,09,7500000,500,0);

.....

IF RE %M10 THEN %MW100:=%QW2.0.1;%QW2.0.1:=0;

IF FE %M10 THEN %QW2.0.1:=%MW100;



**Notes :**

- this command is deactivated at a STOP command or blocking fault,
- if the target position has been reached when movement is stopped as a result of an immediate pause command, the current movement is considered to be finished. In this case the trajectory is restarted with the next movement waiting in the buffer memory.

### 5.4-3 Event processing

The channels of TSX CFY modules can activate an event-triggered task. For this the function must be enabled in the configuration screen and the event processing number associated with the channel must be defined (see section 4.3-9).

#### Activating an event-triggered task

The following instructions start the transmission of an event which activates the event-triggered task :

- move until an event is detected, codes **10** and **11** : the application event processing is activated when the event is detected,
- await an event, code **05** : the application event processing is activated at the end of the instruction,
- memorize the current position when an event occurs, code **07** : the application event processing is activated at the end of the memorization of position PREF.

The application event processing is activated if four-bit byte no. 3 of parameter M of the SMOVE function associated with the instruction is 1 (see section 5.3-2).

#### Variables which can be used by the event-triggered task

- if several event sources are chosen the following bits are used to determine what triggered the application event processing :
  - EVT\_G07 (%Ixy.i.37) memorization of position,
  - EVT\_G05 (%Ixy.i.38) end of G05 on event,
  - TO\_G05 (%Ixy.i.39) time period of G05 elapsed,
  - EVT\_G1X (%Ixy.i.40) end of G10 or G11 on event,
- bit OVR\_EVT (%Ixy.i.36) this bit detects a delay in the transmission of the event or loss of the event.
- value of memorized position PREF (%IDxy.i.7).

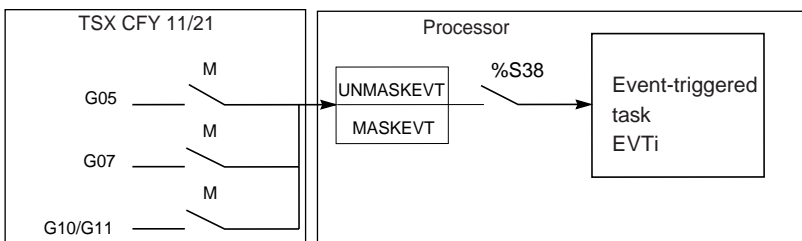
**Note** : this data is only updated when the task is activated.

#### Masking of events

PL7 language offers 2 ways of masking events :

- MASKEVT instruction for global masking of events (instruction UNMASKEVT unmask them),
- bit %S38 : 0 = global inhibition of events (the bit is normally at 1).

#### Summary diagram





---

## 5.5 Managing the operating modes

---

### Switching on the module

When the TSX CFY module is switched on or connected it performs a number of self-tests with the outputs in safety position (outputs at 0).

At the end of the self-tests :

- if the self-tests have detected no errors : the module tests the configuration with the outputs in safety position, and if the configuration is correct, the module changes to inhibiting mode (OFF),
- if the self-tests have detected any errors or if the configuration is incorrect, the module signals a fault and maintains the outputs in safety position.

### PLC in RUN

All the operating modes of the configured channels can be used.

### PLC changes from RUN to STOP (or Processor $\leftrightarrow$ module communication is lost) :

The moving part decelerates and stops, and the module changes to stop mode (OFF). The axis changes to non-referenced state.

Reminder : bit %S13 detects when the PLC changes to STOP. It is set to 1 during the first scan after the PLC is set to RUN.

### Changing the configuration (reconfiguration)

- the moving part decelerates and stops,
- the channel is deconfigured,
- the channel tests the new configuration with the outputs in safety position,
- if the new configuration is correct, the channel changes to stop mode (OFF),
- if the configuration is incorrect, the module signals a fault and maintains the outputs in safety position.

### Power outage and return

The moving part stops at a power outage.

On a cold start (%S0) or warm restart (%S1), the channel configuration is automatically transmitted by the processor to the module, which changes to stop mode (OFF).

---

## 5.6 Fault management

---

### 5.6-1 Role

Checking for faults is of primary importance in the area of position control because of the inherent risks when moving parts are in motion.

The checks are performed internally and automatically by the module.

4 types of fault are detected :

- module faults. These are hardware faults in the module. All the axes controlled by the module are therefore affected when this type of fault occurs. They may be detected during the self-tests (when the module is reinitialized) or during normal operation (I/O fault).
- hardware faults external to the module (eg : brake output short circuit)
- application faults linked to the axes (eg : soft stop overshoot).  
Continuous checks are made for axis level faults when the axis is configured.
- command failures. These are faults which may occur during the execution of a movement command, the transfer of the configuration, the transfer of adjustment parameters or a change of operating mode.

Note :

- checking for certain axis level faults can be enabled or inhibited by the error control parameters of the axis. These error control parameters can be adjusted in the adjustment screen.
- in stop mode (DIRDRV), the application fault check is inhibited.

---

### 5.6-2 Principle

When a fault is detected, the following occur :

- fault indication,
- deceleration of the moving part to stop,
- disabling of the translator, activation of the brake,
- deletion of all the commands which have been memorized,
- wait for acknowledgment.

The fault must have disappeared and been acknowledged before the application can be restarted.

**Note:**

1) if the emergency stop input is opened or the translator is not enabled : (%Qxy.i.10 = 0), the deceleration phase is not executed and the moving part stops immediately.

2) appearance of a **loss of step** message is not considered as a fault, but simply signalled to the application.



### 5.6-3 Programming

Faults can be displayed, corrected and acknowledged from the debug screen, but it may be more useful during operation to be able to control the moving part and correct faults from an operator terminal. The application has all the necessary data and commands for this purpose.

#### Fault indication

The module provides a large amount of data in the form of status bits and words which can be accessed via the PL7 program. These bits make it possible to deal with faults hierarchically :

- so that they can be used in the main program,
- to indicate the fault.

There are 2 levels of indication :

<p><b>• 1st level : general data</b></p> <p>%Ixy.i.ERR : module fault</p> <p><b>AX_OK</b> : (%Ixy.i.3) no blocking fault (with stopping of the moving part) is detected</p> <p><b>AX_FLT</b>: (%Ixy.i.2) fault (covers all faults)</p> <p><b>HD_ERR</b> : (%Ixy.i.4) external hardware fault</p> <p><b>AX_ERR</b> : (%Ixy.i.5) application fault</p> <p><b>CMD_NOK</b> : (%Ixy.i.6) command failure</p>
<p><b>• 2nd level : detailed information</b></p> <p>Module and axis fault status words (%MWxy.i.2 and %MWxy.i.3)</p>

Each fault is described in detail in the following sections. Status words are also described in detail in the quick reference guide.

In general it is advisable to stop the evolution of the sequential processing assigned to the axis when a blocking fault occurs and to control the moving part in manual mode while the fault is corrected. Correction of the fault should be followed by an acknowledgment.

#### Fault acknowledgment

When a fault occurs :

- fault bits AX\_FLT, AD\_ERR, AX\_ERR, CMD\_NOK and bits extracted from the status words concerned by the fault change to 1.
- if the fault is with stop, bit AX\_OK changes to 0.

When the fault disappears, the status of all the fault bits remains unchanged. The fault is memorized until it is acknowledged : bit ACK\_DEF %Qxy.i.9 is set to 1 (or the module is reinitialized). The fault must be acknowledged after it has disappeared (except in the case of soft stop faults).

If several faults are detected, the acknowledgment command only applies to those faults which have totally disappeared. Faults which remain must be acknowledged again when they have disappeared.

**Note** : a fault can also be acknowledged when the PLC is initialized, or when a new, correct command is accepted in the case of a command failure.

### 5.6-4 Summary table

The following table summarizes the various types of fault and the associated bits.

Module faults	Process faults bit <b>AX_FLT</b> %lxy.i.2		
	<b>AX_OK</b> %lxy.i.3 (no blocking fault has been detected)		
bit %lxy.i.ERR	External hardware bit <b>HD_ERR</b> %lxy.i.4	Application bit <b>AX_ERR</b> %lxy.i.5	Command failure bit <b>CMD_NOK</b> %lxy.i.6
Internal fault Communication fault Configuration or adjustment fault	<ul style="list-style-type: none"> <li>• Emergency stop</li> <li>• Translator</li> <li>• 24 volt supply</li> <li>• Brake output short-circuit</li> </ul>	<ul style="list-style-type: none"> <li>• Soft stops</li> </ul>	Fault coded in word <b>CMD_FLT</b> %MWxy.i.7

### 5.6-5 Description of module faults

Bit %lxy.i.ERR covers all module faults :

- internal fault MOD\_FLT (%MWxy.i.2:X4) = module missing, off or performing self-tests,
- communication fault COM\_FLT (%MWxy.i.2:X6) = communication fault with the processor,
- configuration fault CONF\_FLT (%MWxy.i.2:X5) = fault between the declared position of the module in the configuration and the actual position.

**Note** : %MW words require a READ\_STS command to be updated.

### 5.6-6 Description of external hardware faults

These faults are indicated by bit **HD\_ERR** (%Ixy.i.4)

#### Fault : Emergency stop

<b>Cause</b>	Open circuit between the +24V and Emergency stop inputs.
<b>Parameter</b>	None
<b>Consequence</b>	The axis is not referenced and the moving part is forced to stop.
<b>Indication</b>	Bit EMG_STOP (%Ixy.i.29) and bit EMG_STP (%MWxy.i.3:X5) (1)
<b>Remedy</b>	Reconnect the 2 inputs and acknowledge the fault.

#### Fault : 24 volt supply

<b>Cause</b>	The 24 volt channel input is no longer powered.
<b>Parameter</b>	None
<b>Consequence</b>	The axis is not referenced and the moving part is forced to stop.
<b>Indication</b>	Bit AUX_SUP (%MWxy.i.3:X6) (1)
<b>Remedy</b>	Reconnect the supply and acknowledge the fault.

#### Fault : Brake output short-circuit

<b>Cause</b>	Short-circuit detected on the module brake output.
<b>Parameter</b>	None
<b>Consequence</b>	The axis is not referenced and the moving part is forced to stop.
<b>Indication</b>	Bit BRAKE_FLT (%MWxy.i.3:X1) (1)
<b>Remedy</b>	Correct the short-circuit and acknowledge the fault.

#### Fault : Translator

<b>Cause</b>	The translator control input is not receiving the "translator OK" level defined during channel configuration.
<b>Parameter</b>	None
<b>Consequence</b>	The axis is not referenced and the moving part is forced to stop.
<b>Indication</b>	Bit DRIVE_FLT (%MWxy.i.3:X2) (1)
<b>Remedy</b>	Correct and acknowledge the fault.

(1) %MW words require a READ\_STS command in order to be updated.

### 5.6-7 Description of application faults

These faults are indicated by bit **AX\_ERR** (%lxy.i.5). The parameters can be accessed via the configuration editor adjustment screen.

#### Fault : **Soft stops**

<b>Cause</b>	The moving part is no longer located between the 2 limit values : upper and lower soft stop limits (this check is activated as soon as the axis is referenced).
<b>Parameter</b>	Upper soft stop limit : SLMAX (%MDxy.i.14) Lower soft stop limit : SLMIN (%MDxy.i.16)
<b>Consequence</b>	The moving part is forced to stop.
<b>Indication</b>	<ul style="list-style-type: none"> <li>• bit %MWxy.i.3:X3 upper soft stop overshoot</li> <li>• bit %MWxy.i.3:X4 lower soft stop overshoot</li> </ul>
<b>Remedy</b>	<p>Acknowledge the fault and return the moving part from outside the soft stop limits to within the valid measurement area in manual mode. To do this, check that :</p> <ul style="list-style-type: none"> <li>• there is no movement in progress</li> <li>• manual mode has been selected</li> <li>• the STOP command is at zero</li> <li>• the axis to which this command applies is referenced</li> <li>• there is no other fault with stop on the axis.</li> </ul>



**5.6-8 Description of command failure faults**

These faults are indicated by the Cmd Fail indicator lamp in the debug screens. The source of the command failure can be ascertained by pressing the DIAG key at channel level. They can also be accessed via the program using bit CMD\_NOK (%Ixy.i.6) and word CMD\_FLT (%MWxy.i.7).

**Fault : Command failure**

<b>Cause</b>	<ul style="list-style-type: none"> <li>• unauthorized movement command,</li> <li>• incorrect configuration or parameter transfer.</li> </ul>
<b>Parameter</b>	-
<b>Consequence</b>	<ul style="list-style-type: none"> <li>• immediate stop of current movement,</li> <li>• resetting of the buffer memory receiving the movement commands in automatic mode.</li> </ul>
<b>Indication</b>	<ul style="list-style-type: none"> <li>• bit CMD_NOK (%Ixy.i.6) : movement command failure,</li> <li>• word CMD_FLT (%MWx.y.i.7) : word coding the type of fault detected.</li> </ul> <div style="text-align: center;"> <p>CMD_FLT</p> <p>Byte      MSB    LSB</p> </div> <p>More detailed information is given in section 14</p>
<b>Remedy</b>	<ul style="list-style-type: none"> <li>• implicit acknowledgment on reception of a new accepted command,</li> <li>• acknowledgment is also possible using the command ACK_DEF (%Qxy.i.9).</li> </ul>

**Nota :** in the case of linked movements in automatic mode, it is advisable to make the execution of each movement conditional on the end of execution of the previous movement, using bit AX\_FLT (%Ixy.i.2). This ensures that a link is not made to the following command when the current command has failed.

## 5.7 Management of manual mode (MAN)

Manual mode can be selected and controlled from the application-specific debug screen (see section 7.2-6) or, using the application program, from a front panel, man-machine interface terminal or supervision terminal.

In this case, the dialog is programmed in Ladder, Instruction List or Structured Text language, using elementary commands (movements, reference point, etc).

### 5.7-1 Selecting manual mode

This is performed by assigning the value 2 to word `MODE_SEL` (%QWxy.i.0).

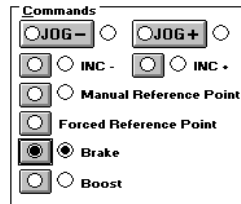
The change from the current mode to manual mode is made as soon as the moving part stops.

When the manual mode command is recognized, bit `IN_MANU` (%Ixy.i.18) is set to 1.

### 5.7-2 Execution of manual commands

The following elementary commands can be accessed via command bits %Qxy.i.j associated with manual mode :

- visual move `JOG_P` (%Qxy.i.1) and `JOG_M` (%Qxy.i.2),
- incremental move `INC_P` (%Qxy.i.3) and `INC_M` (%Qxy.i.4),
- manual reference point `SET_RP` (%Qxy.i.5),
- forced reference point `RP_HERE` (%Qxy.i.6),



These commands are equivalent to those which are accessed from the module debug screen.

#### General conditions for executing commands in manual mode :

- target position within the soft stop limits (1),
- axis with no blocking fault (bit `AX_OK` : %Ixy.i.3 = 1),
- no command being executed (bit `DONE` : %Ixy.i.1 = 1),
- command `STOP` (%Qxy.i.8) inactive and translator relay enable bit `ENABLE` (%Qxy.i.10) at 1.
- if the moving part is outside the limits : the required direction of movement is the direction to return between the limits.

#### A movement can be stopped by :

- appearance of the `STOP` command (%Qxy.i.8), `ENABLE` bit (%Qxy.i.10) at 0, or `STOP` input,
- occurrence of a blocking fault,
- change of operating mode,
- reception of a configuration.
- reaching a positive (negative) limit switch when moving in a positive (negative) direction.

(1) except, in the event of a soft stop fault, for the `JOG_P` or `JOG_M` release command, after acknowledgment of the fault.





**5.7-3 Detailed description of manual commands**

**Visual movement command : JOG\_P and JOG\_M**

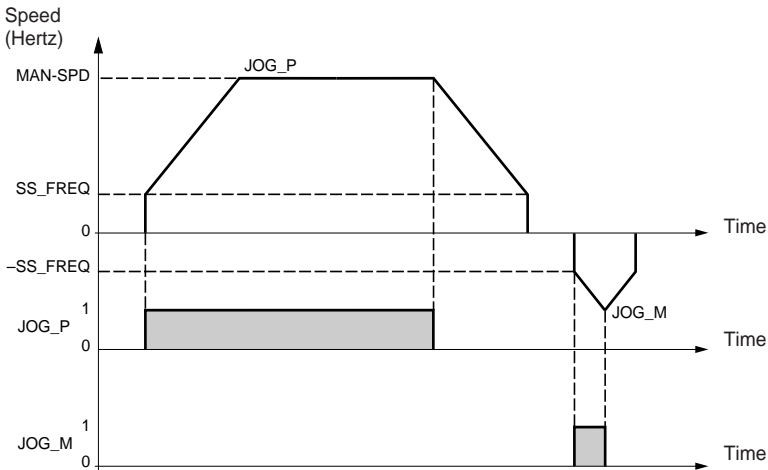
Bits JOG\_P (%Qxy.i.1) and JOG\_M (%Qxy.i.2) control the movement of the moving part in a positive or negative direction. The operator should visually follow the position of the moving part. The movement continues as long as the command is present and is not inhibited by a STOP command or a fault.

Commands JOG\_P and JOG\_M are taken into account on an edge and maintained active on a state, whether the axis is referenced or not.

The movement is performed at the speed of manual mode MAN\_SPD defined in the adjustment screen (or in double word %MDxy.i.20).

The speed can be adjusted while moving using coefficient CMV (%QWxy.i.1).

Any operating speed above FMAX (maximum speed of the axis defined at configuration) is peak limited to value FMAX.



**Note :**

- these commands are also used to release the moving part when a soft stop fault is detected and has been acknowledged.
- if bit JOG\_P or M is at 1 when changing to manual mode, this command is not taken into account. It will only be taken into account after the bit changes to 0 and is reset to 1.

## Incremental movement command : INC\_P and INC\_M

Bits INC\_P (%Qxy.i.3) and INC\_M (%Qxy.i.4) control the movement of the moving part by one position increment in a positive or negative direction.

The value of the position increment PARAM is entered in the module debug screen (or in %QDxy.i.2).

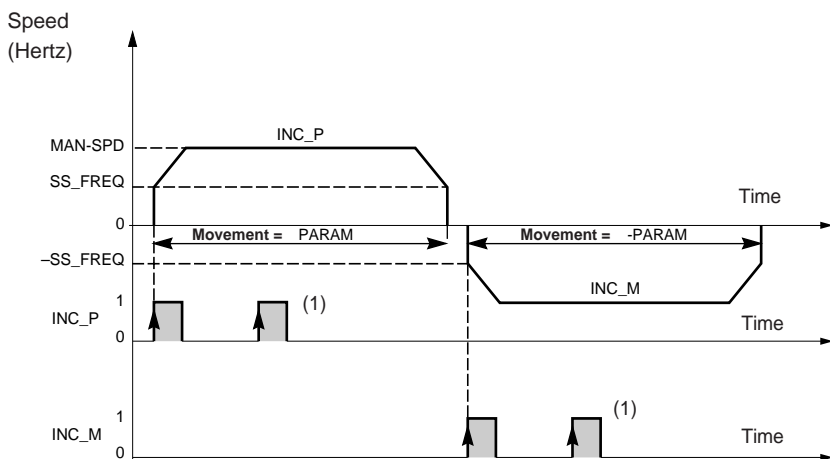
In addition to the general conditions for execution in manual mode, commands INC\_P and INC\_M are active on a rising edge when :

- the axis is referenced
- the target position is within the soft stop limits.

The movement is performed at the speed of manual mode MAN\_SPD defined in the adjustment screen (or in double word %MDxy.i.20).

The speed can be adjusted while moving using coefficient CMV (%QWxy.i.1).

Any operating speed above FMAX (maximum speed of the axis defined at configuration) is peak limited to value FMAX.



(1) Commands not taken into account (command failure indicated).

**Note :** the direction of movement is fixed by the INC\_P or INC\_M command irrespective of the sign of the value contained in PARAM.



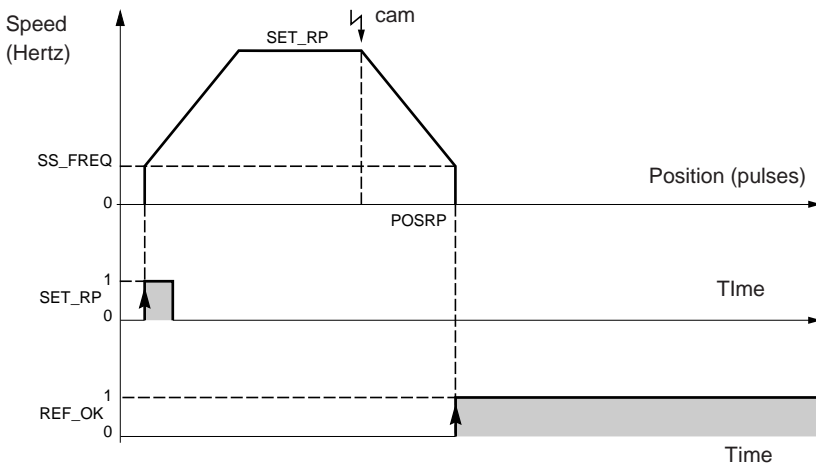
**Reference point command : SET\_RP**

Bit SET\_RP (%Qxy.i.5) sets a manual reference point and moves the part.

The type and direction of the reference point are defined during configuration in the Reference point parameter (see section 4). The reference point value is defined in the parameter RP Value in the adjustment screen (or double word RP\_POS in %MDxy.i.22). The approach speed is the manual speed MAN\_SPD defined in the adjustment screen (or in double word %MDxy.i.20) multiplied by the speed correction coefficient CMV. The reference point speed varies according to the type of reference point selected.

Any operating speed above FMAX (maximum speed of the axis defined during configuration) is peak limited to value FMAX.

Example : short cam only and + direction



**Forced reference point command : RP\_HERE**

Bit RP\_HERE (%Qxy.i.6) forces a reference point without moving the part to the value defined in parameter PARAM entered in the TSX CFY 11/21 module debug screen (or in %QDxy.i.2).

This command references the axis without moving the part.

**Notes :**

- command RP\_HERE does not modify the value of parameter RP\_POS.
- the value of parameter PARAM must be within the soft stop limits.
- no blocking faults are tolerated during execution of this command.

---

## 5.8 Managing direct drive mode (DIRDRIVE)

---

### 5.8-1 Selecting direct drive

This is performed by assigning value 1 to word MOD\_SEL %QWxy.i.0

The change from the current mode to direct mode is only made when the moving part is stopped.

When the direct drive command is taken into account, bit IN\_DIRDR (%Ix.y.i.17) is set to 1.

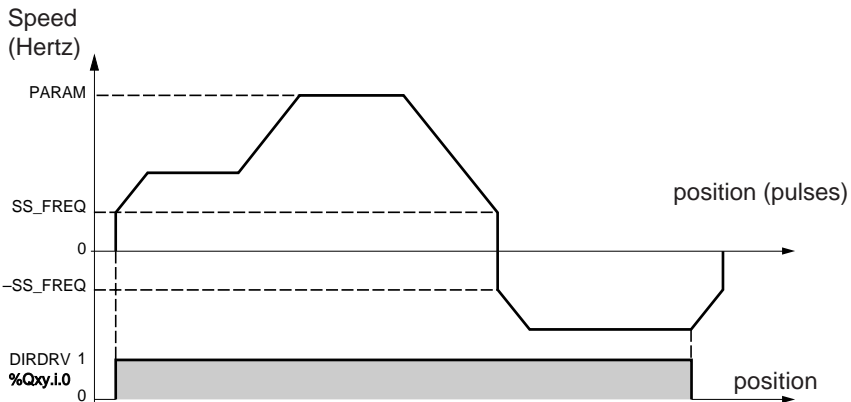
---

### 5.8-2 Executing commands in direct drive mode

Direct drive mode has a movement command DIRDRV %Qxy.i.0. This mode controls the movement of the moving part, the speed setpoint is given by variable PARAM %QDxy.i.2. The sign of this variable gives the direction.

The speed of the translator is controlled between SS\_FREQ and FMAX. These values are defined in the configuration screen (FMAX) and the adjustment screen (SS\_FREQ).

Status bit ST\_DIRDR (%Ix.y.i.20) indicates that a movement is in progress in DIRDRIVE mode.



When the reference changes, the output reaches the new reference according to a trapezoid speed profile respecting the configured acceleration.

If the new reference requires a change of direction, the moving part is stopped respecting the deceleration, then the new reference is applied, respecting acceleration.

**General conditions for executing the DIRDRIVE command :**

- axis with no blocking fault (bit AX\_OK : %lxy.i.3 = 1),
- command STOP (%Qxy.i.8) inactive and translator enable bit ENABLE (%Qxy.i.10) at 1,
- parameter PARAM = speed in the interval [- FMAX, - SS\_FREQ], or [SS\_FREQ, FMAX] of the selected axis.
- if the moving part is outside the limits : the required direction of movement is the direction to return between the limits.

**A movement can be stopped by the :**

- activation of the STOP command, or translator ENABLE bit (%Qxy.i.10) at 0,
- occurrence of a blocking fault or a soft stop fault,
- change of operating mode,
- reception of a configuration.
- reaching a positive (negative) limit switch when moving in a positive (negative) direction.

(1) Control of limit switches remains active if the axis has been referenced beforehand.

To check this command : force the loss of the axis reference by a temporary disabling ENABLE (%Qxy.i.10)=0 then=1 or by pressing the enable button.

**5.9 Management of stop mode (OFF)**

This mode is usually used in debug mode from the configuration editor. It can, however, be controlled by the program.

Stop mode is selected by assigning value 0 to word MOD\_SEL %QWxy.i.0. It is also the mode selected by the module when the PLC is in STOP, and the default mode following channel configuration.

Stop (OFF) mode has no associated movement command.

In this mode, the axis control module only feeds back information on the current position (%lDxy.i.0) and speed (%lDxy.i.2).

The motion of the moving part is not controlled, and the checks for software faults are inhibited (except for the soft stop check).

The translator enable relay is unlocked, irrespective of the state of bit ENABLE (%Qxy.i.10).

The translator enable output is controlled by the ENABLE command (%Qxy.i.10).

---

## 5.10 Presymbolization

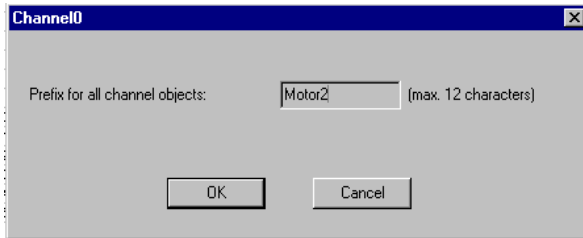
---

### Presymbolization procedure

Presymbolization is performed from the variables editor.

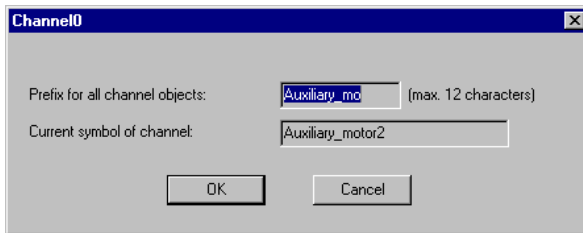
To do this : select the I/O module, then click on the letter "p" preceding the %CH identifier of the channel to be symbolized.

The following entry box is displayed, in order to enter the prefix for the symbolization of the objects in this channel.



A screenshot of a dialog box titled "Channel0". It contains a text input field with the text "Motor2" and a label "(max. 12 characters)". Below the input field are two buttons: "OK" and "Cancel".

**Note** : if an object is already symbolized, a dialog box is displayed so that the symbolization of that object may be modified if required.



A screenshot of a dialog box titled "Channel0". It contains two text input fields. The first field has the text "Auxiliary\_mo" and a label "(max. 12 characters)". The second field has the text "Auxiliary\_motor2". Below the input fields are two buttons: "OK" and "Cancel".

---

## 5.11 Transferring the program to the PLC

---

When the setup is complete, the PLC should be connected. The program which has been entered must be transferred to the PLC processor.

To do this, launch the **PLC/Transfer** command.

---

## 6.1 Operations prior to adjustment

---

### 6.1-1 Preliminary conditions

- TSX CFY module(s) installed in the PLC,
- axis control application(s) connected to the TSX CFY modules,
- terminal connected to the PLC via the terminal port or network,
- axis control configuration and program created and transferred to the PLC processor,
- PLC in RUN (axis control program inhibited), using a program execution condition bit for example, to facilitate adjustment operations.

---

### 6.1-2 Preliminary checks

- check the wiring,
- check that the movements can be safely performed,
- check that the limit switches are wired in accordance with safety regulations (these generally act directly on the translator power supply sequence),

---

### 6.1-3 Adjusting the translator

Adjust the translator in accordance with the manufacturer's instructions.



---

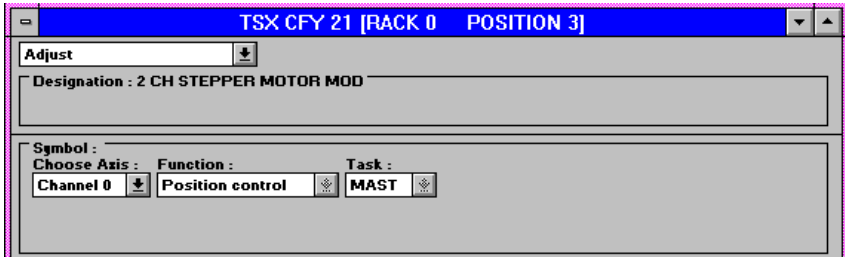
## 6.2 Adjusting the parameters

---


C

### 6.2-1 Access to the adjustment parameters

The adjustment parameters are accessed via the **Adjustment** command in the **View** menu in the TSX CFY module configuration screen (or by selecting Adjustment in the pulldown list in the module zone of the parameter or debug configuration screen).



Select the channel to be adjusted from the "Select axis" pulldown menu : channel 0 to 1.

The  button is used to display either the current or the initial parameters.

The **initial parameters** are :

- parameters entered (or defined by default when the configuration is validated, see section 4.4) in offline mode and provided when transferring the program to the PLC,
- parameters taken into account at the last reconfiguration in online mode.

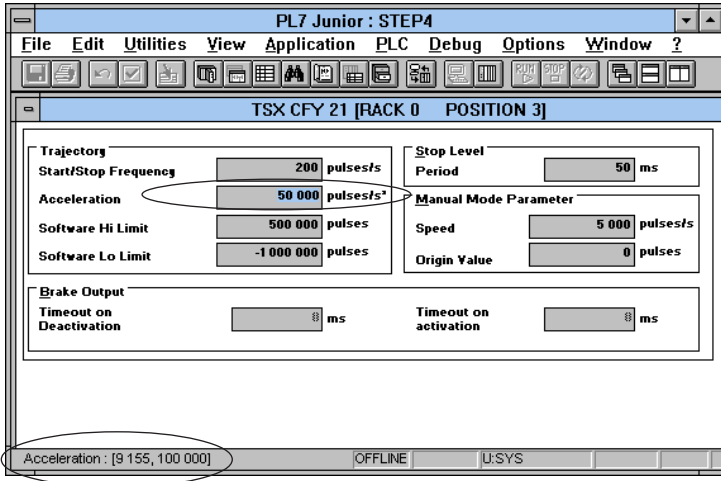
These cannot be modified from this screen. However, they can be updated from the current adjustment parameters (see section 6).

The **current parameters** are those modified and validated from the adjustment screen in online mode (or via the program, by explicit exchange). These parameters are replaced by the initial parameters on a cold restart.

It is essential that the parameters are saved following this adjustment parameter definition phase, see section 6.

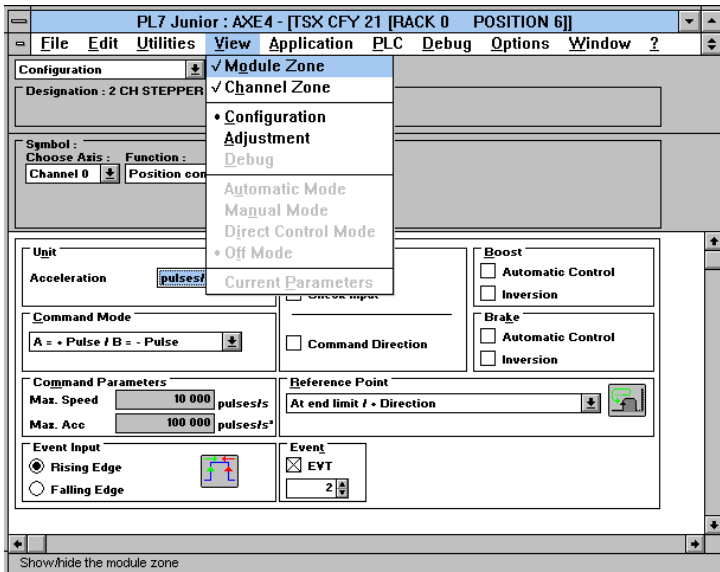


The lower part of the screen then displays the adjustment parameters.



**Note :** the limits of each parameter are displayed in the status bar.

To display the entire adjustment parameters zone, select the **View/Module Zone** and **View/Channel Zone** commands (to restore these zones, use the same commands).



## 6.2-2 Trajectory

These fields are used to define the start/stop frequency, acceleration and upper and lower soft stops of the axis.

Trajectory		
Start/Stop Frequency	100	pulses/s
Acceleration	3 155	pulses/s <sup>2</sup>
Software Hi Limit	10 000 000	pulses
Software Lo Limit	-10 000 000	pulses

- **Start/stop frequency (SS\_FREQ):** this is the minimum movement speed for the moving part.

If FMAX (**maximum speed** defined in configuration) is less than 4 KHz, the SS\_FREQ parameter must be between 0 and the FMAX value. Otherwise, (FMAX greater than 4 KHz), SS\_FREQ must be between 0 and 4 KHz.

If this parameter is not entered, the start/stop frequency is the smallest frequency in the range. See section 4 and section 10 for limitations.

- **Acceleration (ACC) :** this parameter represents either the acceleration and deceleration curve of the moving part, or the duration of the acceleration and deceleration of the moving part (see section 4.3-2).

This parameter must be :

- in the case of user units in **Hertz/s**: between the lower acceleration limit for the **max. speed** (see the two tables in section 4.3-4 on lower and upper acceleration limits) and the acceleration entered in configuration.
- in the case of user units in **ms**: between the **max. acceleration** value entered during configuration (which is a minimum time) and **5000 ms**.

### Soft stops (software Hi and Lo limits) :

- **Upper soft stop SLMAX :** this is the maximum movement position of the moving part in a positive direction.
- **Lower soft stop SLMIN :** this is the maximum movement position of the moving part in a negative direction.

The soft stops must respect the following inequality :

- lower soft stop  $\leq$  upper soft stop,
- $-16\ 777\ 216 \leq$  soft stop and soft stop  $\leq 16\ 777\ 215$ .

### Special case : zero soft stops

If the two soft stops SLMIN and SLMAX are zero, the soft stop check is not activated. Movement can be made over the whole range from - 16777216 to 16777215 (without however overshooting either of these limits).

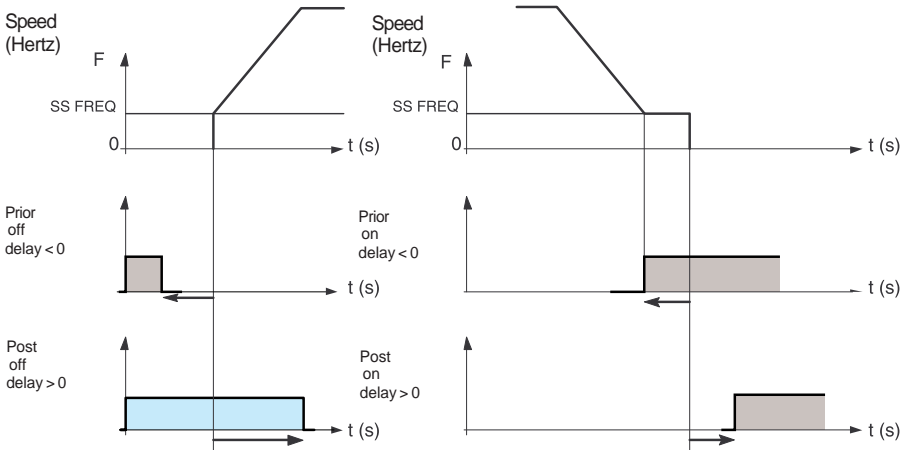
### 6.2-3 Brake output

During configuration, when automatic control of the brake output is selected (see section 4.3-8), the two parameters below are used to define the brake on and off delays.

<b>Brake Output</b>	
<b>Timeout on Deactivation</b>	<input type="text" value="5"/> ms
<b>Timeout on activation</b>	<input type="text" value="0"/> ms

- **Off delay** : this parameter is between -1000 and 1000 milliseconds.
- **On delay** : this parameter is between -1000 and 1000 milliseconds.

A negative parameter indicates prior activation of the brake command relative to the start or end of the movement ; a positive parameter indicates post activation.



## 6.2-4 Stop plateau

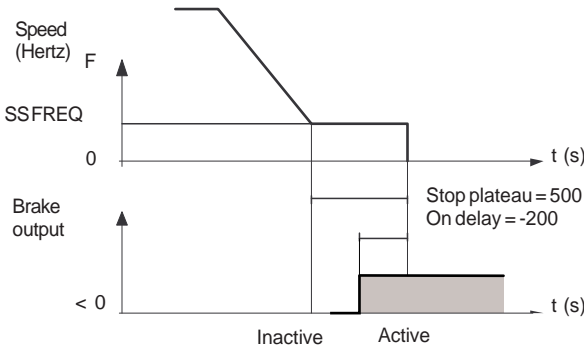
This data item is the duration of the **stop plateau** when the speed equals the start/stop speed (SS\_FREQ). This data item must be between **0 and 1000 ms**.



There is, however, a relation between the **stop plateau** and the brake on delay (when this delay is negative) if **automatic control** of the brake is configured (see section 4).

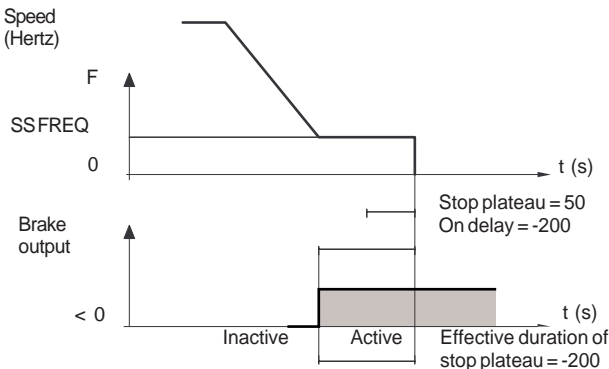
- **Duration of stop plateau > -(brake on delay):**

In this case the brake is activated after the speed has reached the start/stop speed (SS\_FREQ) of the moving part. The duration of the stop plateau will be that of the adjustment parameter.



- **Duration of stop plateau < -(brake on delay):**

In this case the duration of the stop plateau is forced to the duration of the brake on delay and the brake will be activated as soon as the speed has reached the start/stop speed (SS\_FREQ) of the moving part.



### 6.2-5 Manual mode parameters

The data to be entered relates to the control of the moving part in manual mode (MAN: see section 5.7).

Manual Mode Parameter	
Speed	300 pulses/s
Origin Value	10 000 pulses

- **Speed** : speed of movement (MAN\_SPD) of the moving part in manual mode :

The field value determines the speed of movement when the moving part is controlled in manual mode via instructions JOG-, JOG+, INC+, INC- and the speed of approach in SET\_RP, etc. The value of this field must be between the start/stop speed (SS\_FREQ) and the **maximum speed** (FMAX) defined during configuration (see section 4.3-4). As in automatic mode, the actual speed of movement is adjusted by the **correction coefficient** CMV.

- **Reference point** : value loaded in the current position when the reference point is set manually :

The value of this **reference point** (RP\_POS) field will be transferred to the momentary position (X\_POS) when the reference point is set manually and when axis control is in manual mode.


In general, the value of this field must be between SLMIN and SLMAX.

In the special case where SLMIN = SLMAX = 0, the value of this field must be between - 16 777 216 and 16 77 215. The default value is 0.

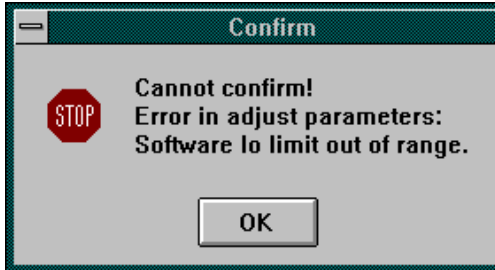
## 6.3 Confirming and saving adjustment parameters

C

### 6.3.1 Confirming

When the adjustment parameters have been entered, confirm these parameters using the **Edit/Confirm** command or activate the  icon.

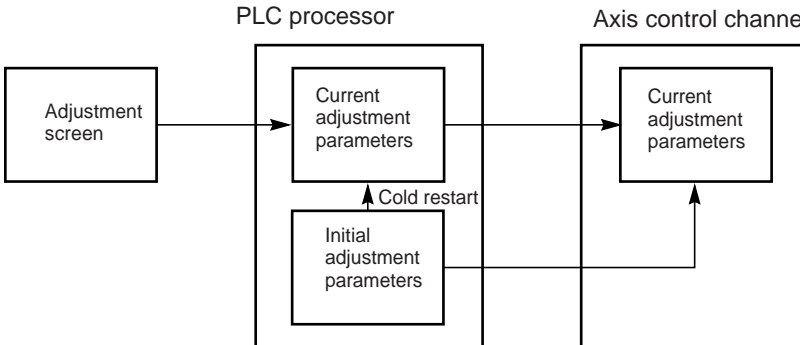
If one or more of the parameter values are not within permitted limits, an error message appears indicating the parameter concerned.



Correct the incorrect parameter(s) then confirm.

Modifying the adjustment parameters does not modify the operation in progress if none of the configuration parameters has been modified and if confirmation is via the adjustment screen.

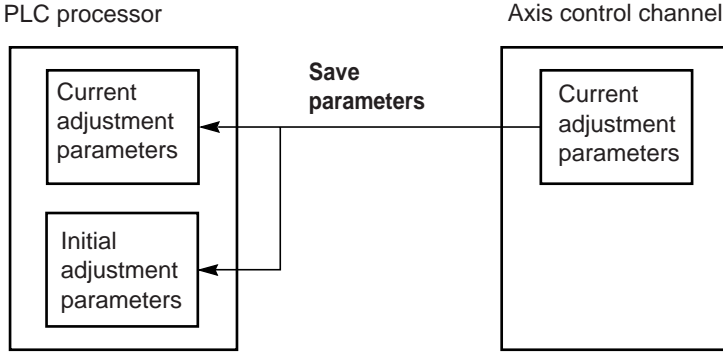
Current parameters are modified in this way (the initial parameters remain unchanged).



**Note** : on a cold restart the current parameters are replaced by the initial parameters. The initial parameters can be updated by the save command (see next page) or by a reconfiguration operation.

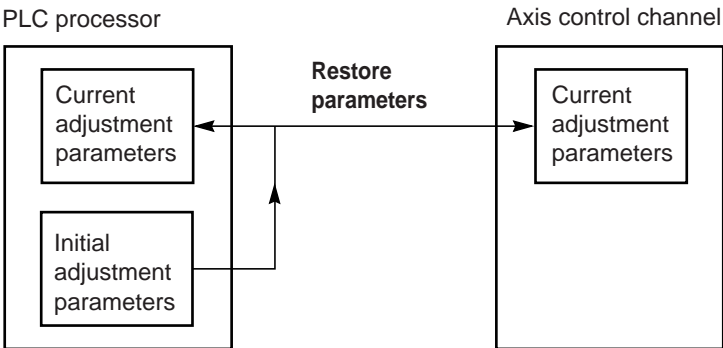
**6.3-2 Save**

To save the current parameters (update the initial parameters) activate the **Utilities/Save Parameters** command.




**6.3-3 Restore**

The **Utilities/Restore Parameters** command replaces the current parameters with the initial values.

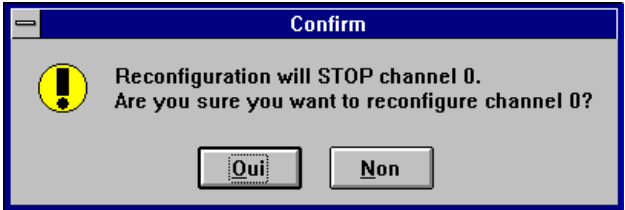


## 6.4 Reconfiguration in online mode

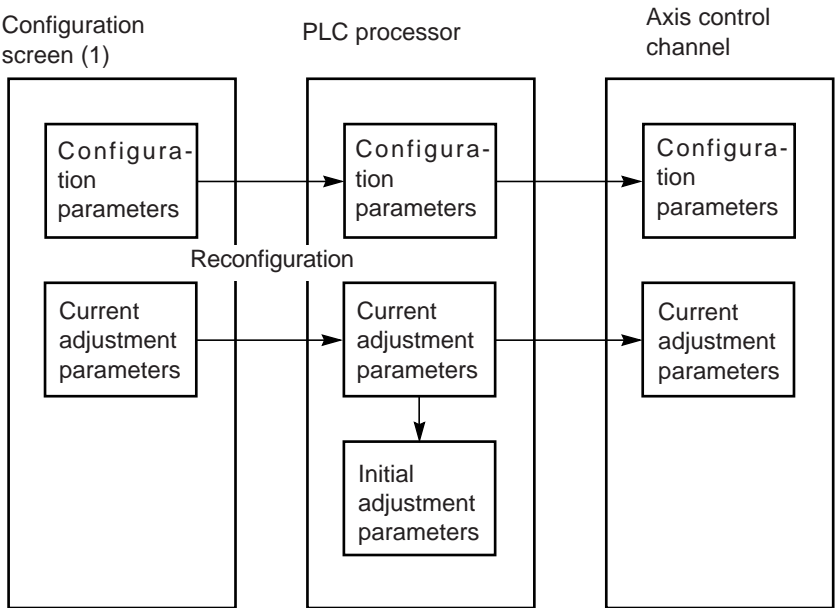
C

When the configuration parameters have been modified, confirm these parameters using the **Edit/Confirm** command or activate the  icon.

Only those parameters which are not grayed out can be modified in online mode. The other parameters must be modified in offline mode.



**Any reconfiguration in online mode stops the operation of the channel concerned, and thus the current movement.**



(1) or the adjustment screen if a configuration parameter has already been modified in the configuration screen



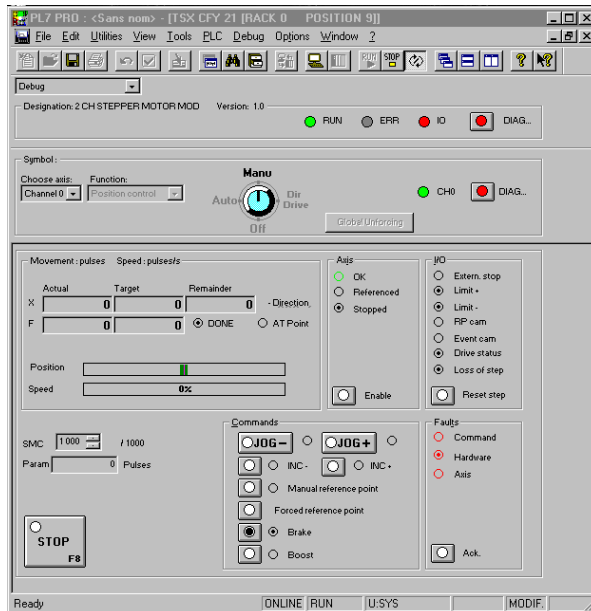
## 7.1 Principle of debugging an axis control program

Axis control, which is integrated in the PL7 program, uses the PL7 debug functions.

### Reminder of the options provided by PL7 :

- realtime display and animation of the program,
  - Grafcet : as each movement is programmed in one Grafcet step, it is easy to identify the current movement,
  - In Structured Text or Ladder language : highlighted display of the SMOVE function being executed,
- insertion of breakpoints and execution cycle by cycle, rung by rung or statement by statement,
- access to the animation tables, which enables status bits and words to be displayed and the command bits of the SMOVE function to be controlled. It also enables bit objects to be forced, and the evolution of the Grafcet chart to be blocked.

### TSX CFY axis control function debug screens



This screen comprises 3 zones :

- module zone,
- channel zone,
- control zone for controlling the moving part and the program. It depends on the operating mode selected via the mode switch : Automatic (AUTO), Manual (MAN), Direct (DIRDRIVE) or Stop (OFF).

---

## 7.2 Debug screens

---

### 7.2-1 Accessing the debug screens

The terminal must be in online mode.

Access the axis control module debug screens by selecting the **Configuration** editor and selecting and confirming (or double-clicking) the position in the rack containing the axis control module.

In online mode, the debug screen is selected by default.


---

### 7.2-2 User interface

**Command buttons :** 

- For commands on a state (except for JOG commands),
- Pressing the button activates the associated command and the indicator lamp in the button comes on when this command is accepted (the corresponding command bit %Q is set to 1),
- Pressing the button a second time deactivates the command, and the indicator lamp in the button goes off when this command is accepted (the corresponding command bit %Q is set to 0),
- For commands on an edge, the command is activated as soon as the button is pressed and then released. The indicator lamp in the button comes on and then goes off automatically.

**Entry field :**

All values entered in entry fields must be confirmed with the  key.

- The indicator lamp beside the button corresponds to the command being accepted by the module.

**Keyboard :**

**Shift F2 :** moves from one zone to another.

**Tab :** moves from one set of commands to another within one zone.


**Arrow keys :** moves from one command to another within one set of commands.

**Space key :** activates or deactivates a command.

#### **Warning**

There could be "conflicts" between the PL7 program which executes commands or writes variables, and the commands executed from the debug screen. The last command to be accepted takes priority.

**Note**

The **Utilities/Stop Animation** command or the  icon stops the animation in the display zones and inhibits the command buttons.

The **Utilities/Animate** command or the  icon reactivates the animation.

**7.2-3 Description of the debug screens**

The debug screens have a common header, comprising module and channel zones.


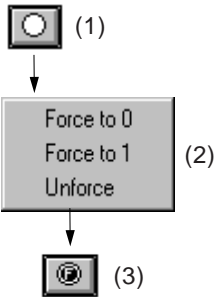


**Module zone**

Indicator lamps	Status	Meaning
RUN	on	module operating
ERR	on flashing	module off communication fault with the processor
I/O	on	process fault (bit AX_FLT %Ixy.i.2 see section 5)
DIAG	on	fault on the module : pressing this button (a button is associated with the indicator lamp) displays a module diagnostics dialog box which gives the source of the fault (see section 7 Diagnostics).

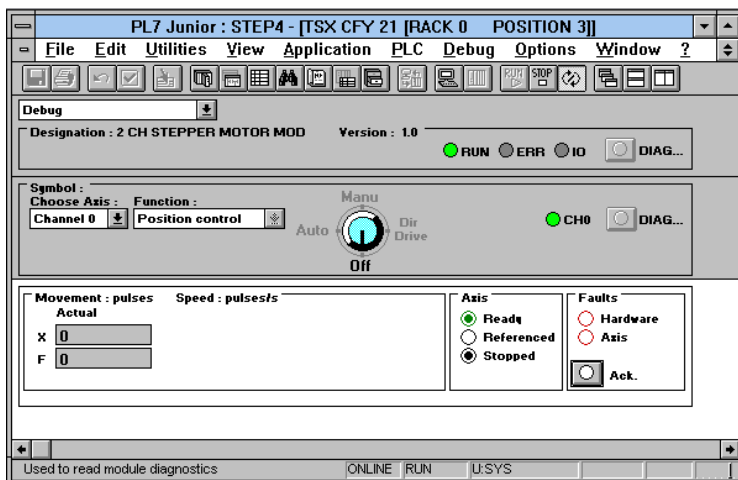
## Channel zone

In addition to the "Select Axis" and "Function" fields (common to all screens), this zone contains the following commands and indicator lamps :

Command	Role
	<p>Operating mode selection button. To access another mode click on the text of the mode you wish to select (or click as many times as necessary on the button). Using the keyboard : select the button using the <b>Tab</b> key and press as many times as necessary on the <b>Space</b> bar.</p> <p>Modes can also be accessed from the <b>View</b> menu. When the selected mode is accepted by the module, the movement control zone in the required mode is displayed.</p> <p><b>Caution</b> : although it has been selected, the chosen mode may not be accepted by the module channel (for example if the PLC is stopped).</p>
	<p>Forcing command menu. If an object can be forced, a click with the right-hand mouse button on the corresponding button (1) displays a menu (2) which accesses the forcing commands : <b>Force to 0</b>, <b>Force to 1</b> or <b>Unforce</b>.</p> <p>After selecting the command, by clicking on it, forcing is applied and the forcing status is displayed on the button (3) :</p> <ul style="list-style-type: none"> <li>• F for forcing to 0,</li> <li>• F reverse video for forcing to 1,</li> </ul> <p>The <b>Global Unforcing</b> button in the module zone unforces all the forced objects.</p>
<p><b>Chi</b></p>	<p>on axis (channel) flashing axis fault off axis not configured</p>
<p><b>DIAG</b></p>	<p>on</p> <p>channel fault : pressing this button (a button is associated with the indicator lamp) displays a channel diagnostics dialog box which gives the source of the fault (see section 7 Diagnostics).</p>

### 7.2-4 Stop mode (Off)

In this mode the axis control channel only feeds back position and speed data. The movement of the moving part is not controlled by the channel. The translator enable output is still controlled by the ENABLE command (%Qxy.i.10).



### Description of the information displayed

Movement :/Speed :

<b>X</b>	displays the position of the moving part in number of pulses
<b>F</b>	displays the speed of the moving part in Hertz

Axis

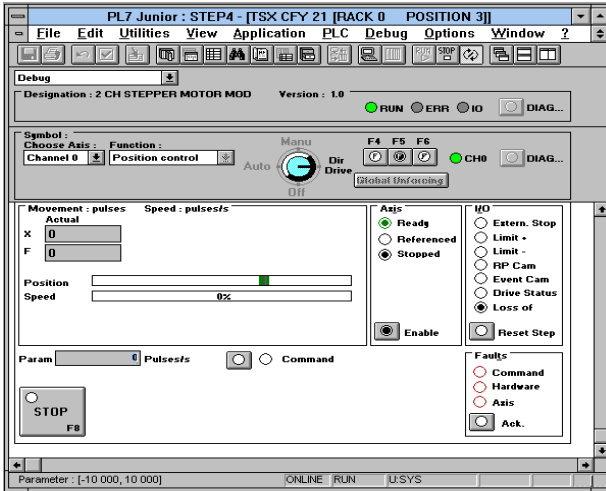
Indic. lamps	Status	Meaning
<b>OK</b>	on	axis in operating state (no blocking fault))
<b>Referenced</b>	on	axis referenced
<b>Stopped</b>	on	the moving part is stopped

Faults

Indic. lamps Buttons	Status	Meaning
<b>Hardware</b>	on	external hardware fault (translator, brake short-circuit, etc)
<b>Axis</b>	on	application fault (soft stops)
<b>Ack</b>		fault acknowledgment button (all faults which have disappeared are acknowledged).

## 7.2-5 Direct mode (Dir Drive)

Direct mode is used to control the movement of the moving part following the movement reference indicated in the variable PARAM.



### Description of the commands and information displayed

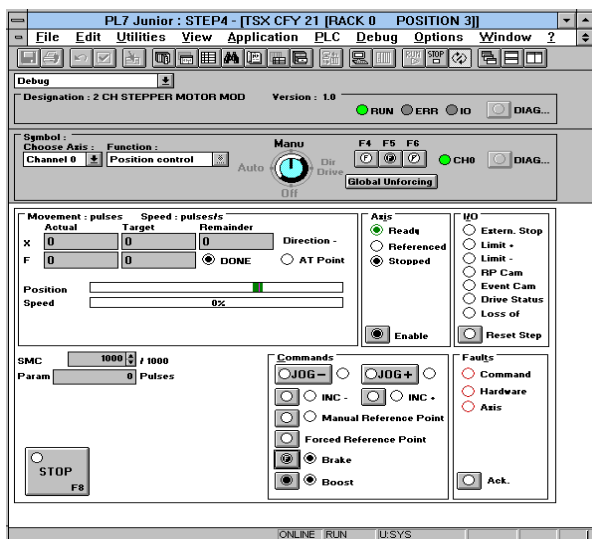
Movement : / Speed :

<b>X</b>	displays the position of the moving part in number of pulses
<b>F</b>	displays the speed of the moving part in Hertz
<b>Position</b>	bargraph showing the evolution of the moving part between the soft limits. It is red if there is an overshoot of the limits, otherwise it is green.
<b>Speed</b>	bargraph showing the speed of the moving part as a % of the maximum speed. It is red if there is an overshoot of the limits, otherwise it is green.
<b>Command</b>	<b>Role</b>
<b>STOP</b>	stops the moving part, while respecting deceleration
<b>Param</b>	enables the setpoint value from -FMAX to - SS_FREQ and from SS_FREQ to FMAX to be entered.
<b>Command</b>	applies the value entered in the <b>Param</b> field to the translator

For Axis Zones and faults and I/O see section 7. The Axis zone also contains the **Enable** command which is used to control the translator enable function. The faults zone also contains a command failure indicator lamp.

## 7.2-6 Manual mode (Man)

In manual mode, movement of the moving part is controlled directly from the debug screen, using elementary commands JOG+, JOG-, INC+, etc.



### Description of commands and information displayed

Movement : / Speed :

<b>X Current</b>	displays the position of the moving part in number of pulses
<b>X Target</b>	displays the target position for the moving part in number of pulses
<b>X Remainder</b>	displays the number of pulses remaining
<b>F Current</b>	displays the speed of the moving part in Hertz
<b>F Target</b>	displays the target speed for the moving part (manual speed corrected by the coefficient CMV).
<b>Position</b>	bargraph showing the evolution of the moving part between the soft limits. It is red if there is an overshoot of the limits, otherwise it is green.
<b>Speed</b>	bargraph showing the speed of the moving part as a % of the maximum speed. It is red if there is an overshoot of VMAX, otherwise it is green.

Indicators	Status	Meaning
<b>+ Direction</b>		indicates a movement of the moving part in a positive direction
<b>- Direction</b>		Indicates a movement of the moving part in a negative direction
<b>At Point</b>	on	indicates that the current movement has been completed, and the moving part is at the target point.
<b>DONE</b>	on	indicates that the current movement has been completed

I/O

Indicator lamps	Meaning
<b>External stop (2)</b>	state of the signal (0 or 1) at the "External stop" input. The indicator lamp is on when the external stop is activated = 24 V voltage on the input.
<b>+/- limit switch (2)</b>	"+/- limit switch" function activated. The indicator lamp is on when the moving part has reached the limit switch = 24 V not present at the input.
<b>RP cam</b>	state of the signal (0 or 1) at the "Reference point cam" input. The indicator lamp is on when the moving part is on the cam = 24 V present.
<b>Evt cam</b>	state of the signal (0 or 1) at the "Event" input. The indicator lamp is on when the moving part is on the event cam = 24 V present.
<b>Transl ctrl</b>	indicator lamp on if the translator does not give the "ready" signal. Indicator lamp off if the translator gives the OK signal. Levels depend on the choices made during configuration.
<b>Loss of step</b>	this indicator lamp signals the state of the "Loss of step check" input : the signal is supplied by the translator. It is on when the input is at 1 (or cable disconnected), otherwise it is off.

Button	Meaning
<b>Reset step</b>	Command to reset the translator loss of step detection system

Axis and faults (see section 7). The Axis zone also contains the **Enable** command which is used to control the translator enable function.

In this screen, when indicator lamps associated with commands are on, they indicate that the commands have been taken into account and are being executed. For the brake and boost the indicator lamps represent the activity of the associated output.



Command	Role
<b>CMV</b>	field for entering the speed multiplication coefficient with a value of 0 to 2000 in steps of 1/1000 (or a factor of 0 to 2 on the speed in steps of 1/1000).
<b>Param</b> command) forced reference point.	for entering the value of an incremental movement (INC+/INC- or a
<b>STOP</b> (1)	causes the moving part to stop

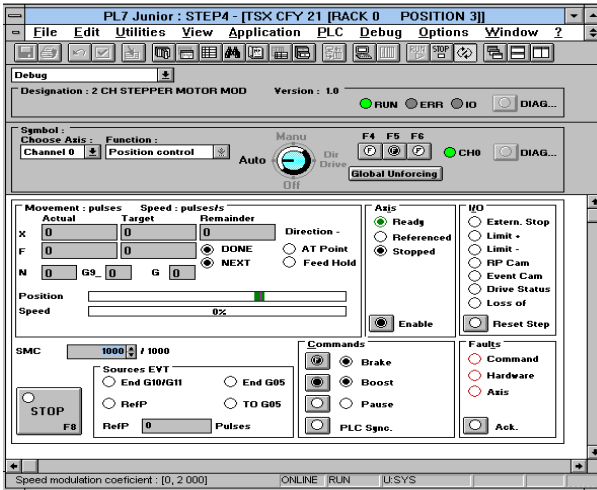
Commands (see section 5)

Command	Role
<b>JOG+</b>	command for unlimited movement in positive direction (1).
<b>JOG-</b>	command for unlimited movement in negative direction (1).
<b>INC+</b>	command for incremental movement in positive direction for a distance entered in the <b>Param</b> field.
<b>INC-</b>	command for incremental movement in negative direction for a distance entered in the <b>Param</b> field.
<b>Manual reference point</b>	command to set a manual reference point. The current position takes the "RP Value" defined in the adjustment screen, having found the reference point cam. The type of reference point is defined in the configuration screen.
<b>Forced reference point</b>	command to set a forced reference point. The current position is forced to the value entered in the <b>Param</b> field. This type of reference point does not lead to any movement of the moving part.
<b>Brake</b>	manual command for activating or deactivating the brake output. Note: if automatic control of the brake is configured, the last edge of the activation or deactivation command between this manual command (object %Qxy.i.13: BRAKE) and the automatic command is taken into account.
<b>Boost</b>	manual command for activating or deactivating the boost output. Note: if automatic control of the boost is configured, the last edge of the activation or deactivation command between this manual command (object %Qxy.i.14: BOOST) and the automatic command is taken into account.

- (1) This is a "latching" command. The button must be pressed again to deactivate it.
- (2) These commands remain active as long as the button is pressed. They are also used to release the moving part from a soft stop overshoot (after acknowledgment of the fault).

## 7.2-7 Automatic mode (Auto)

Automatic mode is that in which the SMOVE functions are executed.



Movement : / Speed :

<b>X Current</b>	displays the position of the moving part in number of pulses
<b>X Target</b>	displays the target position for the moving part in number of pulses
<b>X Remainder</b>	displays the difference between the target and current positions of the moving part.
<b>F Current</b>	displays the speed of the moving part in Hertz
<b>F Target</b>	displays the target speed for the moving part (speed defined in the SMOVE instruction corrected by the coefficient CMV).
<b>N G G9</b>	displays the instruction being executed. N= no. of step, G9 = type of movement, G = instruction code,
<b>Position</b>	bargraph showing the evolution of the moving part between the soft limits. It is red if there is an overshoot of the limits, otherwise it is green.
<b>Speed</b>	bargraph showing the speed of the moving part as a % of the maximum speed. It is red if there is an overshoot of the limits, otherwise it is green.



Indic. lamps	Status	Meaning
NEXT	on	indicates that the module is ready to receive a movement command
DONE	on	indicates that the current movement(s) has (have) been completed
+ Direction		indicates a movement of the moving part in a positive direction
- Direction		Indicates a movement of the moving part in a negative direction
At Point	on	indicates that the current movement has been completed, and the moving part is at the target point.
Immediate pause	on	indicates that the Immediate pause function is activated (coefficient CMV set to 0). The target position at that moment contains the immediate pause stop position.

I/O

Indicator lamps	Meaning
External stop	state of the signal (0 or 1) at the "External stop" input. The indicator lamp is on when the external stop is activated = 24 V voltage on the input.
+/- limit switch	"+/- limit switch" function activated. The indicator lamp is on when the moving part has reached the limit switch = 24 V not present at the input.
RP cam	state of the signal (0 or 1) at the "Reference point cam" input. The indicator lamp is on when the moving part is on the cam = 24 V present.
Evt cam	state of the signal (0 or 1) at the "Event" input. The indicator lamp is on when the moving part is on the event cam = 24 V present.
Transl ctrl	indicator lamp on if the translator does not give the "ready" signal. Indicator lamp off if the translator gives the OK signal. Levels depend on the choices made during configuration.
Loss of step	this indicator lamp signals the state of the "Loss of step check" input : the signal is supplied by the translator. It is on when the input is at 1 (or cable disconnected), otherwise it is off.

Button	Meaning
Reset step	Command to reset the translator loss of step detection system

Axis and faults (see section 7). The Axis zone also contains the **Enable** command which is used to control the translator enable function.

Command	Role
<b>CMV</b>	field for entering the speed multiplication coefficient with a value of 0 to 2000 in steps of 1/1000 (or a factor of 0 to 2 on the speed in steps of 1/1000).
<b>STOP</b>	causes the moving part to stop.

#### Event sources (EVT)

Indicator lamp	Status	Meaning
<b>PRef indic. lamp</b>	on	indicates that the event source is the PRef event cam (event input).
<b>PRef field</b>		displays the PRef memorized position (event input)
<b>End G05</b>	on	indicates the end of execution of instruction G05
<b>TO G05</b>	on	indicates that the TIME OUT defined in instruction G05 has elapsed
<b>End G10/G11</b>	on	indicates the occurrence of the event during execution of instruction G10 or G11

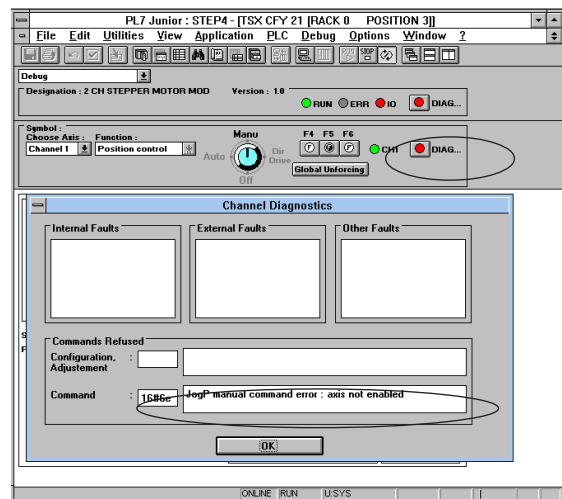
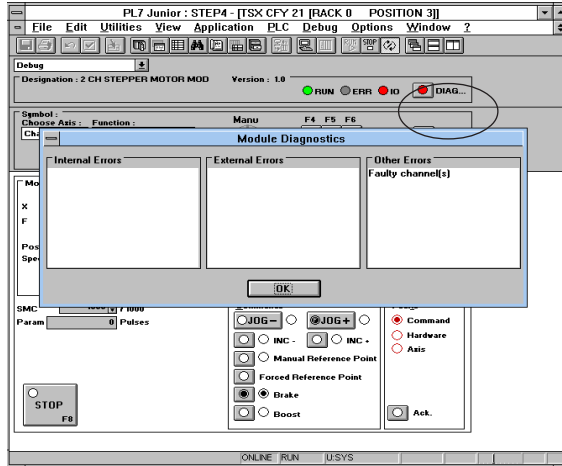
**Note :** this data is only updated if activation of the event-triggered task has been requested.

#### Commands

Command	Role
<b>Brake</b>	manual command for activating or deactivating the brake output. Note : if automatic control of the brake is configured, the last edge of the activation or deactivation command between this manual command (object %Qxy.i.13: BRAKE) and the automatic command is taken into account.
<b>Boost</b>	manual command for activating or deactivating the boost output. Note: if automatic control of the boost is configured, the last edge of the activation or deactivation command between this manual command (object %Qxy.i.14: BOOST) and the automatic command is taken into account.
<b>Pause</b>	stops sequence of movements at the end of the next movement with stop.
<b>CPU synchro</b>	command to trigger an event from the processor.

### 7.3 Diagnostics

In online mode the various Debug, Adjust and Configuration screens display the **DIAG** button, which gives detailed information on faults detected by the module and by the channel.



- **Internal fault** : internal module fault which generally requires replacement of the module.
- **External fault** : fault originating in the operative part (see section 5.6-6).
- **Other fault** : application fault (see section 5.6-7).
- **Command failures** : the cause of the command failure is given in the field concerned.

The list of command failure messages is given in section 14.

---

## 7.4 Archiving

---

When the program has been debugged in online mode :

- save the adjustment parameters if they have been modified, using the **Utilities/Save Parameters** command when the parameter adjustment screen is selected (see section 6).
- save the PL7 application to disk, using the **File/Save** command.

---

## 7.5 Documentation

---

The documentation for the axis control application is included in the complete PL7 application documentation.

It contains :

- the program part,
- the CONFIGURATION and saved ADJUSTMENT parameters which have been saved.

---

## 7.6 Simulation

---

Although no simulation mode has been offered, it is easy to operate TSX CFY module channels when no processes are present.

A TELEFAST discrete SIMULATION terminal block (Ref. ABE - 6TES160), supplied by a 24 V power supply available on the rack, is required, and needs to be linked directly to the CFY auxiliary I/O HE10 connector via a ribbon cable.

For channel 0 : Apply level 1 on inputs 2, 4 and 5. (Emergency stop, limit switches).  
For channel 1(TSX CFY21 only) : inputs 8, 10, and 11. Leave level 0 everywhere else.

During configuration of the CFY channel, check the "Translator inversions... Check input" box. This enables the channel to operate when no connections are present on the SUB-D (translator).

"Confirm" the channel in the setup screen in manual mode.  
Press a JOG button : movement is then simulated.

## 8.1 Designing a man-machine interface

### 8.1-1 Control station

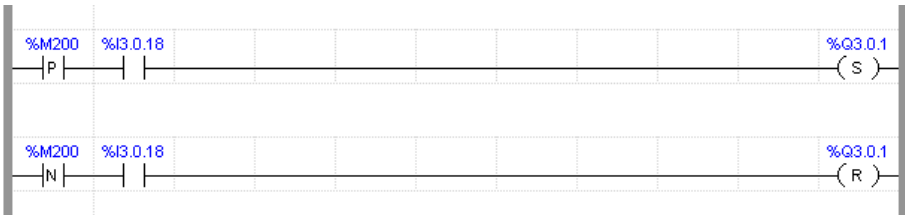
The programmer can use all the commands and elementary data in the form of command bits/words and status bits/words to design a simple or complex control station.

The programming principles are given in section 5.7, and an exhaustive list of all the bits and words is given in the Quick Reference Guide.

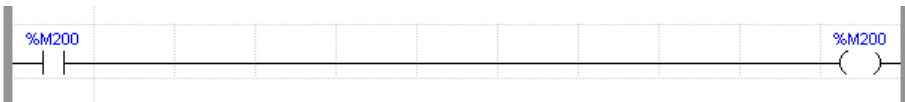
### 8.1-2 Man-machine interface on CCX 17

The following example enables a JOG+ (%Q3.0.1: JOGP) visual movement to be performed in manual mode (%I3.0.18: IN\_MANU) and to change the speed correction coefficient CMV (%QW3.0.1: CMV) on a CCX man-machine interface terminal.

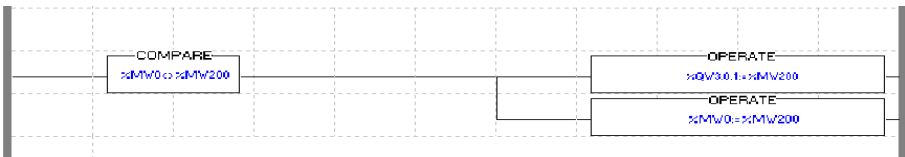
This command or modification can be performed either from the PL7 debug screen or from the CCX terminal.



%M200 corresponds to the status of the CCX terminal button activating a JOG+ command.



The above instruction is used to update bit %M200 on each scan for detecting edges.



%Mw200 contains the coefficient setpoint CMV entered on the CCX terminal. %Mw0 contains the last saved value of CMV entered on the CCX terminal.

---

C



---

## 9.1 Fault monitoring

---

The user has a number of means available to him for detecting faults :

- indicator lamps on the front panel of the module,
- diagnostic screens which can be accessed in online mode via the DIAG key from all the axis control module screens (see section 7),
- debug screens (see section 7),
- fault bits and status words (see section 5 and Quick Reference Guide).

---

## 9.2 Conditions for executing commands

---

**General conditions for movement commands (in auto or Man mode) :**

- Axis configured and with no blocking fault,
- The speed drive ENABLE command must be active (bit %Qxy.i.10 at 1) and the STOP command not active.
- Automatic or manual mode selected as appropriate,
- For absolute position commands : this position should be between the limits SL\_MIN and SL\_MAX,
- For relative position commands : the target calculated based on the relative current position should be between limits SL\_MIN and SL\_MAX,
- The axis should be referenced except for reference point and JOG commands,
- Speed F should be less than or equal to FMAX,
- If the moving part is outside the limits : the required direction of movement is the direction to return between the limits.

### Modifying the speed correction parameter CMV

If, in the case of modifications to the CMV parameter, the conditions mentioned above are no longer checked on the speed, the speed will be limited to FMAX.

**Note** : a movement without stopping which is not followed by any sequencing command continues until the soft stop limits are reached.

---

### 9.3 Diagnostic help

---

**Symptom** : The CFY module appears not to have taken account of the new parameters written by WRITE\_PARAM (1)

**Diagnostics** : Program a READ\_PARAM into your application to find out the values actually used by the CFY.

A WRITE\_PARAM triggered while another adjust exchange is in progress is ignored.

**Procedure** : Test bit %MWxy.i.0:X2 before any adjust exchange.

**Symptom** : Event-triggered processing associated with the axis control channel is not executed.

**Diagnostics** : Check the event feedback circuit.

- Event number declared at configuration is identical to that of the event-triggered processing
- Activation of the event processing requested (M code of SMOVE command = 16#1000)
- Events authorized at system level (%S38 = 1)
- Unmasked events at system level (UNMASKEVT)

**Procedure** : see part concerning common features of application-specific functions, on using events.

**Symptom** : Settings have been lost

**Diagnostics** : A cold restart or reconfiguration of the axis control channel loses the current adjustments made via the screen or a WRITE\_PARAM

**Procedure** : Save the current adjustments using the "Save Parameters" function or the SAVE\_PARAM instruction.

**Symptom** : The operating status words %MWmy.i.1 and 2 are not consistent with the status of the axis control channel.

**Diagnostics** : These words are only updated on an explicit READ\_STS request

**Procedure** : Program a READ\_STS into your application

(1) The syntax of READ\_PARAM and WRITE\_PARAM instructions is described in the part on the common features of application-specific functions.

---

**Symptom** : The commands in the debug screen have no effect.

**Diagnostics** : The application or the task is in STOP mode.

**Procedure** : Switch the application or task to RUN.

**Symptom** : Some commands in the debug screen cannot be modified.

**Diagnostics** : These bits are written by the application.

**Procedure**: Use bit forcing (for %Qxy.i.j objects) or design the application so that it does not write these bits automatically (modification on a transition and not on a state).

**Symptom** : It is not possible to enter more than 3 characters in the numerical fields in the adjustment and configuration screens.

**Diagnostics** : A thousands separator has not been selected in the Windows control panel.

**Procedure**: in the control panel, select the "International" icon in the "Numbers format" field, activate the "Modify" command and select a thousands separator.

**Symptom** : Command failure after stopping on overshoot of soft stops in DIRDRIVE mode.

**Diagnostics** : DIRDRIVE mode is activated after MAN or AUTO mode in which a reference point has been set has been used. The axis is referenced. Control of the soft limits is active.

Overshooting one of the soft stops causes a stop with fault.

No other movement is accepted in DIRDRIVE mode.

**Procedure** :

Two types of action are possible to restart movements :

1) Force the loss of the axis reference : Once the moving part has stopped completely, disable then re-enable the channel : ENABLE (%Qxy.i.10) =0 then 1, THEN acknowledge the fault (Rising edge on the ACK\_DEF %Qxy.i.9 command).

2) Force the position of the moving part between the soft stops : for example, change to MAN mode momentarily, acknowledge the fault (ACK\_DEF), set a forced reference point at a position within the soft stops, return to DIRDRIVE mode.

---

**Symptom** : In AUTO mode, movement commands are not executed correctly after an overshoot of a soft stop.

**Diagnostics** : After an overshoot of a soft stop, the only movement commands accepted are those to return between the stops.

**Procedure** : Check that the movement requested, which has been performed incorrectly, will make the moving part return between the stops.

## 10.1 Characteristics of the stepper motor control functions

### Size of an SMOVE function

	Bit memory	Data zone	Program zone
TSX CFY 11	29	390	170
TSX CFY 21	58	780	220
Overhead for 1st channel configured			2290

Approximate sizes are given in 16-bit words.

### Execution time

Reading of TSX CFY I/O	95 $\mu$ s
Fonction SMOVE	840 $\mu$ s
READ_STATUS	540 $\mu$ s
READ_PARAM	460 $\mu$ s
WRITE_PARAM	760 $\mu$ s
SAVE_PARAM	500 $\mu$ s
RESTORE_PARAM	780 $\mu$ s
Taking an adjustment into account (after a WRITE_PARAM instruction)	60 ms for CFY11 210 ms for CFY 21
Taking the reconfiguration of a channel into account	1.5 s

### Module cycle time

The module cycle time is 10ms.

## 10.2 Limitations of the TSX CFY module

### 10.2-1 Low amplitude movements

A low amplitude movement corresponds to a movement which cannot be used to reach the speed specified in the instruction. The speed profile is triangular rather than trapezoid.

For the instruction :

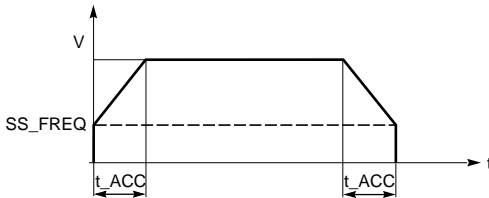
```
SMOVE %CH7.1(1,90,09,X1,V,0)
```

X1 defines the position to be reached, V fixes the "cruise" speed at which movement should be made.

and X0 the starting position of the moving part.

#### 1st case :

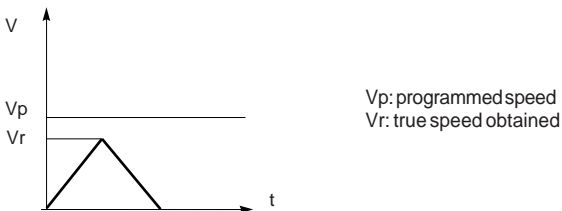
The distance to be covered  $|X1-X0|$  is long enough to reach the specified speed V. Movement is made according to a trapezoid speed trajectory.



This trajectory shows the duration of the acceleration and deceleration phases which are respectively equal to  $t_{ACC}$ .

#### 2nd case :

The distance to be covered  $|X1-X0|$  is not long enough to reach the specified speed V. Movement is made according to a triangular speed trajectory as shown below. Duration of the acceleration and deceleration phases is reduced in proportion to speeds.



### 10.2-2 Maximum start/stop frequency

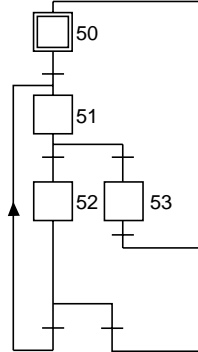
The maximum value of the SS\_FREQ parameter, the start/stop frequency, is 4 KHz (irrespective of the maximum frequency).

## 11.1 Teaching the positions

The PL7 program in the following example :

- teaches 16 positions in the first chart
- describes their use in the second chart

### Chart for teaching positions



STEP 50 ACTION ON ACTIVATION

```
<memorizes %MW99 in order to use it as a limit
!   %MW98:=%MW99;
```

```
<Initializes the index during the teach phase
```

```
!   %MW99:=-1;
```

```
TRANSITION: X50->X51
```

```
!   RE %I2.0
```

STEP 51 ACTION ON ACTIVATION

```
<updates the index
```

```
!   %MW99:=%MW99+1;
```

```
<teach positions
```

```
!   %MD200[%MW99]:=%ID7.0;
```

```
TRANSITION: X51->X52
```

```
!   %MW99<=16
```

```
TRANSITION: X51->X53
```

```
!   %MW99>16
```

```
TRANSITION: X53->X50
```

```
!   RE %I2.1
```

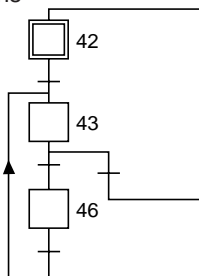
```
TRANSITION: X52->X51
```

```
!   RE %I2.0
```

```
TRANSITION: X52->X50
```

```
!   RE %I2.1
```

Chart describing the use of the positions



STEP 42 ACTION ON ACTIVATION

<initializes %MW97 as the execution index

! %MW97:=-1;

TRANSITION: X42->X43

! RE %I2.2

STEP 43 ACTION ON ACTIVATION

<increments the execution index

! %MW97:=%MW97+1;

<executes the next segment

! SMOVE %CH7.0(%MW97,%KW8,%KW1,%MD200[%MW97],150000,0);

%KW8 : 90 movement with absolute value

%KW1 : 09 go to point and stop

TRANSITION: X43->X46

! %I7.0.0 AND (%MW97<%MW98) AND NOT %I7.0.2

TRANSITION: X43->X42

! (%I7.0.1 AND(%MW97>=%MW98))OR %I7.0.2

TRANSITION: X46->X43

! TRUE



---

**12.1 Glossary**


---

<b>Axis</b>	Motor/translator/mechanism combination which controls the movement of a moving part in a given direction (axis, linear movement) or around a fixed rotation axis (rotating axis, circular movement).
<b>Axis referenced</b>	Module status once the reference point has been set. Position measurements are only significant and movements are only permitted in this state (except JOG).
<b>Boost</b>	Differential output for activating/deactivating the stepper motor boost (1).
<b>Brake</b>	24 V solid state output for controlling the axis brake (1).
<b>Emergency stop</b>	Movement stop, followed by the switching of the outputs to safety position = translator disabled, brake active.
<b>Event</b>	A change of state of the event input or the EXT_EVT bit which is accessed by the program.
<b>External stop</b>	24 V input used to stop movements, equivalent to object %Qxy.i.8: STOP, active at 1.
<b>Forced reference point</b>	Procedure for loading the current position measurement at a preset value. This operation references the axis.
<b>ISO</b>	International Standards Organization. The ISO code is widely used. Formats, symbols and transmission rules are covered by ISO standards. AFNOR is a member of the ISO.
<b>Limit switches</b>	Inputs which cause the moving part to stop (no default). It is possible to use them as a reference cam.
<b>Loss of step</b>	Differential input for a signal supplied by the translator on detection of loss of step. State available in object %lxy.i.28: STEP_FLT.
<b>Lower soft stop</b>	Lower position measurement which the moving part must not overshoot (set by command parameter SL_MIN).
<b>Machine reference point</b>	Machine axis measurement reference point.
<b>Movement profile</b>	This is the profile of variation in position, speed and acceleration references. It is often illustrated by the curve : speed = F (time). For TSX CFY 11/21 stepper motor axis control, the profile is trapezoid.

---

**Parametered indexed position (PREF)**

Index value for calculating indexed positions. Absolute position = index (PREF) + indexed position

**Pulse outputs** Translator control differential outputs.

**Reference point setting**

Procedure for loading the current position measurement by moving the moving part to a reference point cam or a limit switch cam. This operation references the axis.

**Reset loss of step** Differential output for resetting the translator loss of step check system, controlled by object %Qxy.i.15: ACK\_STEPFLT. (2).

**Speed correction coefficient (CMV)**

A coefficient which multiplies all speeds by a value of 0 to 2 in steps of 1/1000.

**Speed reference** Theoretical speed of the moving part calculated by the module using the acceleration profile and the programmed speed.

**Trajectory** Series of elementary movements which pass through intermediate markers between a start marker and a target marker. The movement between the two markers is executed at an appropriate speed or in an appropriate time.

**Translator** Power unit providing an interface between the TSX CFY control and a stepper motor.

**Translator check** Differential input showing the availability of the translator.

**Translator enable** Differential output for enablin/ inhibiting the translator (1).

**Upper soft stop** Upper position measurement which the moving part must not overshoot (set by command parameter SL\_MAX).

**Valid measurement area**

Set of measurement points between the two soft stops.

(1) logic defined during configuration.

(2) no internal effect on movements.

---

**Programming**


---

**SMOVE %CHxy.i(N\_Run,G9\_,G,X,F,M)**

**%CHxy.i**= address of the axis control module in the PLC configuration

**x** = rack number

**y** = position of the module in the rack

**i** = channel number

**N\_RUN**= 0 to 32767, number identifying the movement performed by the SMOVE function. Identifies the current movement in debug mode.

**G9\_** = type of movement

**90** movement to an absolute position value,

**91** movement to a relative value with respect to the current position.

**98** movement to a relative value with respect to memorized position PREF (position PREF is memorized using instruction code G07).

**G** = instruction code,

**09** : move to the position and stop

**01** : move to the position without stopping

**10** : move until an event is detected and stop

**11** : move until an event is detected without stopping

**14** : reference point

**05** : await an event

**07** : memorize the current position when an event occurs

**62** : forced reference point

**X** = position to be reached or towards which the moving part must move (in the event of a move to a position without stopping).

This position can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this word can be indexed).

The unit in which these values are expressed is a number of pulses.

**F** = speed of movement of the moving part. This speed can be :

- immediate
- coded in an internal double word %MDi or internal constant %KDi (this word can be indexed).

The unit of speed is pulses per second.

**M** = Word coded on 4 four-bit bytes (in hexadecimal) 16# 

3	2	1	0

- activation of the application event processing for instructions : 10, 11, 05 and 07 (M=16#1000 for activation)

---

## General module data

---

<b>%Ixy.MOD.ERR</b>	<b>module fault</b>
---------------------	---------------------

<b>%MWxy.MOD.2:Xj</b>	<b>Module standard status word</b>
-----------------------	------------------------------------

bit 0	internal fault (module off)
bit 1	operating fault (see channel status word)
bit 3	module performing self-tests
bit 5	hardware or software configuration fault
bit 6	module absent

---

## Internal command data (implicit exchanges)

---

<b>%Qxy.i.j</b>
-----------------

<b>Processor --&gt; TSX CFY</b>
---------------------------------

bit 0	<b>DIRDRV</b>	State	movement command in direct drive mode
bit 1	<b>JOG_P</b>	State	unlimited manual movement in positive direction
bit 2	<b>JOG_M</b>	State	unlimited manual movement in negative direction
bit 3	<b>INC_P</b>	Edge	incremental movement (PARAM) in positive direction command
bit 4	<b>INC_M</b>	Edge	incremental movement (PARAM) in negative direction command
bit 5	<b>SET_RP</b>	Edge	set manual reference point (RP_POS=reference value)
bit 6	<b>RP_HERE</b>	Edge	reference point forced to a value defined in PARAM
bit 8	<b>STOP</b>	State	immediate stop command (stop moving part)
bit 9	<b>ACK_DEF</b>	Edge	acknowledge channel faults
bit 10	<b>ENABLE</b>	State	enable translator
bit 11	<b>EXT_EVT</b>	Edge	command to trigger an event from the processor
bit 12	<b>PAUSE</b>	State	command to suspend movement at the end of the next movement with stop
bit 13	<b>BRAKE</b>	Edge	stepper motor brake control
bit 14	<b>BOOST</b>	Edge	translator boost
bit 15	<b>ACK_STEPFLT</b>	State	translator step check reset command

<b>%QWxy.i.0</b>	<b>MODE_SEL</b>	<b>Mode selector</b>
------------------	-----------------	----------------------

Value

0	<b>DRV_OFF</b>	off mode : inhibition of translator
1	<b>DIRDRIVE</b>	direct mode : direct movement control by speed
2	<b>MANU</b>	manual mode
3	<b>AUTO</b>	automatic mode

<b>%QWxy.i.1</b>	<b>CMV</b>	<b>Speed correction</b>
------------------	------------	-------------------------

Value : value of speed correction setpoint from 0 to 2000 in steps of 1

<b>%QDxy.i.2</b>	<b>PARAM</b>	<b>Value of the movement increment</b>
------------------	--------------	--

## Internal status data (implicit exchanges)

%lx.y.i.j

processor &lt;-- CAY

bit 0	<b>NEXT</b>	ready to receive a new movement command (in AUTO)
bit 1	<b>DONE</b>	all the instructions have been executed : no more instructions in the stack in AUTO mode
bit 2	<b>AX_FLT</b>	error on the axis
bit 3	<b>AX_OK</b>	no fault causing the moving part to stop
bit 4	<b>HD_ERR</b>	presence of a hardware fault
bit 5	<b>AX_ERR</b>	presence of an application fault
bit 6	<b>CMD_NOK</b>	command failure
bit 8	<b>NOMOTION</b>	moving part stationary
bit 9	<b>AT_PNT</b>	moving part positioned on target (instruction with stop)
bit 11	<b>CONF_OK</b>	the axis is configured
bit 12	<b>REF_OK</b>	reference point set (axis referenced)
bit 13	<b>AX_EVT</b>	copies the physical event inputs
bit 14	<b>HOME</b>	copies the reference point CAM physical input : 1: on cam, 0: off cam
bit 15	<b>DIRECT</b>	indicates the direction of movement : 1: + direction, 0: - direction
bit 16	<b>IN_OFF</b>	stop mode selected
bit 17	<b>IN_DIRDR</b>	DIRDRIVE mode active
bit 18	<b>IN_MANU</b>	MANU mode active
bit 19	<b>IN_AUTO</b>	AUTO mode active
bit 20	<b>ST_DIRDR</b>	DIRDRIVE control active
bit 21	<b>ST_JOG_P</b>	unlimited movement in + direction in progress
bit 22	<b>ST_JOG_M</b>	unlimited movement in - direction in progress
bit 23	<b>ST_INC_P</b>	incremental movement in + direction in progress
bit 24	<b>ST_INC_M</b>	incremental movement in - direction in progress
bit 25	<b>ST_SETRP</b>	current manual reference point
bit 26	<b>ON_PAUSE</b>	movement sequencing suspended
bit 27	<b>IM_PAUSE</b>	movement suspended (immediate PAUSE)
bit 28	<b>STEP_FLT</b>	loss of step input state
bit 29	<b>EMG_STOP</b>	emergency stop input state
bit 30	<b>EXT_STOP</b>	external stop input state
bit 31	<b>HD_LMAX</b>	positive limit switch state
bit 32	<b>HD_LMIN</b>	negative limit switch state
bit 33	<b>ST_BRAKE</b>	stepper motor brake output image
bit 34	<b>ST_BOOST</b>	BOOST output activity image
bit 35	<b>ST_DRIVE</b>	translator status
bit 36	<b>OVR_EVT</b>	event overrun
bit 37	<b>EVT_G07</b>	event source : position memorization (1)
bit 38	<b>EVT_G05</b>	event source : end of G05 on detection of event (1)
bit 39	<b>TO_G05</b>	event source : G05 timeout elapsed (1)
bit 40	<b>EVT_G1X</b>	event source : end of G10 or G11 on detection of event (1)

<b>%IDxy.i.0</b>	<b>POS</b>	<b>momentary position</b>
<b>%IDxy.i.2</b>	<b>SPEED</b>	<b>momentary speed</b>
<b>%IDxy.i.4</b>	<b>REMAIN</b>	<b>number of pulses still to go</b>
<b>%IWxy.i.6</b>	<b>SYNC_NRUN</b>	<b>number of step in progress</b>
<b>%IDxy.i.7</b>	<b>PREF</b>	<b>value of PREF register (1)</b>

(1) only updated when event processing is activated.

## Internal status data (explicit exchanges)

### **%MWxy.i.0:Xj**    **EX\_STS**    exchange management

bit X0	<b>STATUS</b>	exchange of status parameters in progress (STATUS)
bit X2	<b>ADJUST</b>	exchange of adjustment parameters in progress
bit X15	<b>CONFIG</b>	reconfiguration of module in progress

### **%MWxy.i.1:Xj**    **EX\_RPT**    exchange report

bit X2	<b>ADJ_RPT</b>	adjustment parameters exchange report
bit X15	<b>CONF_FLT</b>	configuration fault

### **%MWxy.i.2:Xj**    **CH\_STS**    channel operating status

bit X0	<b>EXT_FLT</b>	external fault (same as bit HD_ERR of %Ixy.i.4)
bit X4	<b>MOD_FLT</b>	internal fault : module absent, off or performing self-tests
bit X5	<b>CONF_FLT</b>	hardware or software configuration fault
bit X6	<b>COM_FLT</b>	communication fault with CPU
bit X7	<b>APP_FLT</b>	application fault : incorrect configuration, command failure
bit X8	<b>CH_LED_LOW</b>	status of channel indicator lamps: off: X9, X8=00, flashing: X9, X8=01, on: X9, X8=10
bit X9	<b>CH_LED_HIGH</b>	status of channel indicator lamps: off: X9, X8=00, flashing: X9, X8=01, on: X9, X8=10

### **%MWxy.i.3:Xj**    **AX\_STS**    Axis operating status

Hardware faults : %Ixy.i.4 **HD\_ERR** (includes all the faults below)

bit X1	<b>BRAKE_FLT</b>	brake output short-circuit fault
bit X2	<b>DRV_FLT</b>	translator fault
bit X5	<b>EMG_STP</b>	emergency stop fault
bit X6	<b>AUX_SUP</b>	24 V supply fault

Application faults : %Ixy.i.5 **AX\_ERR** (includes all the faults below)

bit X3	<b>SLMAX</b>	SLMAX upper soft stop fault
bit X4	<b>SLMIN</b>	SLMIN lower soft stop fault

<b>%MWxy.i.4</b>	<b>N_RUN</b>	<b>current step number</b>
<b>%MWxy.i.5</b>	<b>G9_COD</b>	<b>current movement type</b>
<b>%MWxy.i.6</b>	<b>G_COD</b>	<b>current instruction</b>
<b>%MWxy.i.7</b>	<b>CMD_FLT</b>	<b>command failure report</b>
<b>%MDxy.i.8</b>	<b>T_XPOS</b>	<b>target position</b>
<b>%MDxy.i.10</b>	<b>T_SPEED</b>	<b>target speed</b>

#### Note :

%MWxy.i.0 and 1 words are managed by the PLC processor and are continually updated. The other words contain data from the TSX CFY module. They are only updated after a READ\_PARAM %CHxy.i instruction.

## Adjustment parameters (explicit exchanges)

%MWxy.i.j or %MDxy.i.j

%MDxy.i.12	<b>ACC</b>	acceleration value : depends on the user unit
%MDxy.i.14	<b>SL_MAX</b>	upper soft stop
%MDxy.i.16	<b>SL_MIN</b>	lower soft stop
%MDxy.i.18	<b>SS_FREQ</b>	start stop speed: from 0 to FMAX
%MDxy.i.20	<b>MAN_SPD</b>	speed in manual mode : from SS_FREQ to FMAX
%MDxy.i.22	<b>RP_POS</b>	reference point value in manual mode from SL_MIN to SL_MAX
%MWxy.i.24	<b>BRK_DLY1</b>	shift on deactivation of brake : from -1000 to 1000
%MWxy.i.25	<b>BRK_DLY2</b>	shift on activation of brake : from -1000 to 1000
%MWxy.i.26	<b>STOP_DLY</b>	duration of stop plateau at start/stop speed: from 0 to 1000

### Adjustment screen

#### Note :

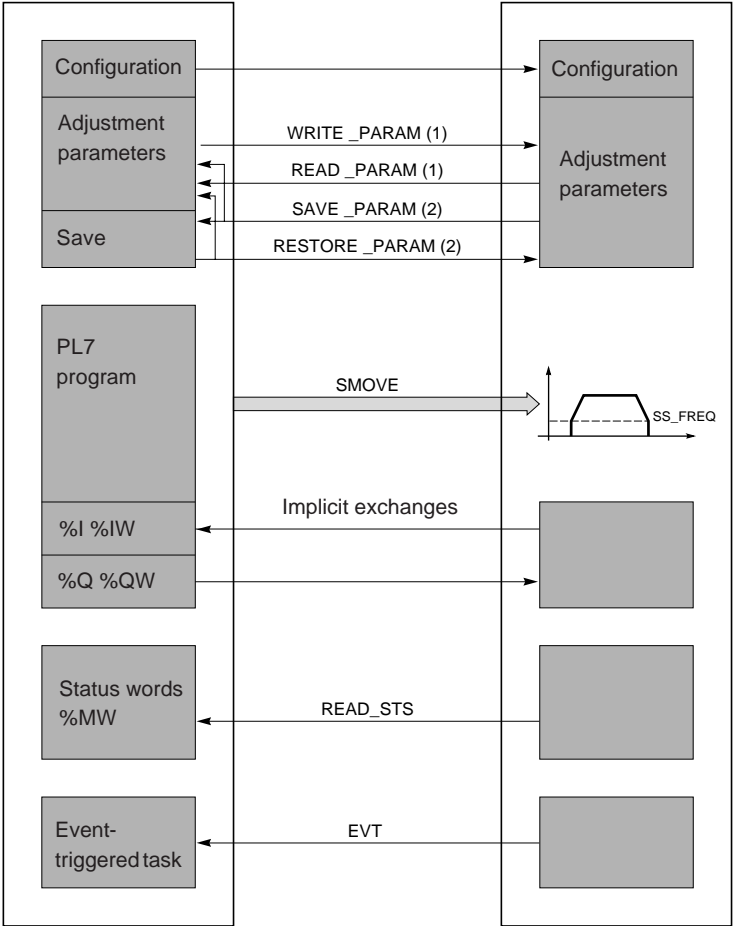
Adjustment parameters (words %MDxy.i. 12 to 26) are transferred to the TSX CFY module when the READ\_PARAM %CHxy.i instruction is executed. The user must ensure that the module has taken the parameters into account by testing the bits %MWxy.i.0:X2 (0 : end of exchange ; 1 : exchange in progress) and %MWxy.i.1:X2 (0:parameters accepted ; 1 : parameters refused).

**Block diagram of data exchanges**

C

**Processor**

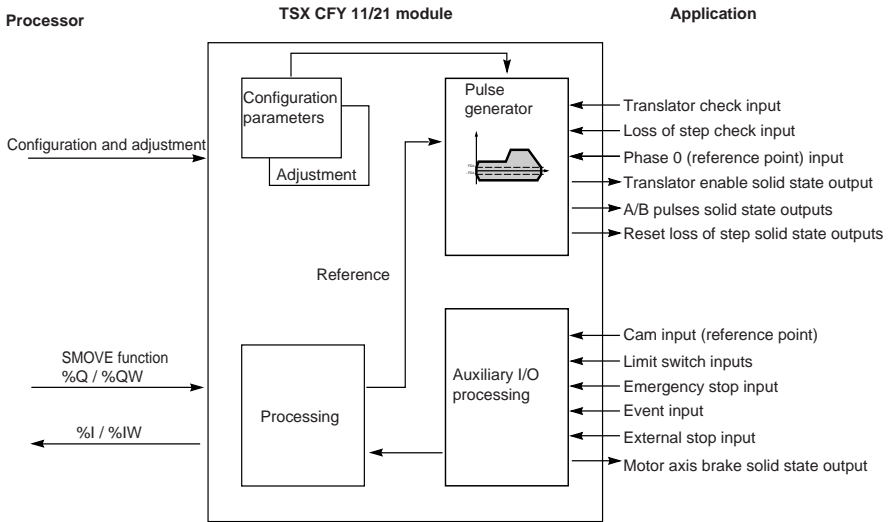
**TSX CFY 1 module**



(1) read or write from the adjustment screen  
 (2) save or restore using the Save Parameters or Restore Parameters commands in the PL7-Junior Utilities menu or using the SAVE\_PARAM or RESTORE\_PARAM instructions.



**Block diagram of the TSX CFY module**





### 14.1 List of *CMD\_FLT* error codes

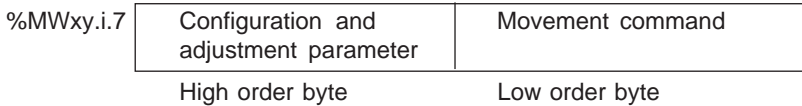
A list of messages explaining the *CMD\_FLT* (%MWxy.i.7) command failure word is given on the following pages.

This word is read by explicit exchange.

Messages also appear in clear text in the Diagnostics dialog boxes which can be accessed using the **DIAG** key (see section 7.3).

The *CMD\_FLT* word consists of two bytes.

Each byte corresponds to a specific class of error.



Low order byte : error in the configuration and adjustment parameters. (XX00)

High order byte : failure to execute the movement command. (00XX):

Example :

**0004**

└── Low order byte : JogP command error

---

The values in brackets are the code values in hexadecimal.

**Configuration** (in the high order byte of word %MWxy.i.7)

- 2 (2) = Reference point configuration error
- 3 (3) = Event configuration error
- 4 (4) = Maximum frequency configuration error
- 5 (5) = Maximum acceleration configuration error

**Adjustment parameter** (in the high order byte of word %MWxy.i.7)

- 7 (07) = Acceleration parameter error
- 8 (08) = Upper soft stop parameter error
- 9 (09) = Lower soft stop parameter error
- 10 (0A) = Start/stop frequency parameter error
- 11 (0B) = Manual mode frequency parameter error
- 12 (0C) = Reference point value parameter error
- 13 (0D) = Brake off delay parameter error
- 14 (0E) = Brake on delay parameter error
- 15 (0F) = Stop plateau parameter error
- 32 (20) = WRITE\_PARAM positive parameter in progress error

**Movement command failure** (in the low order byte of word %MWxy.i.7)

- 1 (01) = Insufficient conditions (Mode, Value, etc) manual command error
- 2 (02) = Current manual movement manual command error
- 3 (03) = Simultaneous commands manual command error
- 4 (04) = JogP manual command error
- 5 (05) = JogM manual command error
- 6 (06) = IncP position below lower limit (soft stop or physical) manual command error
- 7 (07) = IncM position above upper limit (soft stop or physical) manual command error
- 8 (08) = IncP parameter manual command error
- 9 (09) = IncM parameter manual command error
- 10 (0A) = Manual RP manual command error
- 11 (0B) = Forced RP manual command error
- 12 (0C) = Insufficient conditions (parameters) auto command error
- 13 (0D) = Current auto movement manual command error
- 14 (0E) = Insufficient conditions (Mode) movement command error
- 15 (0F) = G01" Move command error (1)
- 16 (10) = G09" Move command error (1)
- 17 (11) = G10" Move command error (1)
- 18 (12) = G11" Move command error (1)

(1) indicates that one of the SMOVE function parameters is incorrect. Examples :  
incorrect movement type code, soft stop overshoot, speed greater than FMAX, etc.

- 
- 21 (15)** = G14" Move command error (1)  
**22 (16)** = G05" Move command error (1)  
**23 (17)** = G07" Move command error (1)  
**24 (18)** = G62" Move command error (1)  
**25 (19)** = Execution Move command error  
**26 (1A)** = Move in progress Auto command error  
**27 (1B)** = Auto command error: stack full  
**48 (30)** = Insufficient conditions DirDrv command error  
**49 (31)** = Change of operating mode DirDrv command error  
**50 (32)** = Axis moving DirDrv command error  
**51 (33)** = Axis stopped DirDrv command error  
**52 (34)** = Axis not enabled DirDrv command error  
**53 (35)** = Blocking fault DirDrv command error  
**54 (36)** = Frequency less than SS\_FREQ DirDrv command error  
**55 (37)** = Frequency greater than FMax DirDrv command error  
**56 (38)** = Axis on upper physical limit DirDrv command error  
**57 (39)** = Axis on lower physical limit DirDrv command error  
**58 (3A)** = Axis above upper physical limit DirDrv command error  
**59 (3B)** = Axis below lower physical limit DirDrv command error  
**60 (3C)** = Axis above upper soft stop DirDrv command error  
**61 (3D)** = Axis below lower soft stop DirDrv command error  
**96 (60)** = Upper soft stop JogP manual command error  
**97 (61)** = Axis stopped JogP manual command error  
**101 (65)** = JogM current movement JogP manual command error  
**102 (66)** = JogP on upper physical limit manual command error  
**103 (67)** = Position above upper physical limit JogP manual command error  
**108 (6C)** = Blocking fault other than soft stop JogP manual command error  
**109 (6D)** = Soft stop blocking fault not acknowledged JogP manual command error  
**110 (6E)** = Axis not enabled JogP manual command error  
**113 (71)** = Axis stopped JogM manual command error  
**116 (74)** = JogP current movement JogM manual command error  
**118 (76)** = JogM on lower physical limit manual command error  
**119 (77)** = Position below lower physical limit JogM manual command error  
**124 (7C)** = Blocking fault other than soft stop JogM manual command error  
**125 (7D)** = Soft stop blocking fault not acknowledged JogM manual command error  
**126 (7E)** = Axis not enabled JogM manual command error  
**127 (7F)** = JogM on lower soft stop manual command error  
**130 (82)** = Position below lower physical limit IncP manual command error  
**131 (83)** = Position above upper physical limit IncP manual command error  
**132 (84)** = Current movement in JogP IncP manual command error  
**133 (85)** = Current movement in JogM IncP manual command error

- 
- 134 (86)** = IncP on lower physical limit manual command error  
**135 (87)** = Position above upper physical limit IncP manual command error  
**136 (88)** = Axis not referenced IncP manual command error  
**137 (89)** = IncP causing movement of the lower soft stop manual command error  
**138 (8A)** = Stop condition IncP manual command error  
**141 (8D)** = Axis not enabled IncP manual command error  
**146 (92)** = Position below lower soft stop IncM manual command error  
**147 (93)** = Position above upper soft stop IncM manual command error  
**148 (94)** = Current movement in JogP IncM manual command error  
**149 (95)** = Current movement in JogM IncM manual command error  
**150 (96)** = IncM on lower physical limit manual command error  
**151 (97)** = Position above upper physical limit IncM manual command error  
**152 (98)** = Axis not referenced IncM manual command error  
**154 (9A)** = Stop condition IncM manual command error  
**155 (9B)** = IncM causing overshoot of upper soft stop manual command error  
**158 (9E)** = Axis not enabled IncM manual command error  
**164 (A4)** = Current movement in JogP Manual RP command error  
**165 (A5)** = IncM current movement in JogM Manual RP command error  
**170 (AA)** = Stop condition Manual RP command error  
**174 (AE)** = Axis not enabled Manual RP command error
- 178 (B2)** = Position below lower soft stop Forced RP command error  
**179 (B3)** = Position above upper soft stop Forced RP command error  
**180 (B4)** = Current movement in JogP Forced RP command error  
**181 (B5)** = Current movement in JogM Forced RP command error  
**189 (BD)** = Soft stop error not acknowledged Forced RP command error

**A**

Absolute movements	5/6
Adjusting the speed controller	6/1
Archiving	7/14
AUTO	5/1
Automatic	5/1, 7/10
Await event	5/10
Axis	12/1
Axis referenced	12/1

**C**

Channel address	5/3
Characteristics	10/1
CMD_FAIL	14/1
CMV	7/9, 12/2
Command buttons	7/2
Command failure	5/25, 14/1
Current parameters	6/2

**D**

Debug screens	7/3
Deferred PAUSE	5/16
Diagnostics	7/13
DIRDRIVE	5/1, 5/30, 7/6
Direct drive	5/30
Direct drive mode	7/6
Documentation	7/14
DONE	5/13

**E**

Elementary movements	5/6
Emergency stop	1/2, 5/22, 12/1
Encoder	12/1
Encoder offset	6/4
Encoder type	4/6
Event	5/11, 12/1
Event processing	5/5, 5/18, 13/1
Event task	4/12

**F**

F	5/5
Fault indication	5/21
Fault management	5/20
Fault monitoring	9/1
Forced reference point	5/10, 5/29, 7/9, 7/12
Forced reference point setting	12/1

**G**

G	5/4
G9_	5/4

**I**

Immediate PAUSE	5/17
INC_M	5/28, 7/9
INC_P	5/28, 7/9
Incremental encoder	4/5, 4/7, 4/8
Incremental movement	5/28
Initial parameters	6/2
Instruction code	5/4
G01	5/7
G05	5/10
G07	5/11
G09	5/7
G10	5/8
G11	5/8
G14	5/8
G62	5/10
G90	5/6
G91	5/6
G98	5/6
ISO	12/1

**J**

JOG_M	5/27, 7/9
JOG_P	5/27

**L**

Limit switches	1/2, 1/3
Loss of step	1/3

**M**

M	5/5
Machine reference point	12/1
MAN	5/1, 7/7
Man-machine interface	8/1
Manual	5/1
Manual mode	5/26, 7/7
Manual mode parameters	6/7
Manual reference point	7/9
Masking of events	5/18
Memorize current position when an event occurs	5/11
Methodology	3/1
Module faults	5/22
Move to a position and stop	5/7
Move to a position without stopping	5/7
Move until an event is detected	5/8
Movement codes	5/4
Movement identifier	5/3
Movement profile	12/1

**N**

N_Run	5/3
NEXT	5/13

**O**

OFF	5/1, 5/31, 7/5
Operating modes	5/1, 5/19

**P**

Param	7/6, 7/9
Parameter configuration	4/4
Parametered indexed position	12/2
Position to be reached	5/5
Power outage and return	5/19

**R**

Reconfiguration	5/19
Reference point	4/12, 5/8, 5/29
Reference point setting	12/2
Resolution	4/7
Restore Parameters	6/9

**S**

Save Parameters	6/9
Sequence of movement commands	5/13
SETRP	5/29
SMOVE	5/3
SMOVE function	5/3
Soft stop	12/1
Soft stops	5/24
Speed correction coefficient	12/2
Speed of movement	5/5
Speed reference	12/2
Step-by-step	5/16
STOP	7/6
Stop mode	5/31, 7/5
Stopping a movement	5/26
Synchronization of several axes	5/16

**T**

Teaching the positions	11/1
Trajectory	12/2
Transfer	5/32
Tutorial	2/1
Debugging	2/14
Description	2/1
Design	2/4
Programming	2/9
Type of movement	5/4

**U**

Units of measurement	4/5
Upper and lower limits	4/9, 4/10
User interface	7/2

**V**

Visual movement	5/27
-----------------	------

**X**

X	5/5
---	-----



**VOLUME 3**

**Analog**

**Application-Specific Functions**

---

**PID Control**

---

**Process Control**

---

**Weighing**

**A**  
**B**  
**C**  
**D**

---

**VOLUME 1**

**Application-Specific Common Functions**

**Application-Specific Functions**

**Discrete**

**AS-i Bus Setup**

**Man-Machine Interface**

---

**VOLUME 2**

**Counting**

**Application-Specific Functions**

**Axis Control**

**Stepper Motor Axis Control**

---



<b>Section</b>	<b>Page</b>
<b>1 Analog function configuration</b>	<b>1/1</b>
1.1 Introduction	1/1
1.2 The configuration editor	1/2
1.2-1 Accessing the configuration editor	1/2
1.3 Configuring in-rack analog modules	1/3
1.3-1 Choosing the modules	1/3
1.3-2 Maximum number of in-rack analog channels in a configuration	1/3
1.3-3 Accessing the parameter settings of an analog module	1/4
1.4 Configuring distributed analog modules	1/6
1.4-1 Accessing the FIPIO configuration screen	1/6
1.4-2 Selecting a FIPIO connection point	1/6
1.4-3 Selecting the device to be connected	1/7
1.4-4 Accessing parameter settings for channels	1/8
1.4-5 Using OTHER_xxx references (Momentum)	1/9
<b>2 Setting the channel parameters on an analog module</b>	<b>2/1</b>
2.1 Presentation	2/1
2.2 Displaying the channel parameters	2/2
2.3 Modifying the channel parameters	2/3
2.4 Parameters of analog modules	2/4
2.4-1 Parameters of in-rack analog modules	2/4
2.4-2 Parameters of TBX distributed analog modules	2/6
2.4-3 Parameters of Momentum distributed analog modules	2/7

<b>Section</b>	<b>Page</b>
2.5	Modifying the parameters of TSX and TBX analog inputs 2/8
2.5-1	Modifying the input range 2/8
2.5-2	Modifying the task associated with the input channel 2/8
2.5-3	Modifying the display format 2/9
2.5-4	Modifying the filtering value 2/10
2.5-5	Modifying the channel scan cycle 2/10
2.5-6	Modifying terminal block detection 2/11
2.5-7	Modifying the channels used 2/11
2.5-8	Modifying the wiring check 2/12
2.5-9	Modifying the under/overrun check 2/12
2.5-10	Selecting event processing 2/13
2.5-11	Selecting the event processing number 2/13
2.5-12	Modifying the threshold values 2/13
2.5-13	Cold junction compensation 2/14
2.5-14	High precision mode 2/15
2.6	Modifying the parameters of TSX and TBX analog outputs 2/16
2.6-1	Modifying the output range 2/16
2.6-2	Modifying the task associated with the output 2/16
2.6-3	Modifying the fallback mode 2/17
2.6-4	Modifying the range under/overrun check 2/17
2.6-5	Selecting the output power supply 2/18
2.6-6	Modifying the power supply fault check 2/18
2.6-7	Modifying terminal block detection 2/18
2.7	Modifying the parameters of Momentum modules 2/19
2.7-1	Selecting the task 2/19
2.7-2	Modifying parameters 2/19
2.8	Confirming the configuration 2/20
2.8-1	Confirming after modification 2/20
2.8-2	Global reconfiguration 2/20

<b>Section</b>	<b>Page</b>
<b>3 Debug function</b>	<b>3/1</b>
3.1 Introduction to the Debug function	3/1
3.2 Displaying the channel parameters	3/1
3.3 Displaying the module diagnostics	3/3
3.4 Removing module channel forcing	3/4
3.5 Adjusting a channel	3/4
3.5-1 Displaying the detailed channel diagnostics	3/5
3.5-2 Modifying the filter value	3/6
3.5-3 Forcing/removing channel forcing	3/7
3.5-4 Aligning an input channel	3/8
3.5-5 Modifying output fallback value	3/9
<b>4 Calibration function</b>	<b>4/1</b>
4.1 Introduction to the Calibration function	4/1
4.2 Description of the calibration screen	4/2
4.3 Calibrating TSX AEY 800 / 801 / 1600 / TBX AES 400 / TBX AMS 620 modules	4/3
4.4 Calibrating the TSX AEY 414 module	4/5
4.4-1 Recalibrating the analog input module	4/5
4.4-2 Recalibrating the current source for a channel	4/7
4.5 Calibrating TSX AEY 1614 modules	4/8
4.5-1 Recalibrating the analog input module	4/8
4.5-2 Recalibrating the current source for a channel	4/10

<b>Section</b>	<b>Page</b>
<b>5 Bits and words associated with the analog function</b>	<b>5/1</b>
5.1 Addressing in-rack analog module objects	5/1
5.2 Addressing distributed analog module objects	5/2
5.3 Language objects associated with the analog I/O	5/3
5.3-1 Implicit exchange objects associated with inputs	5/3
5.3-2 Explicit exchange objects associated with inputs	5/5
5.3-3 Implicit exchange objects associated with outputs	5/7
5.3-4 Explicit exchange objects associated with outputs	5/7
5.4 %CH language objects	5/9
<b>6 Index</b>	<b>6/1</b>

---

---

## 1 Analog function configuration

---

### 1.1 Introduction

---

This part concerns :

- analog I/O modules mounted in rack
- distributed analog I/O modules connected on the FIPIO bus.  
To access distributed modules, the configured processor must have an integrated FIPIO link.

Before creating an application program, the physical operating context in which it will be executed must be defined, ie. the rack and the modules located in the rack : power supply, processor and discrete I/O and application-specific modules (analog, communication, counter, etc).

The use of analog I/O means that the analog channel parameters used must also be defined (input range, filter level, etc).

To do this, PL7 Junior software offers the **configuration editor**.

In online application operation, this editor also offers debugging functions which are used :

- to adjust certain parameters (for example, filtering) in order to best adapt them to the application,
- to carry out sensor measurement alignment, to compensate their offsets,
- to recalibrate the module.

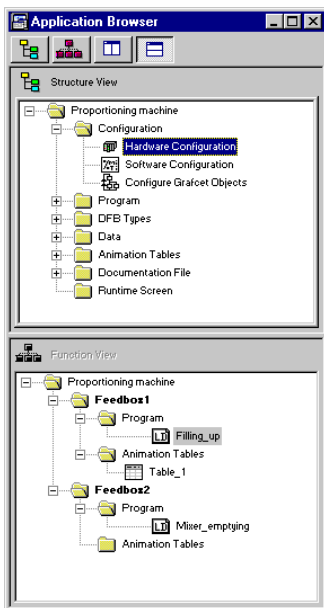
---

## 1.2 The configuration editor

---

### 1.2-1 Accessing the configuration editor

Use the Application Browser to select the Station folder and then the Configuration folder, then double-click on the "Hardware configuration" icon.



If the Application Browser is not displayed :

- click on the Application Browser icon
- or select the **Tools/Application Browser** menu

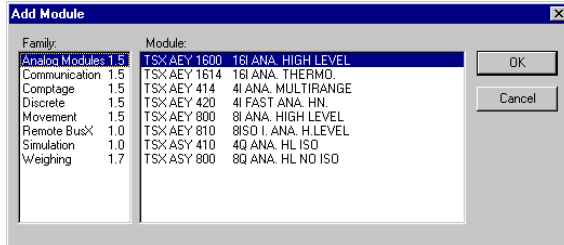




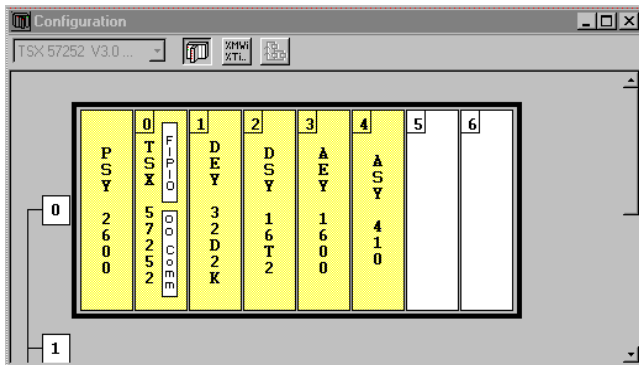
## 1.3 Configuring in-rack analog modules

### 1.3-1 Choosing the modules

This is performed by double-clicking on the position of the module to be configured (for example 4). This displays the following dialog box :



In the **Family** field, select the type of module (for example Analog), then, in the **Module** field, the reference of the module to be configured (for example TSX ASY 410). After confirming with **OK**, the module is declared in its position (this is framed and contains the module reference).



### 1.3-2 Maximum number of in-rack analog channels in a configuration

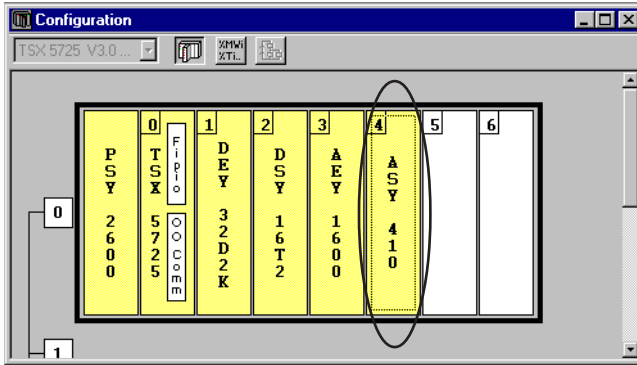
- 24 channels with a TSX/PMX/PCX 57-1• processor,
- 80 channels with a TSX/PMX 57-2• processor,
- 128 channels with a TSX/PMX/PCX57-3• processor,
- 256 channels with a TSX/PMX 57-4• processor.

#### Notes

- In order to delete a module from its position, click on it to select it then press the <Del> key, which displays a dialog box. Then confirm the deletion of the module.
- If TSX ASY 800 modules are supplied by the internal 24V power supply, the number of modules is reduced to 2 per rack (with a double format power supply).

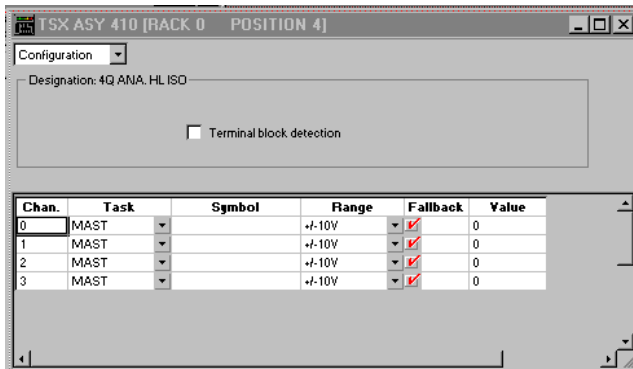
### 1.3-3 Accessing the parameter settings of an analog module

To set the parameters for the module channels, double-click on the position of the selected module in the rack, or select **Utilities/Open Module**

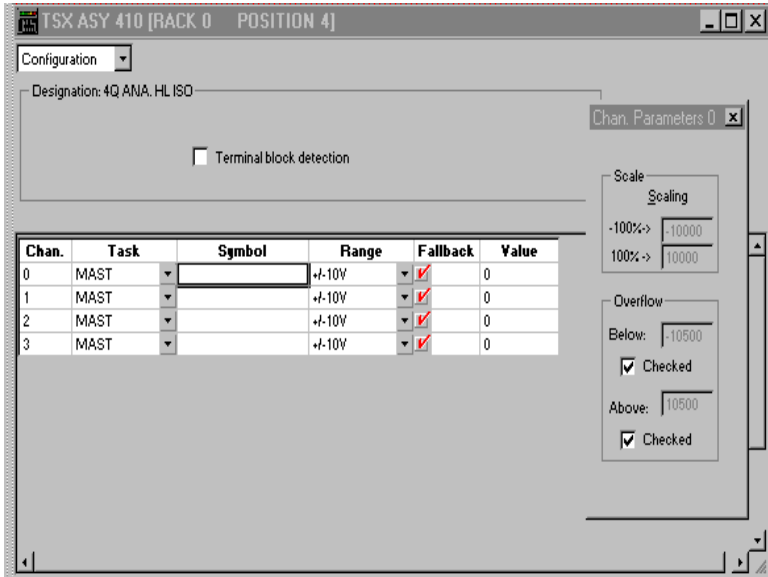


Procedure :


- 1 Click on the position of the module for which parameters are to be set (5 for example), the following parameter setting screen appears :



- Set the parameters for each of the channels. For some modules, a dialog box can be used to select additional parameters.



- Confirm the configuration screen by closing the parameter setting window.

To confirm, click on the  icon

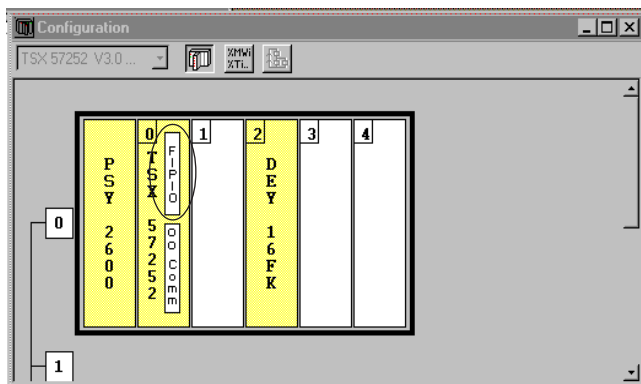
or select **Edit / OK**  
Ctrl+W

or select **OK** from the shortcut menu.

## 1.4 Configuring distributed analog modules

### 1.4-1 Accessing the FIPIO configuration screen

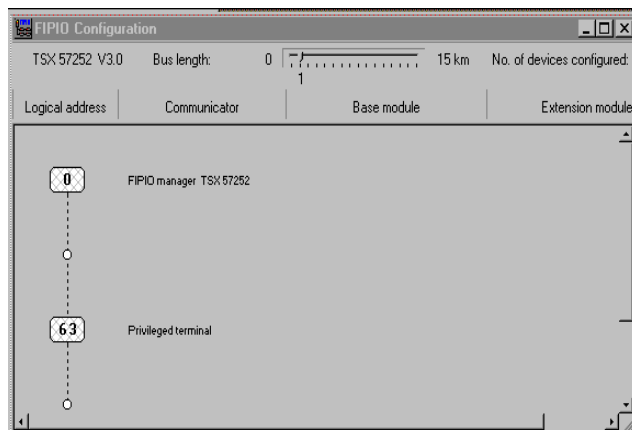
To access the FIPIO configuration screen, **double-click** on the FIPIO zone of the processor (which must have an integrated FIPIO link) :



Maximum configuration on FIPIO bus : see Communication manual, part H.

### 1.4-2 Selecting a FIPIO connection point

The screen shows the FIPIO bus, with the addresses occupied in the Logical Address column. On start-up, if no device is configured, only the first and the last connection point, together with connection point 63 are displayed (addresses 0 and 63 are reserved by the system).



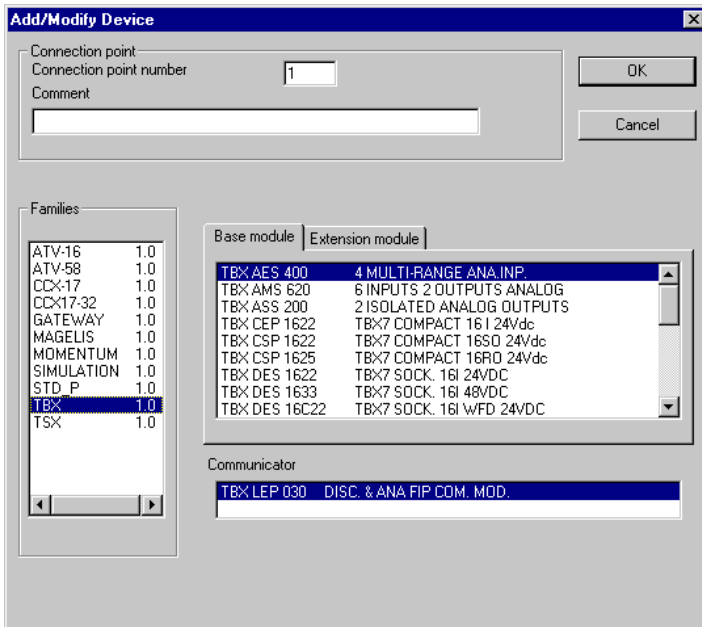
### 1.4-3 Selecting the device to be connected

**Double-click** in the "Logical Address" column at the place where the device is to be connected.

Enter the address of the connection point, enter a comment (optional).

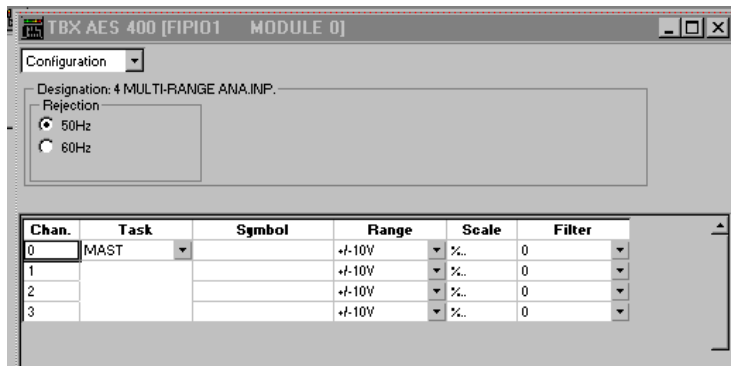
To select a device, select the family (TBX or MOMENTUM), then the base module and the extension module, then the communicator.

For MOMENTUM, if the reference of the sub-base used is not offered, select one of the following references : OTHER FED, OTHER FED\_P, OTHER FSD, or OTHER FSD\_P, (see section 1.4-5).




## 1.4-4 Accessing parameter settings for channels

To set the parameters for the module channels, double-click on the position of the selected module on the bus, or select **Utilities/Open Module**.



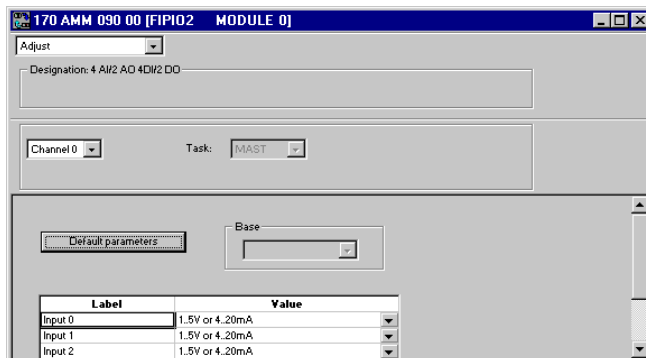
Procedure :

- 1 Click on the position of the module for which parameters are to be set (2 for example), the following parameter setting screen appears :
- 2 Set the parameters for each of the channels. For some modules, a dialog box can be used to select additional parameters.
- 3 Confirm the configuration screen by closing the parameter setting window.

To confirm, click on the  icon  
or select **Edit / OK**  
Ctrl+W  
or select **OK** from the shortcut menu.

### Comment

For Momentum modules, go to the adjustment screen.

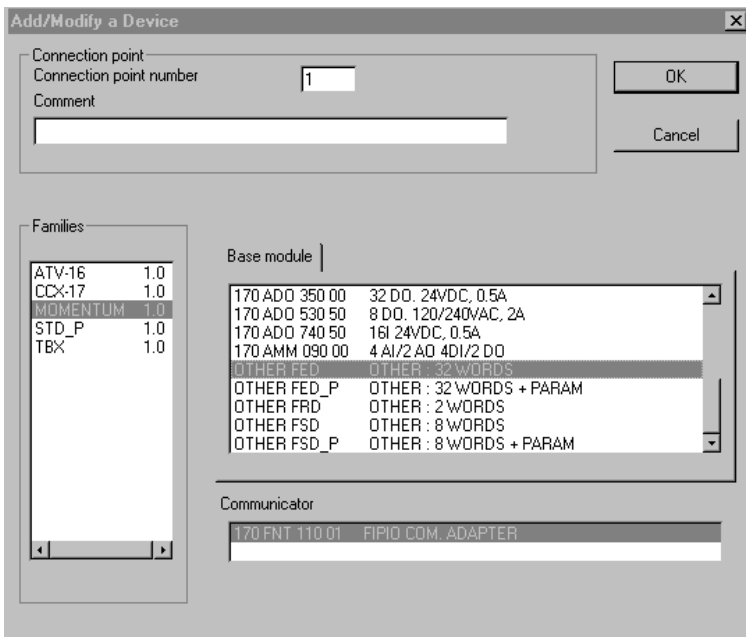


### 1.4-5 Using OTHER\_xxx references (Momentum)

If the reference of the sub-base used is not offered in the configuration tool, select one of the following references OTHER FED, OTHER FED\_P, OTHER FSD, or OTHER FSD\_P according to the criteria below :

	Sub-base without parameters	Sub-base with parameters
Number of input words $\leq$ 8 and Number of output words $\leq$ 8	OTHER FSD	OTHER FSD_P
Number of input words $>$ 8 or Number of output words $>$ 8	OTHER FED	OTHER FED_P

To find out the number of input and output words required for the sub-base used, refer to the installation manual for the sub-base.



If the sub-base uses parameters, the format and syntax of the parameters to be entered will be found in the FIPIO communicator setup manual.

Configuration

Designation: OTHER: 32 WORDS + PARAM

Channel 0 Task: MAST I/O Data type: WORD

Default configuration Base

Parameters	Symbol	Value
%KW0.2.1v0.0.0		2#0
%KW0.2.1v0.0.1		0
%KW0.2.1v0.0.2		0
%KW0.2.1v0.0.3		0
%KW0.2.1v0.0.4		0
%KW0.2.1v0.0.5		0
%KW0.2.1v0.0.6		0
%KW0.2.1v0.0.7		0
%KW0.2.1v0.0.8		0



## 2 Setting the channel parameters on an analog module

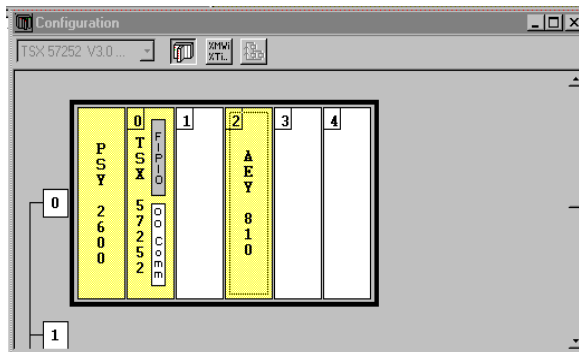
### 2.1 Presentation

Module input and output channels contain configuration parameters which can be displayed and modified by the **configuration** editor. These parameters offer the following functions :

- assignment of channels to a task,
- filtering time for fast inputs,
- fallback mode for outputs on a fault,
- etc

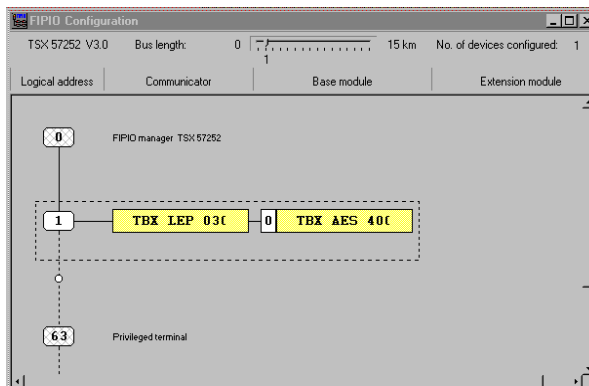
#### For in-rack analog modules :

The parameter setting screen for the module channels is accessed by double-clicking on the module to be configured in the rack.



#### For distributed analog modules :

The parameter setting screen for the module channels is accessed by double-clicking on the FIPIO bus module to be configured.

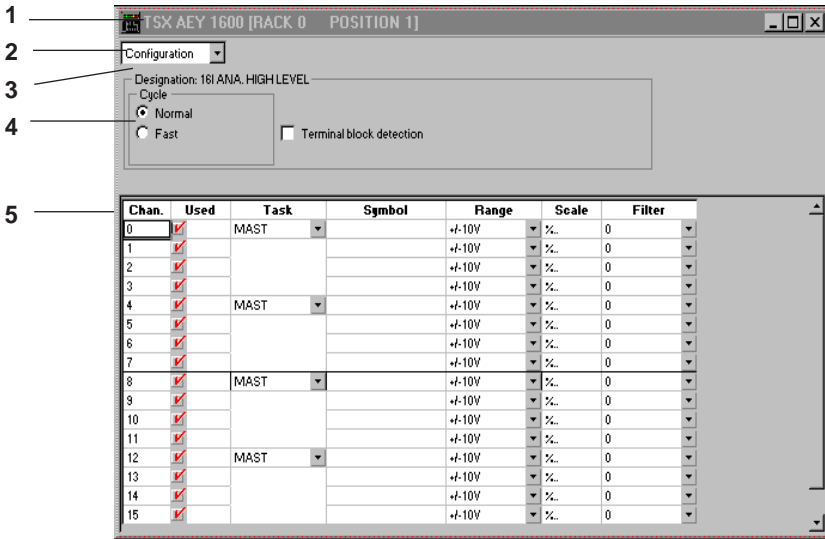


If no parameters are modified, the channels are configured according to the preset **default** parameters.

## 2.2 Displaying the channel parameters

The configuration screen of the module selected in the rack or on the FIPIO bus displays the parameters associated with the input or output channels.

This screen access **display** and **modification** of parameters in offline mode, and **Debugging** in online mode.



### Description

(1) The **title bar** shows the reference of the module selected and its physical position as well as the rack number.

(2) Method of use : **Configuration** and **Adjustment** (for Momentum modules) in progress; in online mode the window also provides access to **Debugging** (diagnostics) and to calibration for input modules.

(3) **Designation** of the selected module :

Example: 16 I. ANA. HIGH LEVEL : 16 high level analog inputs,

(4) Displaying the module zone is optional. Access is via the **View/Module Zone** command.

(5) **Channel selection boxes** :

All **channels** and associated **symbols**. The name (symbol) is defined by the user via the variables editor. The scroll bar on the right is used to display all the module channels, both up and down the list.

For Momentum modules, a channel parameter entry zone can be accessed from the **View/Channel Zone** menu

---

## 2.3 Modifying the channel parameters

---

The configuration editor has a number of functions for facilitating the entry or modification of the module parameters.

### Shortcut menus

These can be accessed by right-clicking with the mouse, and provide fast access to the main commands.

at table cell level :

Copy Parameters
Paste Parameters
Properties

at module zone level (outside a table) :

Undo
OK
Animate

### Selecting a channel

A channel is selected by clicking on the number of the channel required in the channel column.

### Selecting a channel parameter

Click directly in the associated cell.

### Selecting several consecutive cells

Click on the first cell and, holding the mouse button down, drag the mouse upwards or downwards, releasing the button when the last cell is reached.

### Copy/paste

To copy : select the cell or channel to be copied and select the "Copy Parameters" command from the shortcut menu.

To paste : select the cell(s), or the channel to be pasted and select the "Paste Parameters" command from the shortcut menu.

This function is not available on Momentum modules.

## 2.4 Parameters of analog modules

### 2.4-1 Parameters of in-rack analog modules

The parameters of each of the analog modules are as follows (the default parameters are underlined in the tables) :

Module	TSX AEY 1600	TSX AEY 800	TSX AEY 810	TSX AEY 420
Numberofchannels	16 inputs	8 inputs	8 inputs	4 inputs
Channel used	<u>Yes</u> / No	<u>Yes</u> / No	<u>Yes</u> / No	<u>Yes</u> / No
Scan cycle	<u>Normal</u> Fast	<u>Normal</u> Fast	<u>Normal</u> Fast	/
Range	<u>±10 V</u> 0..10 V 0..5 V 1..5 V 0..20 mA 4..20 mA	<u>±10 V</u> 0..10 V 0.5 V 1.5 V 0..20 mA 4..20 mA	<u>±10 V</u> 0..10 V 0.5 V 1..5 V 0..20 mA 4..20 mA	<u>±10 V</u> 0..10 V 0.5 V 1..5 V 0..20 mA 4..20 mA
Filtering	<u>0</u> ..6	<u>0</u> ..6	<u>0</u> ..6	/
Display . standard . high level	<u>%</u> .. User	<u>%</u> .. User	<u>%</u> .. User	<u>%</u> .. User
Task associated with the channel	<u>Mast</u> Fast	<u>Mast</u> Fast	<u>Mast</u> Fast	<u>Mast</u> Fast
Terminal block detection	Yes / <u>No</u>	Yes / <u>No</u>	Yes / <u>No</u>	Yes / <u>No</u>
Range under/overrun check . lower . upper	/ /	/ /	<u>Yes</u> / No <u>Yes</u> / No	<u>Yes</u> / No <u>Yes</u> / No
Range under/overrun limits . lower limit . upper limit	/ /	/ /	<u>min-12.5%</u> (1) <u>max+12.5%</u> (1)	<u>min-12.5%</u> (1) <u>max+12.5%</u> (1)
Threshold 0	/	/	/	<u>0</u>
Threshold 1	/	/	/	<u>0</u>
Event number	/	/	/	0 ..63
Event processing	/	/	/	Yes / <u>No</u>

(1) see maximum and minimum limits in the setup manual.

Module	TSX AEY 414	TSX AEY 1614	TSX ASY 410	TSX ASY 800
Numberofchannels	4 inputs	16 inputs	4 outputs	8 outputs
Channel used	/	<b>Yes</b> / No	/	/
Scan cycle	/	<b>Normal</b> Fast	/	/
Range	<b>±10 V</b> 0..10 V / ±5 V 0..5 V / 1..5 V 0..20 mA/4..20 mA Pt100 / Pt1000 Ni1000 Thermocouples B, E, J, K, L, N, R, S, T, U -13..63 mV 0..400 Ω/0..3850 Ω	<b>ThermocouplesK,</b> E, J, K, L, N, R, S, T, U -80..+80 mV	<b>±10 V</b> 0..20 mA 4..20 mA	<b>±10 V</b> 0..20 mA 4..20 mA
Filtering	<b>0..6</b>	<b>0..6</b>	/	/
Display . high level	<b>%..</b> User	<b>%..</b> User	<b>%..</b> (cannot be modified)	<b>%..</b> (cannot be modified)
. temp. probes . thermocouples	<b>1/10 °C</b> /1/10 °F %..	<b>1/10 °C</b> /1/10 °F %..	/	/
Task associated with the channel	<b>Mast</b> Fast	<b>Mast</b> Fast	<b>Mast</b> Fast	<b>Mast</b> Fast
Term. blk detect.	Yes / <b>No</b>	Yes / <b>No</b>	Yes / <b>No</b>	Yes / <b>No</b>
Fallback	/	/	<b>Fallback to 0</b> Maintain Fallback to a value	<b>Fallback to 0</b> Maintain Fallback to a value
Wiring check	Active / <b>Inactive</b>	Active / <b>Inactive</b>	/	/
24Vsupp.monitoring	/	/	/	Yes / <b>No</b>
Cold junction compensation	<b>Internal</b> External	<b>Telefast</b> /Pt100 cold junction reading	/	/
Power supply	/	/	/	Internal / <b>external</b>
Range under/ overrun check				
. lower	/	<b>Yes</b> / No	<b>Yes</b> / No	<b>Yes</b> / No
. upper	/	<b>Yes</b> / No	<b>Yes</b> / No	<b>Yes</b> / No
Range under/ overrun limits				
. lower limit	/	<b>min-12.5%</b> (1)	/	/
. upper limit	/	<b>max+12.5%</b> (1)	/	/
High precision	/	<b>Yes</b> / No	/	/

(1) see maximum and minimum limits in the setup manual.

## 2.4-2 Parameters of TBX distributed analog modules

The parameters of each of the analog modules are as follows (the default parameters are underlined in the tables) :

Module	TBX AES 400	TBX AMS 620	TBX ASS 200
Numberofchannels	4 inputs	6 inputs/ 2 outputs	2 outputs
Range	<u>±10 V</u> ±5 V 0..20 mA 4..20 mA Pt100 / Pt1000 Ni1000 Thermocouples B, E, J, K, N, R, S, T, ±20mV, ±50mV ±200mV, ±500mV	<u>±10 V</u> 0.5 V (1) 0..20 mA 4..20 mA	<u>±10 V</u> 0..20 mA 4..20 mA
Filtering	<u>0..6</u>	<u>0..6</u> (1)	/
Display			
. standard	<u>%..</u>	<u>%..</u> (1)	<u>%..</u>
. high level	User	User	
. temp. probes	<u>1/10 °C</u>		
. thermocouples	1/10 °F <u>%..</u>		
Task associated with the channel	<u>Mast</u> Fast	<u>Mast</u> Fast	<u>Mast</u> Fast
Sensormonitoring	Active / <u>Inactive</u>	/	/
Rejection	<u>50Hz</u> / 60Hz	/	/
Fallback	/	<u>Fallback to 0</u> (2) Maintain Fallback to a value	<u>Fallback to 0</u> Maintain Fallback to a value

(1) inputs only

(2) outputs only

### 2.4-3 Parameters of Momentum distributed analog modules

The parameters of each of the analog modules are as follows (the default parameters are underlined in the tables) :

Module	170 AAI 030 00	170 AAI 140 00	170 AAI 520 40	170 AAO 12000/92100	170 AMM 090 00
<b>Number of channels</b>	8 inputs	16 inputs	4 inputs	4 outputs	4 inputs 2 outputs
<b>Range</b>	±10 V  <u>±5 V or</u> ± 20 mA  1.5 V or <u>4..20 mA</u>  Inactive	±10 V ±5 V <u>4..20 mA</u> Inactive	±25 mV <u>±100 mV</u> EIC Pt100 EIC Pt1000 US/JIS Pt100 US/JIS Pt1000 Ni100 Ni1000  Thermocouples B, E, J, K, N, R, S, T	<b>170AAO12000</b> ±10 V 0..20 mA  <b>170AAO92100</b> ±10 V 4..20 mA	<b>Inputs</b> ±10 V  <u>±5 V or</u> ± 20 mA  1.5 V ou <u>4..20 mA</u>  Inactive <b>Outputs</b> ±10 V <u>0..20 mA</u>
<b>Display</b> . temp. probes . thermocouples		/	1/10°C 1/10°F		
<b>Task associated with all channels</b>	<u>Mast</u> Fast	<u>Mast</u> Fast	<u>Mast</u> Fast	<u>Mast</u> Fast	<u>Mast</u> Fast
<b>Wiring check</b>	/	/	Active / <u>Inactive</u>	/	/
<b>Output fallback</b>	/	/	/	Fallback to 0 Fallback to FS <u>Maintain</u>	Fallback to 0 Fallback to FS <u>Maintain</u>
<b>Cabling</b>	/	/	2- or 4-wire 3-wire(1)	/	/

(1) temperature probes only

## 2.5 Modifying the parameters of TSX and TBX analog inputs

### 2.5-1 Modifying the input range

This is performed by clicking in the Range column, in the cell corresponding to the channel to be modified. A pull-down list box is used to select the new input range for the channel :  $\pm 10$  V, 0..10 V, 0..5 V, 4..20 mA, etc.

Chan.	Used	Task	Symbol	Range	Scale	Filter
0	✓	MAST		$\pm 10$ V	%..	0
1	✓			$\pm 10$ V	%..	0
2	✓			0..10V	%..	0
3	✓			0..5V	%..	0
4	✓	MAST		1.5V	%..	0
5	✓			0..20mA	%..	0
6	✓			4..20mA	%..	0
7	✓			$\pm 10$ V	%..	0
7	✓			$\pm 10$ V	%..	0

### 2.5-2 Modifying the task associated with the input channel

This is performed via a pull-down list box which is used :

- for 4 consecutive channels : channels 0 to 3, channels 4 to 7, channels 8 to 11 or channels 12 to 15 (TSX AEY 800 / 810 / 1600 / 1614) ,
- for 2 consecutive channels (TSX AEY 420, TSX 800),
- channel by channel (TSX AEY 414),

to define the task at the start of which these channels will be read : MAST task or FAST task.

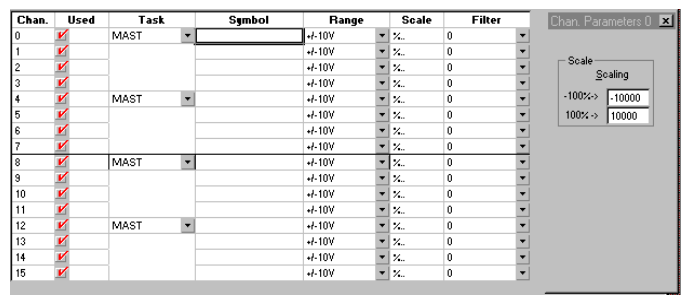
- inputs and outputs for the complete connection point (base + extension) on FIP (TBX AES 400 and TBX AMS 620)

Chan.	Task	Symbol	Range	Scale	Filter
0	MAST		$\pm 10$ V	%..	0
1	MAST		$\pm 10$ V	%..	0
2	FAST		$\pm 10$ V	%..	0
3	MAST		$\pm 10$ V	%..	0



### 2.5-3 Modifying the display format

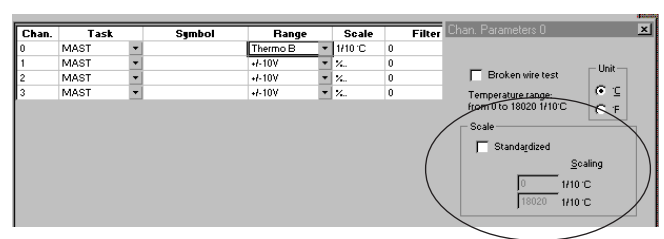
This is performed by double-clicking in the Scale column, in the cell corresponding to the channel to be modified. The Properties dialog box is then used to define the display limits. If the default values (standard display 0... 10000 or ±10000) are selected, the cell shows %..., otherwise (user display), it shows **User**.



In the case of a TSX AEY 414/1614 or TBX AES 400 module and if the user chooses a thermocouple or a temperature probe range, two command buttons are offered in the Properties dialog box for defining the unit in which the channel measurements are displayed : °C (Celsius) or °F (Fahrenheit).

The temperature range can be the default of the thermocouple or the temperature probe chosen, defined in tenths of a degree (for example, - 600 to + 1100 tenths of a °C for an Ni1000 probe). In this case, the Scale cell for the channel shows **1/10 °C** or **1/10 °F**, depending on the temperature unit chosen.

If the **Standard** check box is activated, the display is standard 0..100 %, with default upper and lower limits (for example, - 600 tenths of a °C for 0% to + 1100 tenths of a °C for 100%, with an Ni1000 probe). If at least one of the limits is modified by the user, the display becomes user (for example, 0 to + 1100 tenths of a °C for an Ni1000 probe). When the temperature scale is standard (with default or user limits), the channel parameters display zone shows %..., regardless of which temperature unit is chosen.



## 2.5-4 Modifying the filtering value

This is performed by clicking in the Filter column, in the cell corresponding to the channel to be modified. A pulldown list box is used to select the new filtering value for the channel : 0 (no filtering), 1 and 2 (low level of filtering), 3 and 4 (medium level of filtering), 5 and 6 (high level of filtering). The efficiency value of the chosen filter (coefficient  $\alpha$ ) and the associated response time are then displayed in the status bar at the bottom of the screen.

Chan.	Task	Symbol	Range	Scale	Filter
0	MAST		Thermo B	1/10 °C	0
1	MAST		+/-10V	%..	1
2	MAST		+/-10V	%..	2
3	MAST		+/-10V	%..	3
					4
					5

## 2.5-5 Modifying the channel scan cycle

This is performed via two command buttons which select the channel scan cycle :

- **Normal** cycle : all the channels are scanned, even the channels which are declared unused. The module scan time is 1 ms (TSX AEY 420), 27 ms (TSX AEY 800), 29.7 ms (TSX AEY 810), 51 ms (TSX AEY 1600), 480 ms (TSX AEY 1614)
- **Fast** cycle : only the channels which are declared used are scanned. The module scan time is :
  - $T_v \times (1 + n)$  ms, where  $n$  is the number of channels used,  $T_v$  scan time for one channel (3 ms for TSX AEY 800/1600 and 3.3 ms for TSX AEY 800).
  - $T_v \times n$  ms, where  $n$  is the number of channels used,  $T_v$  scan time for one channel (70 ms for TSX AEY 1614).

This option is used to reduce the scan time for a module when not all the channels are used.

**Filtering is not taken into account in Fast mode.**



**2.5-6 Modifying terminal block detection**

This is performed via the check box **Terminal Block Detection**. When this box is checked, the module monitors the presence of the SubD connector(s) or the terminal block and signals a fault when the latter is missing (disconnected). For TSX AEY 1600/1614 modules, fitted with 2 SubD connectors, a terminal block fault is signaled **if at least one channel is used** on the missing connector.



**2.5-7 Modifying the channels used**

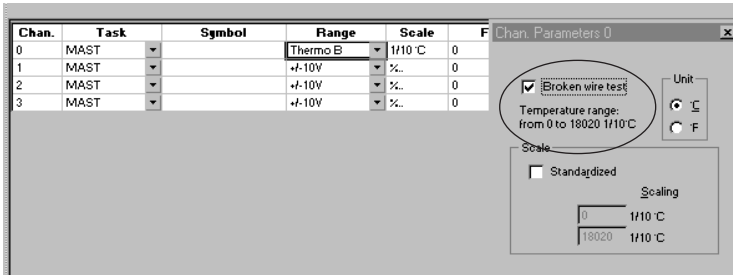
This is performed via the checkbox opposite each channel number. When this box is checked, the channel is declared used and the values measured are "fed back" to the task assigned to the channel.

When a channel is unused, the line is grayed out. The value 0 is fed back to the application program and the faults on that channel (range under/overrun, etc) are inactive.

Chan.	Used	Task	Symbol	Range	Scale	Filter
0	<input checked="" type="checkbox"/>	MAST		+/-10V	%..	0
1	<input checked="" type="checkbox"/>			+/-10V	%..	0
2	<input checked="" type="checkbox"/>			+/-10V	%..	0
3	<input type="checkbox"/>			Not used	%..	0

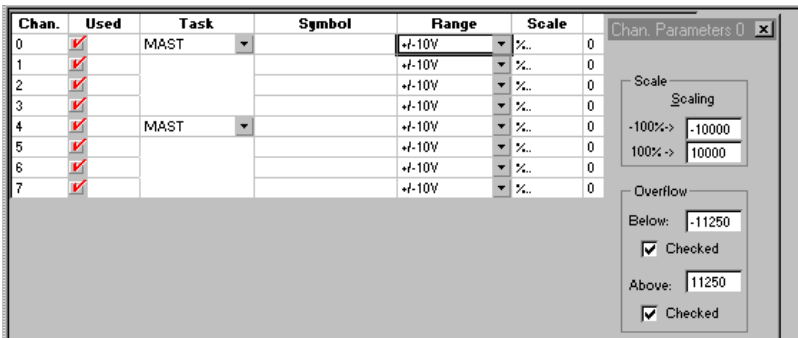
## 2.5-8 Modifying the wiring check

This is performed by double-clicking in one of the cells corresponding to the channel to be modified, once the temperature range has been selected. The Properties dialog box is used to access the **Wiring Fault Check** checkbox. When this box is checked, the module monitors the sensor link and signals a fault when a short-circuit or an open circuit occurs on the corresponding channel.



## 2.5-9 Modifying the under/overrun check

This is performed by double-clicking in one of the cells corresponding to the channel to be modified. The Properties dialog box is used to access the **Check** checkboxes in the Overflow zone. When the box is checked, the corresponding under/overrun check is enabled. The associated entry field is used to specify the value above which an under/overrun is detected.



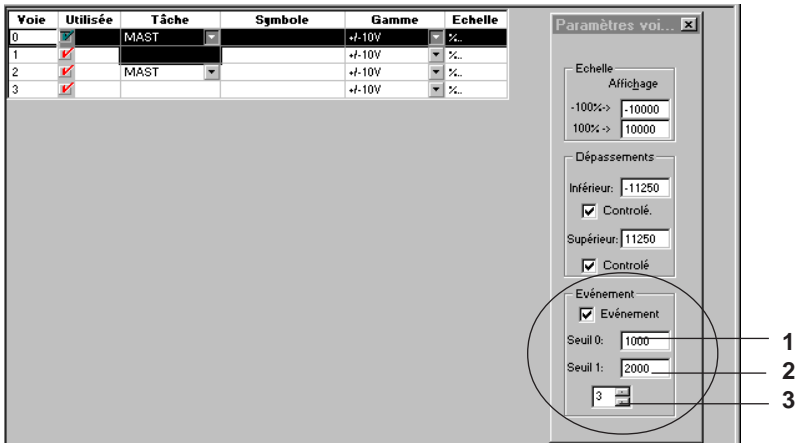
### 2.5-10 Selecting event processing

This is performed via the Properties dialog box.

To activate event processing, check the Event checkbox **(1)**.

**Reminder** (see setup manual) :

Event processing (the event number is shown in this screen) is activated when one of the thresholds is crossed.



### 2.5-11 Selecting the event processing number

This is performed via the Properties dialog box (see screen above).

Enter the event processing number directly in the field **(3)** or use the up and down arrows to increment or decrement the number.

### 2.5-12 Modifying the threshold values

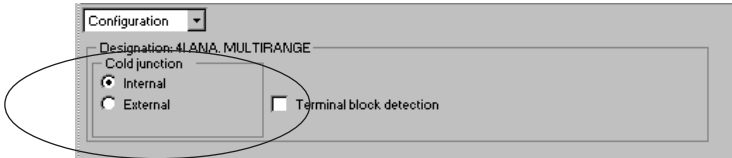
This is performed by double-clicking in one of the cells corresponding to the channel to be modified.

The Properties dialog box is used to access modification of the threshold values **(2)** (if the Event checkbox is checked **(1)**).

---

## 2.5-13 Cold junction compensation

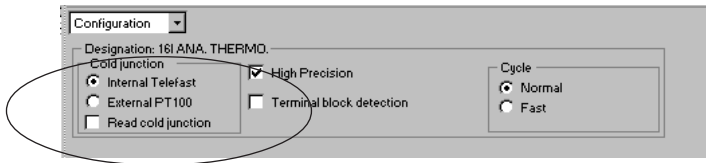
### TSX AEY 414



If a thermocouple range has been selected, two command buttons are used to select the type of cold junction compensation : Internal (by default) or External.

**For external cold junction, channel 0 is forced to the Pt100 range after confirmation.**

### TSX AEY 1614



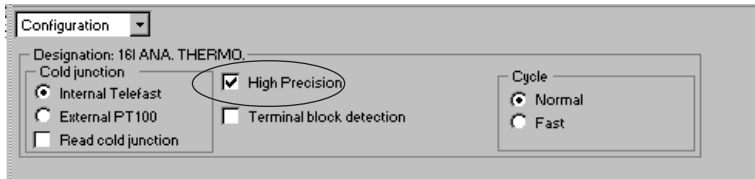
Two command buttons are used to select the type of cold junction compensation :

- Internal via Telefast (by default) : cold junction compensation is performed at the level of the Telefast terminal block, in this case it is possible, by checking the "Cold Junction Reading" box, to feed back the cold junction temperature value via channel 8, after confirming a warning message.
- External via PT100 : cold junction compensation is via a PT100 probe wired on channels 0 and 8. Channel 0 supplies the current to the PT100 probe, channel 8 measures the temperature.

### 2.5-14 High precision mode

This checkbox (checked by default) is used to select high precision mode. This mode gives greater precision in temperature measurements using a self-calibration procedure (see the characteristics section in the setup manual).

**Note** : this self-calibration procedure adds 70ms to each scan.



## 2.6 Modifying the parameters of TSX and TBX analog outputs

### 2.6-1 Modifying the output range

This is performed via a pull-down list box which is used to select the new channel output range :  $\pm 10$  V, 0..20 mA, 4..20 mA.

Chan.	Task	Symbol	Range	Fallback	Value
0	MAST		$\pm 10$ V	<input type="checkbox"/>	-
1			$\pm 10$ V	<input checked="" type="checkbox"/>	0
2	MAST		0..20mA	<input checked="" type="checkbox"/>	0
3			4..20mA	<input checked="" type="checkbox"/>	0
4	MAST		$\pm 10$ V	<input checked="" type="checkbox"/>	0
5			$\pm 10$ V	<input checked="" type="checkbox"/>	0
6	MAST		$\pm 10$ V	<input checked="" type="checkbox"/>	0
7			$\pm 10$ V	<input checked="" type="checkbox"/>	0

### 2.6-2 Modifying the task associated with the output

This is performed via a pull-down list box which is used :

- channel per channel (TSX AEY 414)
- by group of 2 channels (TSX ASY 800),  
to define the task at the end of which the outputs will be updated : **MAST** task or **FAST** task.
- inputs and outputs for the complete connection point (base + extension) on FIP (TBX ASS200 and TBX AMS620), etc.

Chan.	Task	Symbol	Range	Fallback	Value
0	MAST		$\pm 10$ V	<input type="checkbox"/>	-
1	MAST		$\pm 10$ V	<input checked="" type="checkbox"/>	0
2	FAST		$\pm 10$ V	<input checked="" type="checkbox"/>	0
3			$\pm 10$ V	<input checked="" type="checkbox"/>	0
4	MAST		$\pm 10$ V	<input checked="" type="checkbox"/>	0
5			$\pm 10$ V	<input checked="" type="checkbox"/>	0
6	MAST		$\pm 10$ V	<input checked="" type="checkbox"/>	0
7			$\pm 10$ V	<input checked="" type="checkbox"/>	0



### 2.6-3 Modifying the fallback mode

This is performed via a checkbox in the Fallback column. It is used to define the behavior of the outputs in the event of certain faults (communication fault) or when the associated task goes to STOP :

- **Fallback** of outputs to a user-definable value, default 0, (box checked),
- **Maintain** outputs at their value, the last calculated value or the forcing value if the output was forced, (box not checked).

When outputs fall back to a value other than 0 (default value), this is entered in the **Value** field and must be between -10000 and +10000

Chan.	Task	Symbol	Range	Fallback	Value
0	MAST		+/-10V	<input type="checkbox"/>	-
1			+/-10V	<input checked="" type="checkbox"/>	500
2	MAST		+/-10V	<input checked="" type="checkbox"/>	0
3			+/-10V	<input checked="" type="checkbox"/>	0
4	MAST		+/-10V	<input checked="" type="checkbox"/>	0
5			+/-10V	<input checked="" type="checkbox"/>	0
6	MAST		+/-10V	<input checked="" type="checkbox"/>	0
7			+/-10V	<input checked="" type="checkbox"/>	0

### 2.6-4 Modifying the range under/overrun check

This is performed by double-clicking in one of the cells corresponding to the channel to be modified. The Properties dialog box is used to access the **Check** checkboxes in the Overflow zone. When the box is checked, the corresponding under/overrun check is enabled. The associated entry field is used to specify the value above which an under/overrun is detected.

Chan.	Task	Symbol	Range	Fallback	Value
0	MAST		+/-10V	<input type="checkbox"/>	-
1			+/-10V	<input checked="" type="checkbox"/>	500
2	MAST		+/-10V	<input checked="" type="checkbox"/>	0
3			+/-10V	<input checked="" type="checkbox"/>	0
4	MAST		+/-10V	<input checked="" type="checkbox"/>	0
5			+/-10V	<input checked="" type="checkbox"/>	0
6	MAST		+/-10V	<input checked="" type="checkbox"/>	0
7			+/-10V	<input checked="" type="checkbox"/>	0

Chan. Parameters 3

Scale

Scaling

-100% -> -10000

100% -> 10000

Overflow

Below: -10500

Checked

Above: 10500

Checked

---

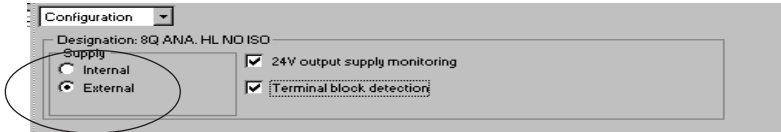
## 2.6-5 Selecting the output power supply

This is performed via 2 buttons :

- **internal** to select a 24V power supply internal to the module to supply the output channels,

**Caution : do not supply more than two TSX ASY 800 modules with a power supply from the same rack.**

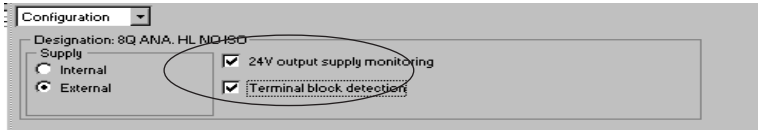
- **external** to select a 24V power supply external to the module to supply the output channels,



---

## 2.6-6 Modifying the power supply fault check

This is performed via the **Output 24V Power Supply Check** checkbox. When this box is checked, the module checks the presence of the external or internal 24V power supply, according to the selection made at power supply level.



---

## 2.6-7 Modifying terminal block detection

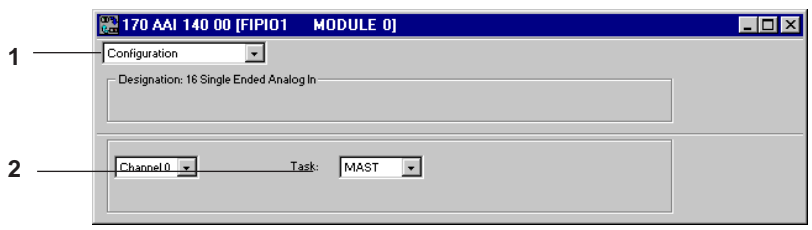
This is performed via the **Terminal Block Detection** checkbox. When this box is checked, the module checks the presence of the terminal block and signals a fault if it is missing (disconnected).



## 2.7 Modifying the parameters of Momentum modules

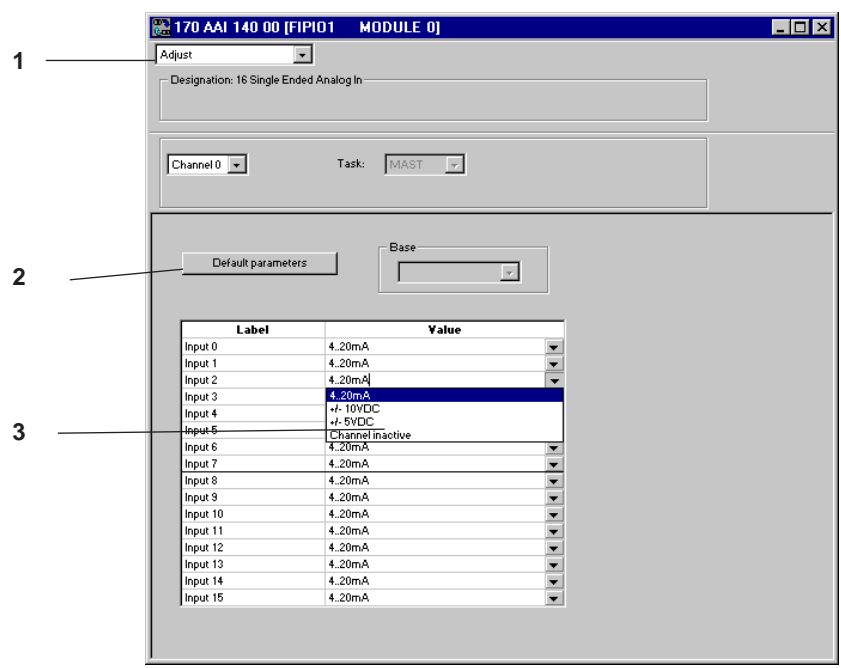
### 2.7-1 Selecting the task

This is performed in Configuration mode (1) via a pulldown list (2) which allows the user to de define the task at the start of which the input channels are read and the output channels are written.



### 2.7-2 Modifying parameters

This is performed in Adjust mode (1) via a pulldown list (3) (a list for each channel). The "Default parameters" button (2) resets all the parameters to their default values. The list of parameters for each module is given in section 2.4-3.



## 2.8 Confirming the configuration

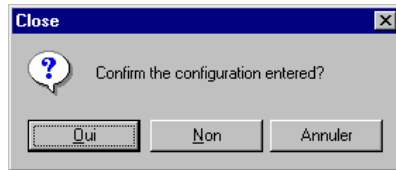
### 2.8-1 Confirming after modification

When quitting the function after modifying the module channel configuration parameters, the new configuration must be confirmed. This can be done in several ways :

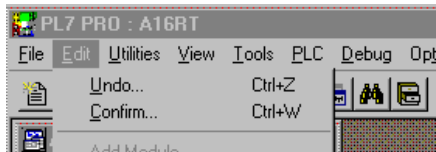
1. Confirm using the toolbar by clicking on the corresponding icon or by selecting the Confirm command in the shortcut menu.



2. Quit the function without confirming the parameters. This displays a dialog box which enables the user to confirm the new configuration.

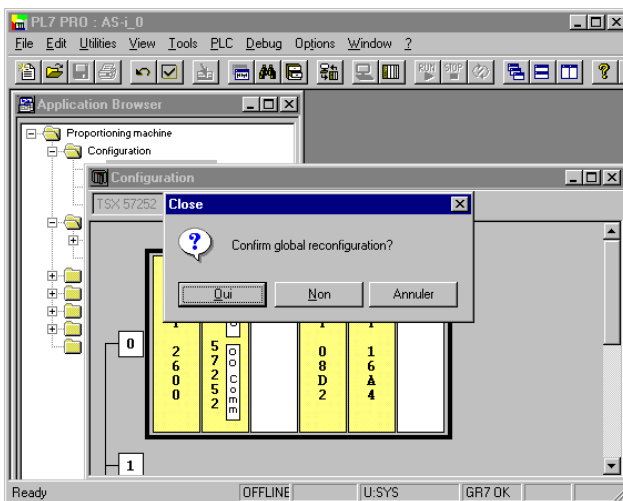


3. Pull down the PL7 Edit menu and select **Confirm**.



### 2.8-2 Global reconfiguration

Quitting the configuration editor after modifying all the configuration parameters of the channels on each module makes a global reconfiguration necessary. When the editor is closed, a dialog box enables this global reconfiguration to be confirmed.



---

Global reconfiguration is required in offline mode, so that the modifications confirmed for each module are accepted by the application.

This reconfiguration is performed :

- using the "Confirm" icon, the **Edit/Confirm** command, or the confirm command in the shortcut menu,
- by closing the configuration editor without global confirmation, and then confirming global reconfiguration.



### 3.1 Introduction to the Debug function

This function can only be accessed online (**PLC** menu, **Connect** command or click on the corresponding icon). For each analog module of the application, it displays the parameters of each channel (measurement value, filter value, etc) and accesses the diagnostics and adjustment of the selected channel (modification of filtering, forcing the channel, sensor alignment, etc).

The function also gives access to module diagnostics in the event of a fault.

The **Debug** function of an analog module is accessed by double-clicking on the Station, Configuration and then Hardware configuration icons in the Application Browser and in the module slot in the rack.

In online mode, the **Debug** function is selected by default. The pulldown dialog box in the command zone can be used to return to the **Configuration** function or to access the **Calibration** function.

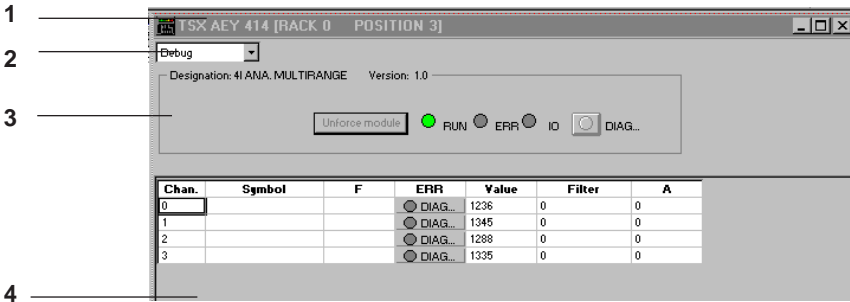


online mode



### 3.2 Displaying the channel parameters

This screen displays the selected module and the value and state of each of its channels in real time. It also gives access to the adjustment of certain channel parameters (forcing the input or output value, adjusting the filter value, etc).



- 1 This line shows the catalogue reference and the slot of the module in the PLC (rack and position).
- 2 This command zone shows the current function (**Debug** function) and selects, via a pulldown list box, the **Configuration** or **Calibration** function (for input modules).
- 3 This "module" level zone contains a short name for the module (for example 4 I. ANA MULTIRANGE) and its version. It also offers feedback of the module status indicator lamps (RUN, ERR, I/O), as well as two command buttons which, respectively :
  - Access the module diagnostics when it is faulty. This is signalled by the indicator lamp integrated into the button giving access to the diagnostics turning red.
  - Remove all the possible channel forcings,
- 4 This "channel" level zone displays in real time the value and status of each of the module channels :
  - **Channel** : number of the input or output channel.
  - **Symbol** : symbol defined by the user and associated with the language object for the channel. If the channel has no associated symbol, this field is empty.
  - **F** : forcing status of the channel : F if the channel is forced or no indication if the channel is not forced.
  - **ERR** : channel status : ERR indicates that the channel is faulty.
  - **Value** : value of the channel,
  - **Filter** : measurement filter value : 0 (no filtering), 1 and 2 (low level of filtering), 3 and 4 (medium level of filtering), 5 and 6 (high level of filtering).
  - **A** : alignment value (offset between the value before alignment and the desired value).
  - **Fallback** : on output cards, displays the fallback value on a fault or value maintained

#### Note

The measurement filter values, input alignment values and output fallback values are for CPU words. They may be different from those taken into account by the module.

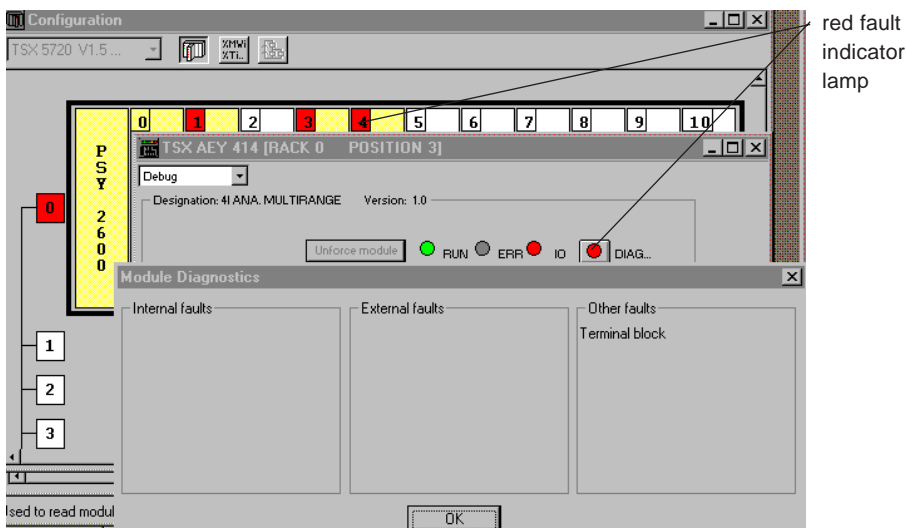


### 3.3 Displaying the module diagnostics

When a module is faulty, indicator lamps accessible in the configuration editor screens turn red :

- Module position indicator lamp on the screen which represents it (first screen of the configuration editor).
- Feedback of module ERR and I/O indicator lamps, in the "module" level zone,
- Indicator lamp integrated into the DIAG command button, also in the "module" level zone.

In addition, activating the DIAG command button gives access to the **Module Diagnostics** screen which displays the current module faults, classified according to their category : internal faults, external faults or other faults.



#### List of module faults

- **Internal faults** : Module failure
- **Other faults** : Faulty channel(s)  
Terminal block fault, self-test in progress, configuration fault, no module present or module not powered up, 24V power supply fault, module not factory calibrated.

#### Note

When a configuration fault occurs or a module is missing, the module diagnostics screen cannot be accessed. The following message appears on the screen : "Module missing or different from the one configured in this position".

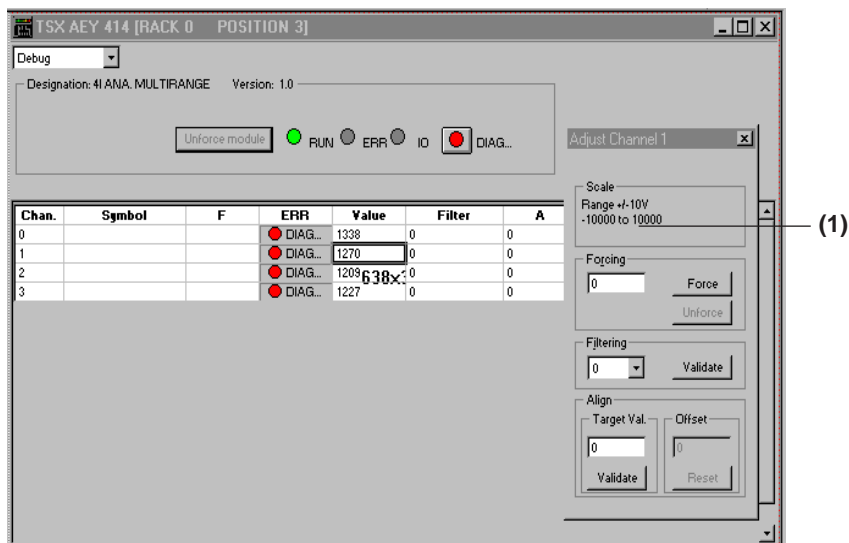
### 3.4 Removing module channel forcing

This is performed using the **Global Unforcing** command button which removes all module channel forcings.



### 3.5 Adjusting a channel

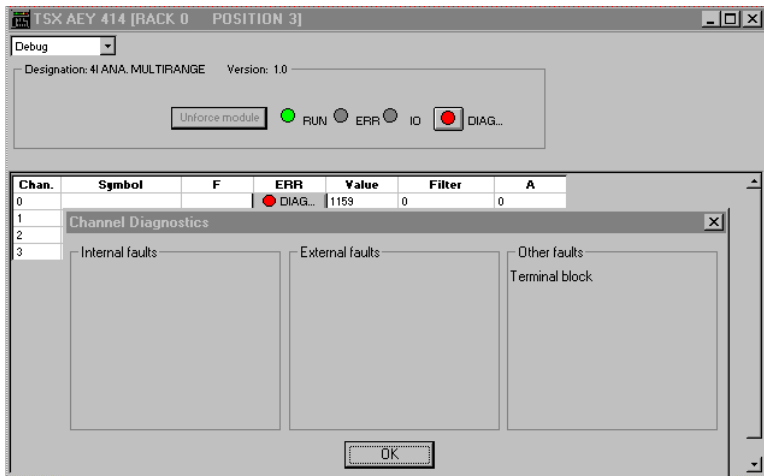
Channel adjustment is accessed by double-clicking on the channel number. The property dialog box **(1)** can then be used to set the parameters of the selected channel.



Choosing a new channel in the display zone gives access to the adjustment of its parameters in the property dialog box **(1)** which displays the number of the selected channel in realtime.

### 3.5-1 Displaying the detailed channel diagnostics

When a channel is faulty, the DIAG button in the **ERR** column becomes active. Activating this button then gives access to a "channel" diagnostics screen (identical to that for "module" diagnostics) which indicates the channel faults, classified according to their category : internal faults, external faults or other faults.



#### List of channel faults

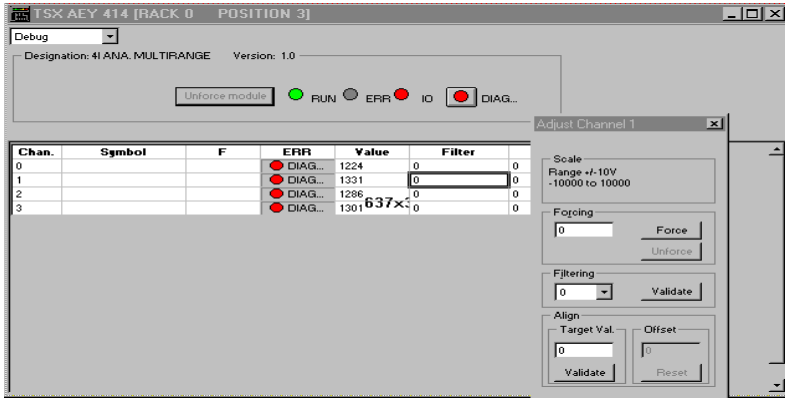
- **Internal faults :** Module failure
- **External faults :** Sensor link fault  
Range underrun or overrun fault  
Calibration fault  
Cold junction compensation fault
- **Other faults :** Terminal block fault  
Configuration fault  
Communication fault  
Application fault  
24V power supply fault  
Value outside limits  
Channel not ready

#### Note

Channel diagnostics can also be accessed via the program (READ\_STS instruction).

### 3.5-2 Modifying the filter value

This is performed via a pulldown list box, located in the property dialog box, which enables the user to choose the new filter value on the selected channel : 0 (no filtering), 1 and 2 (low level of filtering), 3 to 4 (medium level of filtering), 5 and 6 (high level of filtering). Once the choice is made and confirmed by pressing **OK**, it appears in the parameter display zone.



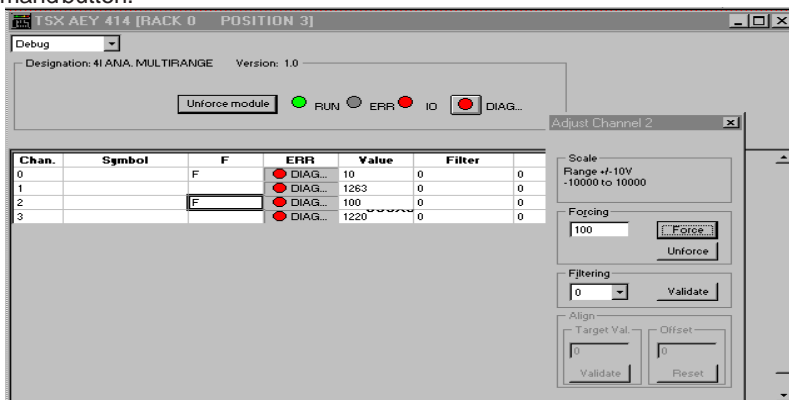
#### Note

Filtering can also be modified via the program (WRITE\_PARAM instruction).

### 3.5-3 Forcing/removing channel forcing

The selected channel is forced via the **Force** command button. The forcing value is defined in the **Forcing** entry field. When a channel is forced, **F** appears in the display zone.

To remove the forcing from a forced channel, select the channel then press the **Unforce** command button.



#### Note

It is only possible to force an output when the task associated with that output is in **RUN**. If the task is in **STOP**, forcing is accepted but not applied : the output is in **Fallback/Maintain**.

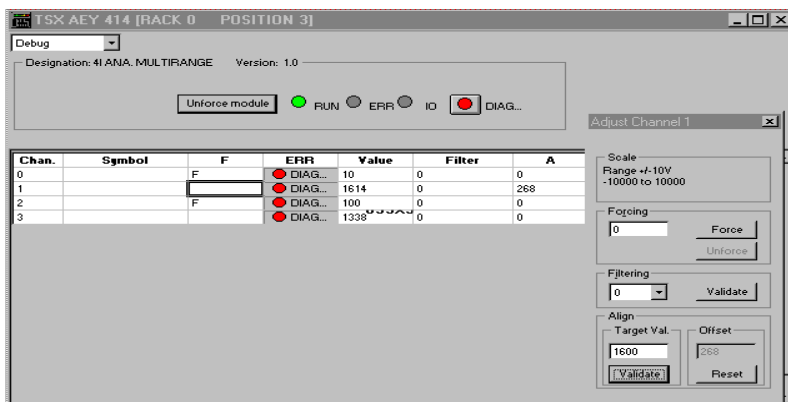
If an output is in forced state, it goes to **Fallback/Maintain** when the associated task goes to **STOP**. When this task goes to **RUN** again, the output takes the **forced** value again.

A forced channel cannot be reconfigured in online mode.

### 3.5-4 Aligning an input channel

The procedure for aligning an input adds an offset value to each value measured by that input in order to compensate for a sensor shift (for example, setting the measurement of a Pt100 probe placed into a bucket of ice for adjustment to 0 °C). To do this, enter the required value in the **Target Value** field, then press **OK**. The offset value, calculated automatically, then appears in the channel parameters display zone.

The **Reset** command button deletes channel alignment.



#### Note

Alignment offset can also be modified via the program (WRITE\_PARAM instruction). The alignment offset should be between +1500 and -1500.

#### Warning

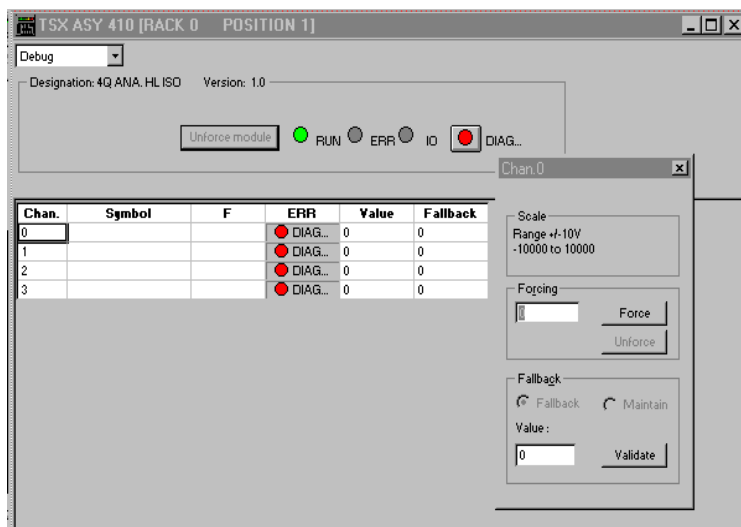
The calculated offset value does not take keyboard commands from the user into account. Simultaneous execution of the alignment adjustment program (RUN) renders the offset incorrect.

### 3.5-5 Modifying output fallback value

When an output is configured to **Fallback**, the corresponding button is selected, but **Fallback/Maintain** data are grayed out, since the fallback mode cannot be modified in Debug.

However, it is possible to modify the fallback value (value adjustment), by entering a new value :

- between -10000..10000 in the  $\pm 10$  V range, and 0..10000 in the 0..20 mA and 4..20 mA ranges for modules TSX ASY 800 and ASY 410 (software version  $\leq 10$ )
  - between -10500..10500 in the  $\pm 10$  V range, and 0..10500 in the 0..20 mA and 4..20 mA ranges for modules TSX ASY 800 and ASY 410 (software version  $> 10$ )
- in the **Value** field, then pressing **OK**. This appears in the channel parameters display zone



#### Note

1. The fallback value can also be modified via the program (WRITE\_PARAM instruction).
2. Fallback/maintain cannot be adjusted on TBX modules.





### 4.1 Introduction to the Calibration function

This function can only be accessed online (**PLC** menu, **Connect** command or click on the corresponding icon). It recalibrates the channels of each analog input module of the application.

For TSX AEY 800 / 810 / 1600, TBX AES 400 and TBX AMS 620 modules, recalibrating channel 0 recalibrates all the module channels.

For TSX AEY 1614 modules, recalibrating channels 0 and 8 recalibrates all the module channels.

For TSX AEY 414 modules, recalibration must be performed channel by channel.

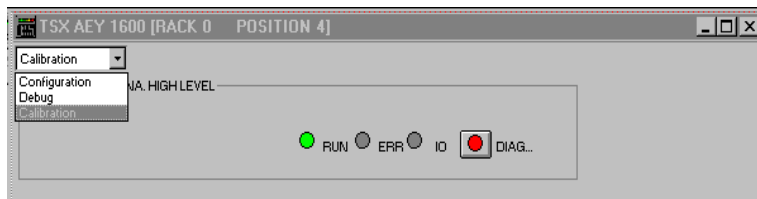
When a channel is recalibrated, the associated PLC task can be in RUN or STOP. In Calibration mode, the measurements of all the module channels are declared invalid (the channel default bit %Ix.i.ERR = 1), filtering and alignment are inhibited and the channel read cycles can be lengthened.

The **Calibration** function of an analog module is accessed by double-clicking on the Station, Configuration and then Hardware configuration icons in the Application Browser and in the module slot in the rack.

In online mode, the **Debug** function is selected by default. The pulldown dialog box in the command zone can be used to access the **Calibration** function.



online mode



**Note** : TBX modules are calibrated on the base connection point. They cannot be recalibrated on the extension.

## 4.2 Description of the calibration screen

This screen displays the selected module and the status of each of its channels (ERR) in realtime, and gives access to their calibration.

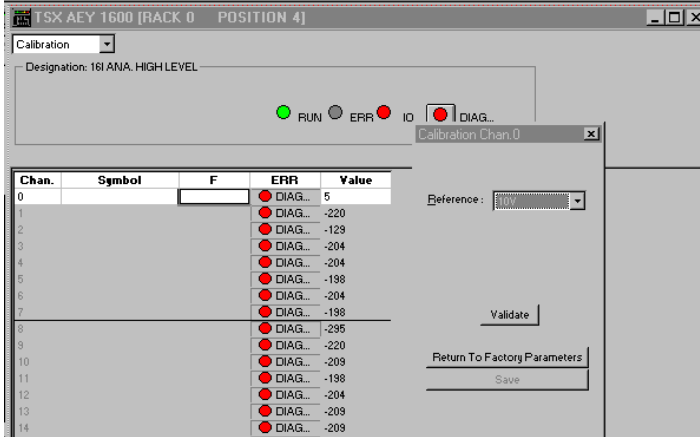
Chan.	Symbol	F	ERR	Value
0			● DIAG...	-301
1			● DIAG...	-220
2			● DIAG...	-129
3			● DIAG...	-204
4			● DIAG...	-204
5			● DIAG...	-198
6			● DIAG...	-204
7			● DIAG...	-198

- 1 This bar shows the catalog reference and the slot of the module in the PLC (rack and position).
- 2 This command zone shows the current function (**Calibration**function) and selects, via a pulldown list box, the**Configuration** or**Debug**function. Activating the**Calibration** check box accesses calibration of the channels (TSX AEY 800 / 810 / 1600, TBX AES 400,TBX AMS 620) or selected channel (TSX AEY 414).
- 3 This "module" level zone contains the module description and version.
- 4 This "channel" level zone displays the **ERR** information for each channel : all measurements are invalid, filtering and alignment are inhibited.

### 4.3 Calibrating TSX AEY 800 / 801 / 1600 / TBX AES 400 / TBX AMS 620 modules

Calibration is performed for the whole module on channel 0. To do this :

- Click on a channel in zone (4) (refer to section 4.2), which displays a warning message. Confirm the change to recalibration mode.



For the TBX AES 400 it is possible to calibrate the voltage 0 by placing a shunt on all the inputs of the module before replying yes to the message displayed by PL7.

No : no 0 calibration.

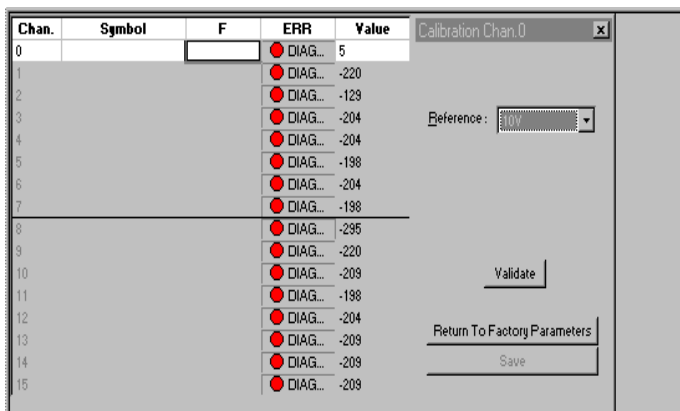
Cancel : no change to calibration mode.

- According to the range to be calibrated, connect a reference voltage to **the voltage input** of channel 0 :
  - reference voltage = 10 V to recalibrate the module for the  $\pm 10$  V and 0..10 V ranges,
  - reference voltage = 5 V to recalibrate the module for the 0.5 V, 1..5 V, 0..20 mA and 4..20 mA ranges.
 The 5V reference enables the entire analog input module to be recalibrated for the 0..20 mA and 4..20 mA ranges, **except for the 250 W current shunt on the current input.**

#### Note

To recalibrate the 0..20 mA and 4..20 mA ranges, connect the 5 V reference to **the voltage input of channel 0.**

- Once the reference has been connected to the voltage input (for example 10 V), use the **Reference** pulldown list box to select this value. Wait for the connected reference voltage to stabilize if necessary, then confirm the choice using the **OK** command button. The ranges linked to this reference (for example  $\pm 10$  V and 0..10 V) are recalibrated automatically.
- Refer to the installation documentation for analog TBX for the calibration reference values (TSX DMTBXV5LE section 1.3-1 for AES 400 modules and section 3.4-1 for AMS 620 modules).
- To calibrate the module for the other ranges if necessary :
  - connect another reference voltage to the voltage input of channel 0 (for example 5 V),
  - use the **Reference** pulldown list box to select this voltage then confirm the new choice with **OK**.
- Use the **Save** command button to take into account and save the new recalibration in the module. When the user exits the **Calibration** screen without saving, a message is displayed to signal that the recalibration operations will be lost.



- The **Return to Factory Parameters** command button is used to cancel all previous recalibrations and to return to the original factory calibration.
- TBX modules are automatically rebooted on exiting calibration mode (for example, when returning to the debug screen).

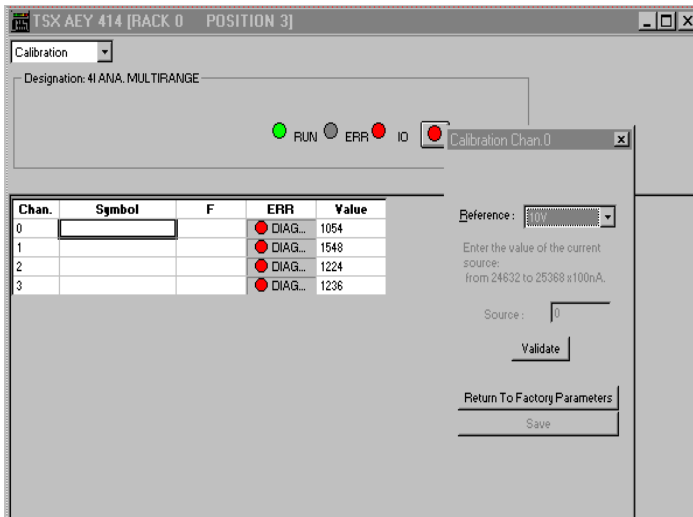
## 4.4 Calibrating the TSX AEY 414 module

Calibration is performed channel by channel. To do this, click on a channel in zone (4) (refer to section 4.2), which displays a warning message. Confirm the change to recalibration mode.

Two types of recalibration are then possible :

- recalibration of the **analog input module** for a channel,
- recalibration of the **current source** necessary for measurements from sensors with resistive probes.

### 4.4-1 Recalibrating the analog input module



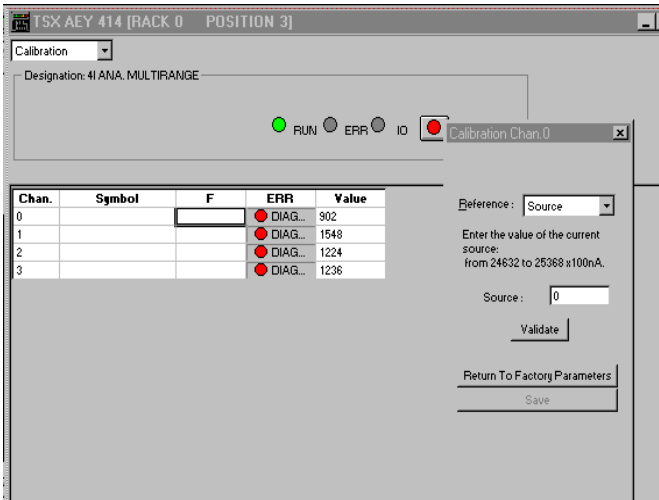
- According to the range to be calibrated, connect a reference voltage to the voltage input of the selected channel :
  - reference voltage = 10 V to recalibrate the channel for the  $\pm 10$  V,  $\pm 5$  V, 0..20 mA and 4..20 mA ranges.
  - reference voltage = 2.5 V to recalibrate the channel for the Pt100, Pt1000 and Ni1000 ranges.
  - reference voltage = 65 mV to recalibrate the channel for the thermocouple ranges.

- 
- Once the reference has been connected to the voltage input (for example 10 V), use the **Reference** pulldown list box to select this value. Wait for the connected reference voltage to stabilize if necessary, then confirm the choice using the **OK** command button. The ranges linked to this reference (for example  $\pm 10$  V,  $\pm 5$  V, 0..20 mA and 4..20 mA) are recalibrated automatically.
  - To calibrate the module for the other ranges :
    - connect another reference voltage to the voltage input of channel 0 (for example 60 mV),
    - use the **Reference** pulldown list box to select this voltage then confirm the new choice with **OK**.
  - Use the **Save** command button to take into account and save the new recalibration in the module. When the user exits the **Calibration** screen without saving, a message is displayed to signal that the recalibration operations will be lost.
  - The **Return to Factory Parameters** command button is used to cancel all previous recalibrations **on the channel concerned** and to return to the original factory calibration (for the channel concerned only).

**Note**

After confirming (OK command), the value  $10000 \pm 2$  points should be displayed for the channel being calibrated; except for 60 mV reference where the value should be  $9523 \pm 2$  points (10000 corresponding to 63 mV).

#### 4.4-2 Recalibrating the current source for a channel



- Using a multimeter, measure the value of the current source supplied by the channel to be recalibrated (this value should be about 2.5 mA).
- Once the current source value has been measured (for example 2.5128 mA), use the **Reference** pulldown list box to select **Source**, then enter this value in the corresponding field. The unit is tenths of  $\mu\text{A}$  (for example 25128 must be entered for 2.5128 mA). Confirm the choice using the **OK** command button.
- Use the **Save** command button to take into account and save the new recalibration in the module. When the user exits the **Calibration** screen without saving, a message is displayed to signal that the recalibration operations will be lost.
- The **Return to Factory Parameters** command button is used to cancel all previous recalibrations **on the channel concerned** and to return to the original factory calibration (for the channel concerned only).

## 4.5 Calibrating TSX AEY 1614 modules

Calibration is performed on channels 0 and 8. To do this, click on a channel in zone (4) (refer to section 4.2), which displays a warning message. Confirm the change to recalibration mode.

Two types of recalibration are then possible for channel 0 :

- recalibration of the **analog input module**,
- recalibration of the **current source** necessary for measurements from sensors with resistive probes.

One single type of recalibration for channel 8 :

- recalibration of the **analog input module**.

### 4.5-1 Recalibrating the analog input module

The screenshot shows the 'Calibration' window for a TSX AEY 1614 module. The window title is 'TSX AEY 1614 [RACK 0 POSITION 6]'. Below the title bar, there is a 'Calibration' dropdown menu and a 'Designation: 161 ANA. THERMO.' label. A status bar contains four indicators: a green circle for 'RUN', a grey circle for 'ERR', a red circle for 'ID', and a red circle for 'DIAG...'. A 'Calibration Chan.0' dialog box is open, displaying a table of channels and their calibration status.

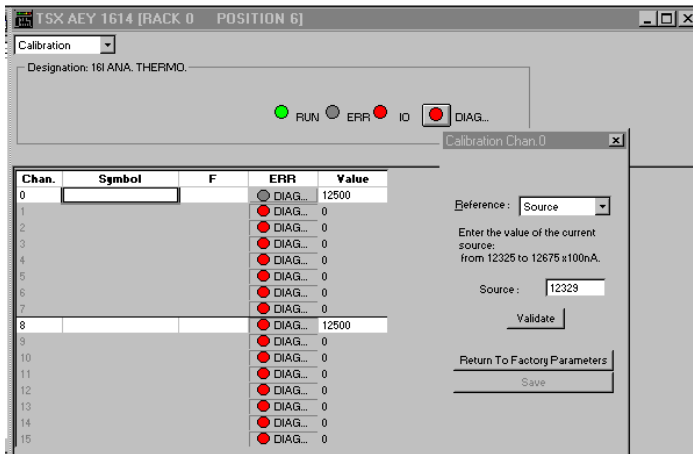
Chan.	Symbol	F	ERR	Value
0			● DIAG...	-3775
1			● DIAG...	0
2			● DIAG...	0
3			● DIAG...	0
4			● DIAG...	0
5			● DIAG...	0
6			● DIAG...	0
7			● DIAG...	0
8			● DIAG...	-3812
9			● DIAG...	0
10			● DIAG...	0
11			● DIAG...	0
12			● DIAG...	0
13			● DIAG...	0
14			● DIAG...	0
15			● DIAG...	0

The dialog box also includes a 'Reference:' dropdown menu, a 'Validate' button, a 'Return To Factory Parameters' button, and a 'Save' button.



- 
- Connect one of the following reference voltages to the voltage input of the selected channel to recalibrate for thermocouple ranges :
    - common reference voltages for channels 0 and 8 : equal to 25 mV, 55 mV or 80 mV
    - or reference voltage for channel 0 : equal to 1.6 V
    - or common reference voltage for channel 8 : equal to 166.962 mV
  - Once the reference has been connected to the voltage input (for example 25 V), use the **Reference** pulldown list box to select this value. Wait for the connected reference voltage to stabilize if necessary, then confirm the choice using the **OK** command button. The ranges linked to this reference are recalibrated automatically.
  - To calibrate the module for the other ranges :
    - connect another reference voltage to the voltage input of channel 0 (for example 55 mV),
    - use the **Reference** pulldown list box to select this voltage then confirm the new choice with **OK**.
  - Use the **Save** command button to take into account and save the new recalibration in the module. When the user exits the **Calibration** screen without saving, a message is displayed to signal that the recalibration operations will be lost.
  - The **Return to Factory Parameters** command button is used to cancel all previous recalibrations **on the channel concerned** and to return to the original factory calibration (for the channel concerned only).
-

## 4.5-2 Recalibrating the current source for a channel



- Use the **Reference** pulldown list box to select **Source**,
- When the information message "You are going to calibrate the current channel. Do you wish to continue?" is displayed, answer Yes.
- Using a multimeter, measure the value of the current source supplied by the channel to be recalibrated (this value should be about 1.25 mA).
- Once the current source value has been measured (for example 1.2329 mA), then enter this value in the corresponding field. The unit is tenths of  $\mu\text{A}$  (for example 12329 must be entered for 1.2329 mA). Confirm the choice using the **OK** command button.
- Use the **Save** command button to take into account and save the new recalibration in the module. When the user exits the **Calibration** screen without saving, a message is displayed to signal that the recalibration operations will be lost.
- The **Return to Factory Parameters** command button is used to cancel all previous recalibrations **on the channel concerned** and to return to the original factory calibration (for the channel concerned only).

### 5 Bits and words associated with the analog function

#### 5.1 Addressing in-rack analog module objects

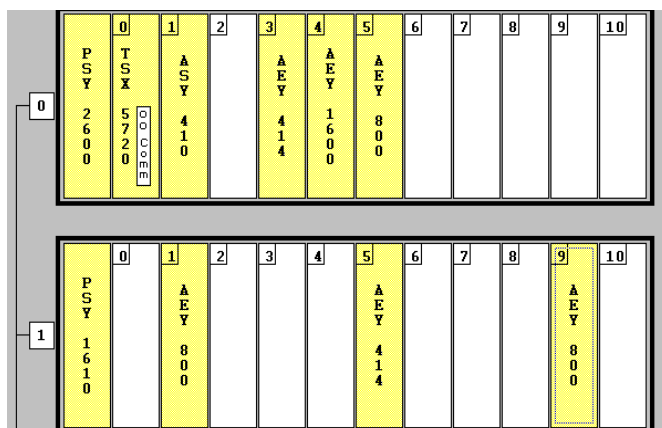
Principle for addressing the image bits of the I/O of TSX Premium PLC in-rack analog modules.

Channel addressing is geographical, and therefore depends on ;

- the rack number (address),
- the physical position of the module in the rack,
- the module channel number.

#### Positions of modules y : 00 to 10

Addresses  
of racks  
x : 0 to 7



Syntax of analog I/O addresses :

%	I W, QW, ...	address rack x	position module y	.	Chann.no. i
Symbol	<b>Type of object</b> IW = input word QW = output word I = input bit Q = output bit MW = internal word	x = 0 to 7	y = 00 to 10		i = 0 to 63

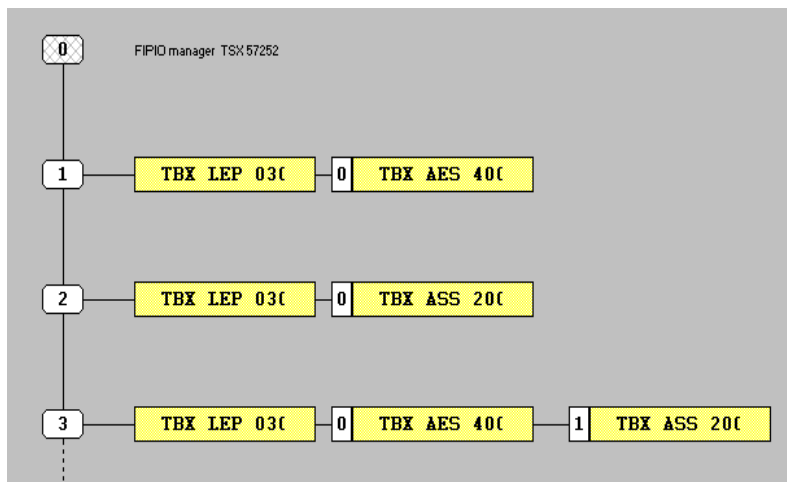
#### Example:

%IW101.5 means : image word of analog input 5 of the module in position 1 in rack 1.  
%QW10.3 designates the image word of analog output 3 of the module in position 10 in rack 0.

## 5.2 Addressing distributed analog module objects

Principle for addressing the image bits of the I/O of distributed analog I/O modules. Channel addressing is geographical, and therefore depends on ;

- the connection point,
- the module type : base or extension,
- the channel number.



Syntax of remote analog I/O addresses :

%	I,Q,M,K	X,W or D	p.2.c	m	i	r
Symbol	Type of object	Format	Module/channel and connection point address	Module number	Channel number	Rank
	I = input Q = output M = internal variable K = internal constant	X = input W = word D = double word F = floating point	p= processor address 0 or 1 2= No. of integrated FIPIO link channel c= No. of connection point from 1 to 255	0= base 1= extension MOD	0 to 127 or ERR	0 to 255 or

### Example :

%IW\0.2.6\0.5 means : image word of analog input 5 of the distributed input base module located at connection point 6 of the FIPIO bus.

%QW\0.2.8\1.7 means : image word of analog output 7 of the distributed output extension module located at connection point 8 of the FIPIO bus.

### 5.3 Language objects associated with the analog I/O

The configuration of an analog input or output module, in a given slot (rack, position), automatically generates a set of language objects necessary for the programming and diagnostics of this module.

There are several types of language object associated with the analog function :

- **Implicit exchange objects** which are exchanged automatically with each scan of the task in which the module channels are configured.
- **Explicit exchange objects** which are exchanged by instruction. These objects are only useful for advanced programming of the application-specific function .

**Note** : in the rest of this document the addressing referred to is that of in-rack modules, for distributed analog modules, simply replace this addressing with that specified in section 5.2.

#### 5.3-1 Implicit exchange objects associated with inputs

Bits :

- **%Ixy. MOD.ERR** : module fault bit. At state 1, this bit signals a fault in the module located in position y of the rack with address x,
- **%Ixy.i.ERR** : channel fault bit. At state 1, this bit signals a fault in channel i of the module located in position y of the rack with address x.

Words :

- **%IWxy.i** : input channel i of the module located in position y of the rack with address x,
- **%QWxy.i** : output channel i of the module located in position y of the rack with address x.

For example, the word %IW105.3 contains the value present at input 3 of the module located in position 5 of the rack with address 1.

#### Summary table

Type of object	Address	Input modules
Module fault bit	%Ixy.MOD.ERR	All
Channel fault bit	%Ixy.i.ERR	All
Measured analog value word	%IWxy.i	All
Measurement status word	%IWxy.i.1	TSX AEY 420/810/1614
Event source status word	%IWxy.i.2	TSX AEY 420
Enable event command word	%QWxy.i	TSX AEY 420

### Details of measurement status word %IWxy.i

Address	Bit rank	Meaning
%IWxy.i.1:X0	0	Aligned channel
%IWxy.i.1:X1	1	Forced channel
%IWxy.i.1:X2	2	Recalibration mode
%IWxy.i.1:X3	3	Recalibration command in progress
%IWxy.i.1:X4	4	Channel recalibrated
%IWxy.i.1:X5	5	Measurement within lower tolerance zone
%IWxy.i.1:X6	6	Measurement within upper tolerance zone
%IWxy.i.1:X7	7	Loss of event (TSX AEY 420)
%IWxy.i.1:X8 to 15	8 to 15	Reserved

### Details of event source status word %IWxy.i.2 :

indicates the source of the event (0=no event, 1=event)

Address	Bit rank	Meaning
%IWxy.i.2:X0	0	Crossing of threshold 0, + direction
%IWxy.i.2:X1	1	Crossing of threshold 0, - direction
%IWxy.i.2:X2	2	Crossing of threshold 1, + direction
%IWxy.i.2:X3	3	Crossing of threshold 1, - direction
%IWxy.i.2:X4 to 15	4 to 15	Reserved

### Details of event enable command %QWxy.1 :

used to enable or mask events (0=masking, 1=enable)

%QWxy.i.1:X0	0	Crossing of threshold 0, + direction
%QWxy.i.1:X1	1	Crossing of threshold 0, - direction
%QWxy.i.1:X2	2	Crossing of threshold 1, + direction
%QWxy.i.1:X3	3	Crossing of threshold 1, - direction
%QWxy.i.1:X4 to 15	4 to 15	Reserved

### 5.3-2 Explicit exchange objects associated with inputs

Type of word	Address	Input modules
Modulestatus	%MWxy.MOD.2	All
Exchangeinprogress	%MWxy.i	All
Exchangerreport	%MWxy.i.1	All
Channelstatus	%MWxy.i.2	All
Command(recalibration/forcing)	%MWxy.i.3	AEY 420/810/1614 TBX AES 400/AMS 620/ASS 200
Command (forcing value)	%MWxy.i.4	AEY 420/810/1614 TBX AES 400/AMS 620/ASS 200
Command (range to recalibrate)	%MWxy.i.5	AEY 810/1614,TBX AES 400/AMS 620
Command (current source to recalibrate)	%MWxy.i.6	AEY 414
Adjustment (filtering coefficient)	%MWxy.i.7	AEY 414/800/810/1600/1614 TBX AES 400/AMS 620
Adjustment (alignment offset)	%MWxy.i.8	AEY 414/420/800/810/1600/1614
Threshold 0	%MWxy.i.9	AEY 420
Threshold 1	%MWxy.i.10	AEY 420

Words %MWxy.MOD, %MWxy.MOD.1, %MWxy.MOD.3 are not used.

For TBX, words i.0, i.1, i.5 and i.6 are only used for channels 0 and 4 of the AMS 620 module. The data in these words concerns the 2 or 4 consecutive module channels.

**Details of word %MWxy.MOD.2** : module status word, only updated on request.

Address	Bit rank	Meaning
%MWxy.MOD.2:X0	0	Module failure
%MWxy.MOD.2:X1	1	Channel(s) fault
%MWxy.MOD.2:X2	2	Terminal block fault
%MWxy.MOD.2:X3	3	Self-test running
%MWxy.MOD.2:X4	4	Reserved
%MWxy.MOD.2:X5	5	Configuration fault
%MWxy.MOD.2:X6	6	Module missing or switched off
%MWxy.MOD.2:X7 to 15	7 to 15	Extension module reserved on FIP

**Details of word %MWxy.i.2** : channel i status word, only updated on request.

Address	Bit rank	Meaning
%MWxy.i.2:X0 (1)	0	Sensor link fault
%MWxy.i.2:X1	1	Range under/overrun fault
%MWxy.i.2:X2	2	Terminal block fault
%MWxy.i.2:X3	3	Not used
%MWxy.i.2:X4	4	Module failure
%MWxy.i.2:X5	5	Configuration fault
%MWxy.i.2:X6	6	Communication fault
%MWxy.i.2:X7	7	Value outside limits
%MWxy.i.2:X8	8	Channel not ready
%MWxy.i.2:X9	9	Action rejected
%MWxy.i.2:X10	10	Calibration fault
%MWxy.i.2:X11	11	Recalibration in progress (2)
%MWxy.i.2:X12	12	Recalibration mode (2)
%MWxy.i.2:X13	13	Forced channel (2)
%MWxy.i.2:X14	14	Recalibrated channel (2) Range underrun (3)
%MWxy.i.2:X15	15	Aligned channel (2) (4) Range overrun (3)

(1) for Momentum I/O : temporary fault external to the sub-base, the meaning depends on the sub-base selected (see Momentum documentation).

(2) for TSX AEY 1600/800/414

(3) for TSX AEY 810/420/1614

(4) except analog TBX

- **%MWxy.i.7** : command word containing the channel filtering coefficient.
- **%MWxy.i.8** : command word containing the channel alignment offset.
- **%MWxy.i.9** : command word containing the value of threshold 0 assigned to the channel.
- **%MWxy.i.10** : command word containing the value of threshold 1 assigned to the channel.



### 5.3-3 Implicit exchange objects associated with outputs

#### Summary table

Type of object	Address	Output modules
Module fault bit	%Ixy.MOD.ERR	All
Channel fault bit	%Ixy.i.ERR	All
Command word containing the values of the analog outputs	%QWxy.i	All

### 5.3-4 Explicit exchange objects associated with outputs

#### Summary table

Type of object	Address	Output modules
Module status	%MWxy.MOD.2	TSX ASY 410/800
Exchange in progress	%MWxy.i	TSX ASY 410/800
Exchange report	%MWxy.i.1	TSX ASY 410/800
Channel status	%MWxy.i.2	TSX ASY 410/800
Adjustment (fallback value)	%MWxy.i.5	TSX ASY 410/800

Words %MWxy.MOD, %MWxy.MOD.1, %MWxy.MOD.3 are not used.  
For TBX AMS 620 and ASS 200 analog output modules, see section 5.3-2

#### Details of word %MWxy.MOD.2 : module status word,

Address	Bit rank	Meaning
%MWxy.MOD.2:X0	0	Module failure
%MWxy.MOD.2:X1	1	Channel(s) fault
%MWxy.MOD.2:X2	2	Terminal block fault
%MWxy.MOD.2:X3	3	Self-test running
%MWxy.MOD.2:X4	4	Reserved
%MWxy.MOD.2:X5	5	Configuration fault
%MWxy.MOD.2:X6	6	Module missing or switched off
%MWxy.MOD.2:X7 to 15	7 to 15	Reserved

---

**Details of word %MWxy.i.2 : channel i status word**

Address	Bit rank	Meaning
%MWxy.i.2:X0	0	24 V supply fault (TSX ASY 800)
%MWxy.i.2:X1	1	Range under/overrun fault
%MWxy.i.2:X2	2	Terminal block fault
%MWxy.i.2:X3	3	Range overrun fault if bit %MWxy.i.2:X1 is at 1 (for TSX ASY 800 and TSX ASY 410(II>10))
%MWxy.i.2:X4	4	Module failure
%MWxy.i.2:X5	5	Configuration fault
%MWxy.i.2:X6	6	Communication fault
%MWxy.i.2:X7	7	Value outside limits
%MWxy.i.2:X8	8	Channel not ready
%MWxy.i.2:X9	9	Action rejected
%MWxy.i.2:X10 to 12	10 to 12	Reserved
%MWxy.i.2:X13	13	Forced channel
%MWxy.i.2:X14 to 15	14 to 15	Reserved

**%MWxy.i.3** : word reserved

**%MWxy.i.4** : command word containing the channel forcing value.

**%MWxy.i.5** : command word containing the channel fallback value.

## 5.4 %CH language objects

The %CH language object is used to simplify explicit reading and writing. It :

- reads the module and channel status words,
- writes the adjustment parameters associated with channels,
- saves the adjustment parameters,
- restores the adjustment parameters.

### Reading the status word

The following syntax is used to read the module status word :

**READ\_STS %CHxy.MOD**      x = rack number, y = position in the rack

For example, READ\_STS%CH103.MOD updates the contents of word %MW103.MOD.2

The following syntax is used to read the channel status words :

**READ\_STS %CHxy.i**      x = rack number, y = position in the rack  
i = channel number

For example, READ\_STS%CH3.0 updates the contents of word %MW3.0.2.

### Writing adjustment parameters

The following syntax is used to write the channel adjustment parameters (alignment, filtering of inputs and fallback of outputs) :

**WRITE\_PARAM %CHxy.i**      x = rack number, y = position in the rack  
i = channel number

### Saving adjustment parameters

The following syntax is used to save the channel adjustment parameters :

**SAVE\_PARAM %CHxy.i**      x = rack number, y = position in the rack  
i = channel number

### Restoring adjustment parameters

The following syntax is used to restore the channel adjustment parameters :

**RESTORE\_PARAM %CHxy.i**      x = rack number, y = position in the rack  
i = channel number

**Note** : for remote devices, see the Communication manual, part H, section 2.4.



**Symboles**

%IWxy.2	5/4
%IWxy.i	5/3
%Ixy.MOD.ERR	5/3
%Ixy.i.ERR	5/3
%MWxy.i.2	5/8
%MWxy.MOD.2	5/5
%QWxy.1	5/4
%QWxy.i	5/3

**A**

Addressing	5/1, 5/2
Adjustment	3/4
Alignment	3/8
Analog channels	1/3
Application Browser	1/2

**C**

Calibration	4/1
Channels used	2/11
Choosing modules	1/3
Cold junction compensation	2/14
Configuration editor	1/2
Confirmation	2/20
Copy/paste	2/3
Current shunt	4/3

**D**

Debug	3/1
Diagnostics	3/5
Display	2/2
Display format	2/9

**E**

Event enable command	5/4
Event processing	2/13
Event source status word	5/4
Explicit exchange objects	5/5, 5/7
External faults	3/5

**F**

Fallback	3/9
Fallback mode	2/17
Family	1/3

Fast cycle	2/10
FAST task	2/8, 2/16
Filtering	2/10, 3/6
FIPIO	5/2
FIPIO configuration	1/6
FIPIO connection point	1/6
Forcing	3/4, 3/7

**H**

High precision	2/15
----------------	------

**I**

Implicit exchange objects	5/3, 5/7
Input range	2/8
Internal faults	3/3, 3/5

**M**

Maintain	3/9
MAST task	2/8, 2/16
Maximum number of channels	1/3
Measurement status word	5/4
Module	1/3
Module diagnostics	3/3
Module faults	3/3
Module reference	1/3
MOMENTUM	1/7, 2/7, 2/19
Momentum parameters	2/7

**N**

Normal cycle	2/10
--------------	------

**O**

OTHER	1/7, 1/9
Other faults	3/3, 3/5
Output power supply	2/18
Output range	2/16

**P**

Parameter settings	1/4, 1/8
Parameters	2/4, 3/1
Power supply fault	2/18

---

## R

Range under/overrun	2/17
READ_STS	5/9
Reconfiguration	2/20
RESTORE_PARAM	5/9
Return to factory parameters	4/4

## S

SAVE_PARAM	5/9
Scale	2/9
Scan cycle	2/10
Shortcut menus	2/3
Standardized	2/9
Status word	5/4

## T

Task	2/8
TBX	1/7, 2/6
TBX parameters	2/6
Temperature	2/9
Temperature probe range	2/9
Terminal block	2/11, 2/18
Terminal block detection	2/11, 2/18
Thermocouple range	2/9
Threshold	2/13
TSX AEY parameters	2/4
TSX ASY parameters	2/5
Type of module	1/3

## U

Under/overrun	2/12
User	2/9

## W

Wiring check	2/12
WRITE_PARAM	5/9

<b>Section</b>	<b>Page</b>
<b>1 Introduction</b>	<b>1/1</b>
1.1 General	1/1
1.2 Constituent elements	1/2
1.3 Control loop principles	1/3
1.4 Methodology	1/4
<b>2 Description of process control functions</b>	<b>2/1</b>
2.1 General points on process control functions	2/1
2.1-1 Accessing the functions	2/1
2.1-2 Operating modes	2/1
2.1-3 Programming	2/3
2.2 The PID function	2/4
2.2-1 Functions	2/4
2.2-2 General schematic	2/5
2.2-3 PID programming	2/8
2.3 The PWM function	2/11
2.3-1 Functions	2/11
2.3-2 Description	2/11
2.3-3 PWM programming	2/13
2.4 The SERVO function	2/15
2.4-1 Functions	2/15
2.4-2 Description	2/15
2.4-3 SERVO programming	2/18
2.5 Behavior of functions in various PLC operating modes	2/20
2.5-1 Cold restart	2/20
2.5-2 Warm restart	2/20
2.5-3 Addition of a new function call in online mode	2/20

Section	Page
<b>3 Man-machine interface on CCX 17</b>	<b>3/1</b>
3.1 Man-machine interface on CCX 17	3/1
3.2 Description of predefined screens	3/2
3.2-1 Selecting a loop	3/2
3.2-2 Controlling a loop	3/3
3.2-3 Adjusting a loop	3/4
3.3 The PID_MMI function	3/5
3.3-1 PID_MMI functions	3/5
3.3-2 Description of the PID_MMI	3/5
3.3-3 PID_MMI programming	3/6
3.3-4 Behavior of the PID_MMI function depending on the operating modes of the PLC and CCX 17	3/9
<b>4 Using a process control application</b>	<b>4/1</b>
4.1 Adjusting loops/debugging the application	4/1
4.1-1 With CCX 17	4/1
4.1-2 Without CCX 17	4/2
4.2 Diagnostics / Maintenance	4/3
4.3 Additional programming	4/4
4.3-1 Adding a bias to the output	4/4
4.3-2 PIDs in cascade	4/5



<b>Section</b>	<b>Page</b>
<b>5 Application example</b>	<b>5/1</b>
5.1 Temperature control	5/1
5.1-1 Application description	5/1
5.1-2 Hardware configuration	5/3
5.1-3 Simplified schematic of the control loop	5/4
5.1-4 Programming	5/4
<b>6 Appendix</b>	<b>6/1</b>
6.1 Reminder of process control	6/1
6.1-1 PID parameter adjustment	6/1
6.1-2 PID parameter influence and effects	6/3
6.2 Function characteristics	6/7
6.2-1 Memory usage	6/7
6.2-2 Function execution time	6/7
<b>7 Index</b>	<b>7/1</b>

---



## 1.1 General

Process control functions are **basic elements** of PL7 Micro language, used for programming control loops on PLCs.

These functions are particularly suitable for :

- meeting the needs of sequential processes which require auxiliary process control functions (examples : shrink-wrapping machines, surface treatment machines, presses, etc),
- meeting the needs of simple process control processes (examples : metal smelting furnaces, ceramic kilns, small cooling assemblies, etc),
- satisfying the particular demands of servo control or mechanical control where the sampling period is crucial (examples : torque control, speed control).

A preconfigured interface for use with the CCX 17 range enables the control and adjustment of process control loops. Hence up to 9 process control loops can be accessed by the CCX 17.

### Comment :

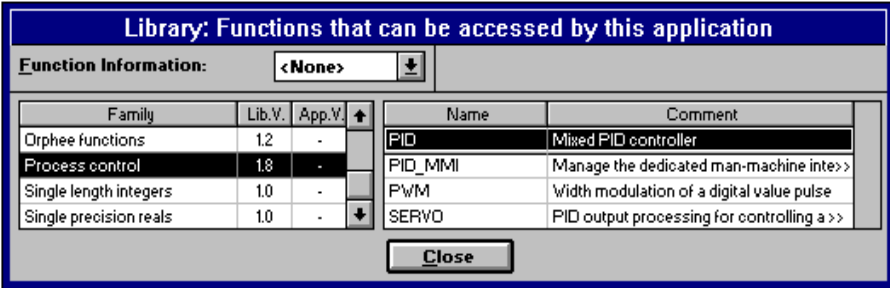
There is no limit to the number of PID control functions in an application.  
In practice, the number of control loops is limited by the maximum number of input and output modules accepted by the PLC.

### Important

The man-machine interface (PID\_MMI) operates with version 2 of the CCX 17 operator panels.

## 1.2 Constituent elements

The software elements comprising the functions required for executing process control applications are present both in the form of a family of functions and in the form of predefined screens for CCX 17 operator panels. The family of functions consists of 3 algorithmic functions and one man-machine interface function.



The basic process control functions are :

- **PID** : for performing mixed PID-type control (serial - parallel),
- **PWM** : for adapting modulation periods on discrete outputs,
- **SERVO** : for motor control adaptations.

The **PID-MMI** function integrates an application program for controlling and adjusting the application PID algorithms on a CCX 17. This function is associated with 3 types of preconfigured screen.

The types of preconfigured screen are as follows :

- an initial screen listing the installed control loops (a maximum of 9) and used to select the required loop,
- a second screen for controlling the selected loop,
- a final screen used to adjust and modify the parameters of the selected loop controller.

### 1.3 Control loop principles

The operation of a PID control loop comprises three distinct phases :

- acquisition of one or more process values (from the process sensors) and setpoints (generally originating from internal PLC variables or from CCX 17 data),
- execution of the PID control algorithm,
- sending of commands adapted to the characteristics of the actuators to be controlled via discrete or analog outputs.

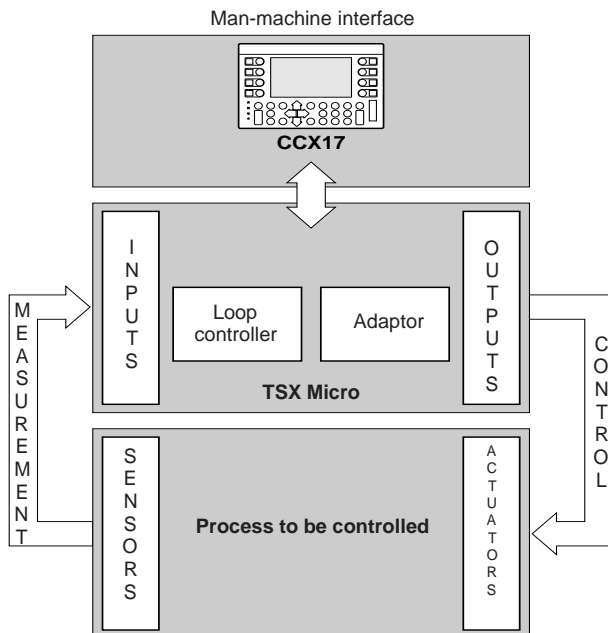
TSX Micro and TSX Premium PLCs have the following interface modules :

- analog : for measurements of current (4-20 mA, etc), voltage ( $\pm 10V$ , etc), PT100, NI1000, thermocouples, etc.
- counter : for measurements from pulse or incremental encoders.

The PID algorithm works out the control signal on the basis of :

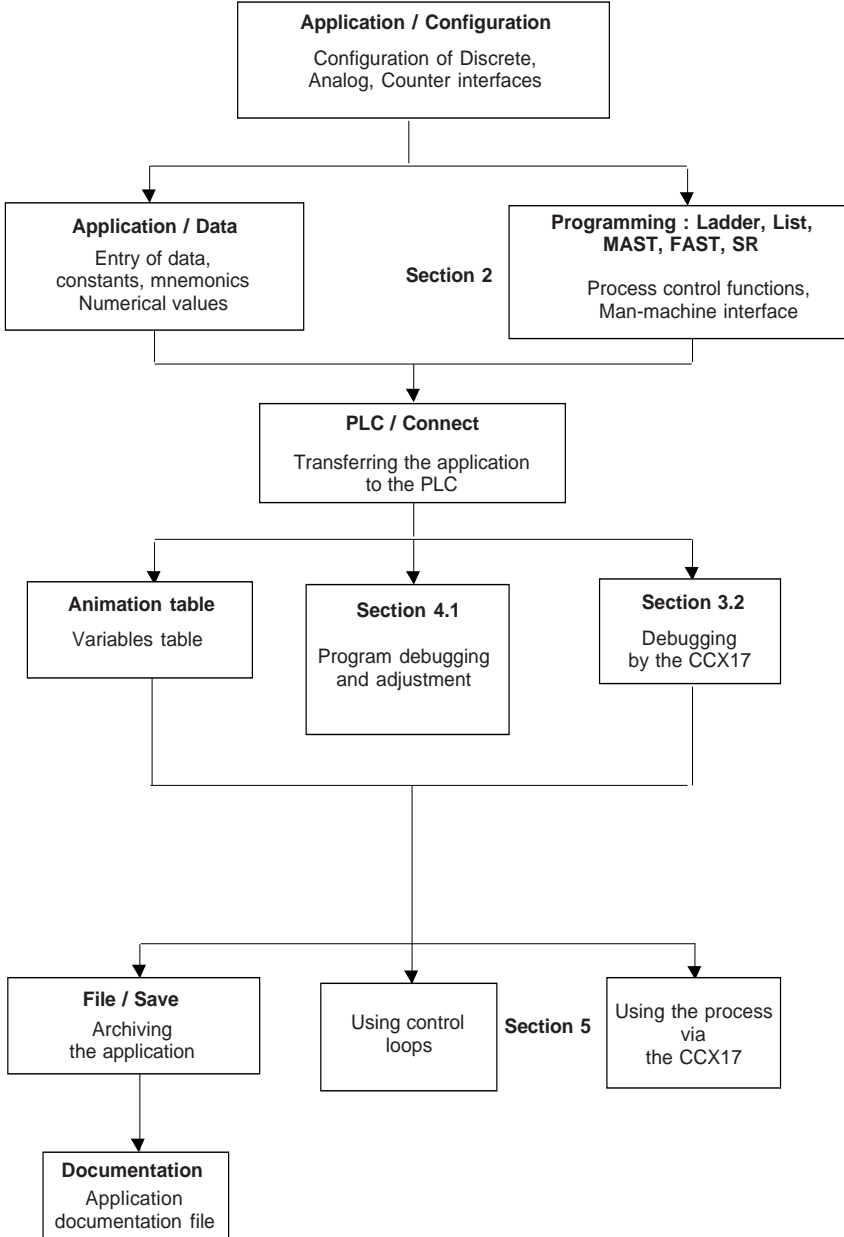
- the measurement sampled by the input module,
- the setpoint value fixed either by the operator or by the program,
- the values of the various loop controller parameters.

The loop controller signal is processed either directly by a PLC analog output module connected to the actuator, or via the PWM or SERVO adaptations depending on the types of actuator to be controlled by a PLC discrete output module.




## 1.4 Methodology

The diagram below describes the sequence of tasks to be performed during the creation and debugging of a process control application.



## 2.1 General points on process control functions


### 2.1-1 Accessing the functions

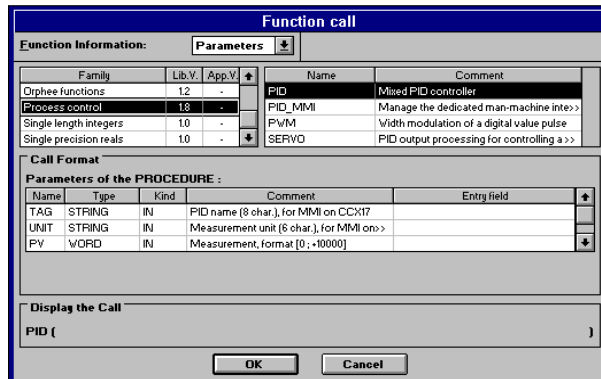
The functions can be accessed under the Function Call menu during programming, by pressing SHIFT-F8 or directly via the icon .

Process control functions are accessible under the "Process control" family.

### 2.1-2 Operating modes

In LADDER language :

Press the SHIFT and F8 keys simultaneously or select the icon  and place it in the rung. The Function Call window is then displayed.



1. Select the required family.

Family	Lib.V.	App.V.	↑
Orphee functions	1.2	-	
<b>Process control</b>	<b>1.8</b>	<b>-</b>	
Single length integers	1.0	-	
Single precision reals	1.0	-	↓

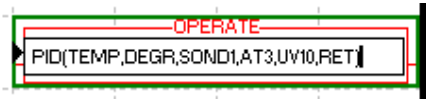
2. Select the function.

Name	Comment
PID	Mixed PID controller
PID_MMI	Manage the dedicated man-machine inte>>
PWM	'width modulation of a digital value pulse
SERVO	PID output processing for controlling a >>

3. Enter the function parameters in the entry field. When you have entered all the parameters, confirm them with OK or by pressing RETURN on the keyboard.

Name	Type	Kind	Comment	Entry field	
TAG	STRING	IN	PID name (8 char.), for MMI on CCX17	TEMP	↑
UNIT	STRING	IN	Measurement unit (8 char.), for MMI on>>	DEGR	
PV	WORD	IN	Measurement, format [0; +10000]	SOND1	↓

4. The function appears. Confirm using RETURN for it to be taken into account by the program.



**In LIST language :**

When entering data in IL language, load the accumulator to 1 to perform an unconditioned function call, then open the function call bracket.



To obtain a list of the functions, either select the 'Enter the Call for a Function' command from the 'Utilities' menu, or press SHIFT and F8 simultaneously. The rest of the procedure is the same as for Ladder.

```
ld true
[PID('TEMPERAT', 'DEGREES', probe1, control, auto_man, para_pid:43)]
```

Once the function has been entered, close the bracket and confirm using SHIFT RETURN.

**In STRUCTURED TEXT language :**

When entering data in Structured Text language, directly enter the unconditioned call to the function :

```
Eg:PID('TEMPERAT', 'DEGREES', probe1, control, auto_man, para_pid:43);
```

To obtain a list of the functions, either select the 'Enter the Call for a Function' command from the 'Utilities' menu, or press SHIFT and F8 simultaneously. The rest of the procedure is the same as for Ladder.

Once the function has been entered, close the bracket and confirm using SHIFT RETURN.

**Note :** For more information on these functions, see part C.



### 2.1-3 Programming

#### Important :

Process control functions must be programmed in a **periodictask** (MAST or FAST). They **do not need to have a condition input**.

All process control function parameters must be entered.

The functions use three types of parameter :

- read-only parameters, taken into account at the beginning of execution of the function,
- write-only parameters, set at the end of execution of the function,
- read and write parameters, the contents of which are taken into account at the beginning of execution of the function and are subsequently updated by the function results.

Word-type input parameters are analog values expressed using the scale [0, +10000] and can be directly connected to the measurement sensors via the %IWxxx analog inputs.

Bit-type output parameters control discrete actuators and can be connected directly to %Qx.y variables.

In the same way, word-type output parameters control analog actuators on the scale [0, +10000] and can be assigned directly to %QWxx variables.

%MWxx:yy word table type parameters include user parameters and the data necessary for internal operation of the function. The function is not executed if a table is not sufficiently long.

#### Important :

In order to maintain the process control function block adjustment parameters on a cold restart, it is necessary to delete the option for resetting %Wi words (in the processor configuration screen).

#### Comment :

Since these are read and write parameters, a table of constants (%KW xx : yy) cannot be used.

Character string type parameters, used by the man-machine interface function, will be entered between 'quotation marks'.

---

## 2.2 The PID function

---

### 2.2-1 Functions

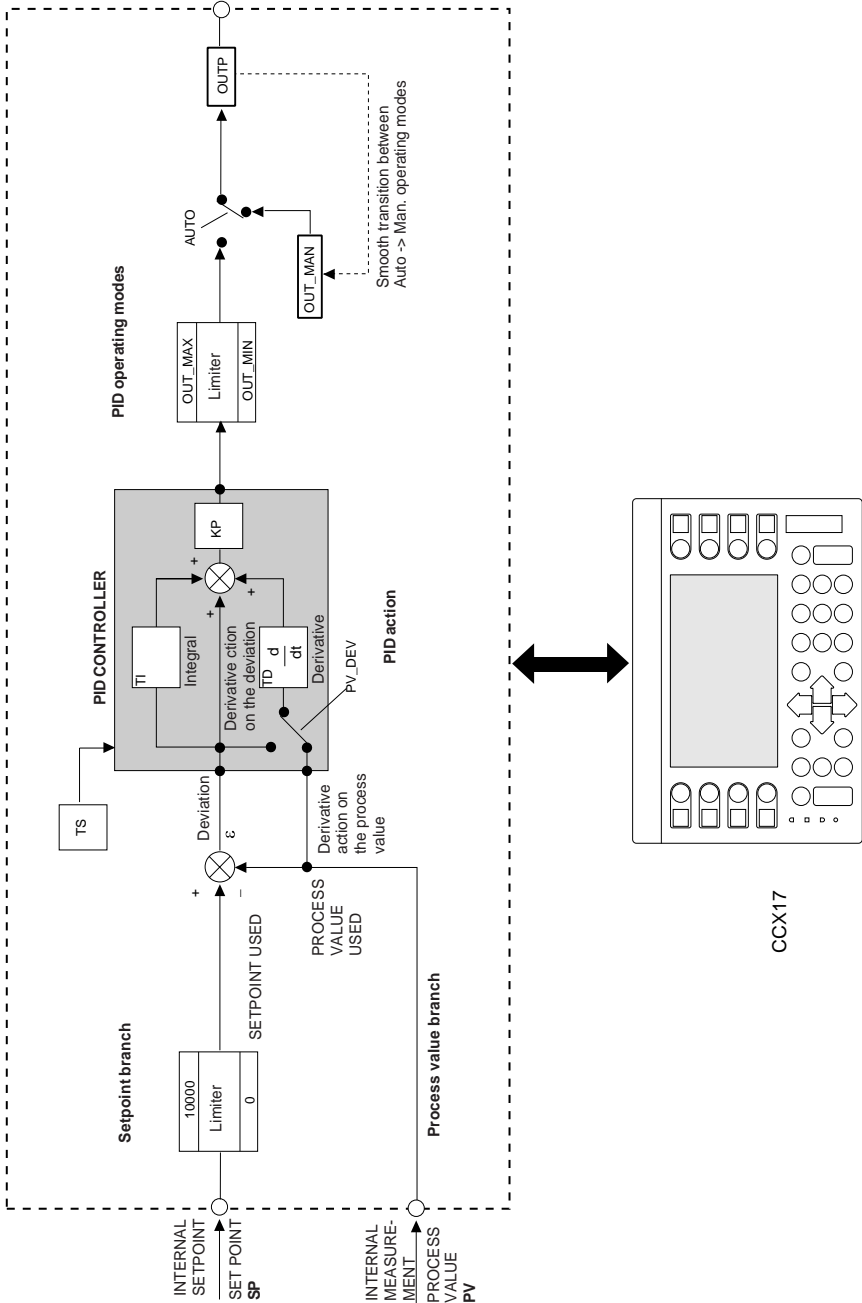
The PID function performs PID correction on the basis of a process value and an analog setpoint with the format [0 - 10000] and issues an analog command with the format [0 - 10000].

The PID control optional function block comprises the following functions :

- serial - parallel PID algorithm,
- direct / inverse action (depending on the sign of the gain KP),
- derivative action on the process value or the deviation,
- upper and lower limit of the setpoint at 0 - 10000,
- upper and lower limit of the output in automatic mode,
- anti-saturation of the integral action,
- smooth transition between Manual and Automatic operating modes,
- control of PID access via the man-machine interface,
- operation as a pure integrator ( $KP = TD = 0$ ).

The display parameters used by the CCX 17 are expressed as physical units.

2.2-2 General schematic



The table below describes the user parameters of the PID function. How the data is structured is described in the programming section.

The default value of the parameters is the value taken when the function is first executed after a cold restart, if all the parameters are at 0 (ie. no prior initialization has been performed, either by the terminal or the program).

Parameter	Type	Nature	Default value	Description
TAG	Characters(8)	Input	-	Name of the PID used by the CCX
UNIT	Characters(6)	Input	-	Unit of measurement used by the CCX
PV	Word	Input	-	Process value in the format 0/10000
OUT	Word	Output	0	PID analog output
AUTO	Bit	I/O	0	PID operating mode 0 : manual, 1 : automatic
SP	Word	I/O	0	Internal setpoint in the format 0/10000
OUT_MAN	Word	I/O	0	Value of the PID manual output (0; 10000)
KP	Word	I/O	100	Proportional gain of the PID algorithm (x100), signed and without a unit. The sign of KP determines the direction of PID action (<0: direct direction, >0: inverse direction) (-10000 - KP - +10000).
TI	Word	I/O	0	PID integral time (between 0 and 20000), (in 1/10 of a second).
TD	Word	I/O	0	PID derivative time (between 0 and 10000), (in 1/10 of a second).
TS	Word	I/O	Period for the task in which the PID is located	PID sampling period (in 1/100 of a second) between 10 ms and 5 min 20 s. The actual sampling period will be a multiple of the task period in which the PID is located, as close to TS as possible
OUT_MAX	Word	I/O	10000	Upper limit of the PID output in automatic mode (between 0 and 10000)

Parameter	Type	Nature	Default value	Description
OUT_MIN	Word	I/O	0	Lower limit of the PID output in automatic mode (between 0 and 10000)
PV_DEV	Word bit	I/O	0	Selection of derivative action on process value (0) or deviation (1)
DEVAL_MMI	Word bit	I/O	0	At 1 to inhibit the man-machine interface from taking this PID into account. If it is at 0, the PID is used by the man-machine interface. This bit is used so that scale conversions need not be performed on PIDs not used by the CCX 17, and to select PIDs which are used, especially if the number in the PL7 application is greater than 9.
PV_SUP	Double word	I/O	10000	Upper limit of the extent of the process value scale, in physical units (x100) (between - 9 999 999 and + 9 999 999)
PV_INF	Double word	I/O	0	Lower limit of the extent of the process value scale, in physical units (x100) (between - 9 999 999 and + 9 999 999)
PV_MMI	Double word	I/O	0	Image of the process value in physical units (x100)
SP_MMI	Double word	I/O	0	Operator setpoint and setpoint image, in physical units (x100)

**Note :**

The values of the variables used by the CCX 17 are multiplied by 100 in order to give a display of two digits after the decimal point on the CCX 17 (CCX 17 does not use floating point format but has a fixed decimal point format).

**Comments :**

- The internal setpoint is not aligned with the process value in manual mode.
- Scaling only occurs when one of the setpoints (SP or DOP\_SP) is modified.
- The algorithm without integral action ( $TI = 0$ ) performs the following operation :

for  $\varepsilon_t = SP - PV$  ,  
 output  $OUT = KP [ \varepsilon_t + D_t ] / 100 + 5000$   
 where  $D_t =$  derivative action,

The algorithm with integral action ( $TI \neq 0$ ) performs the following operation :

for  $\varepsilon_t = SP - PV$  ,  
 output  $\Delta OUT = KP [ \Delta \varepsilon_t + (TS/10.TI).\varepsilon_t + \Delta D_t ] / 100$   
 $OUT = OUT + \Delta OUT$   
 where  $D_t =$  derivative action.

- On a cold restart, the PID restarts in manual mode, with the output at 0. To force automatic mode or a manual non-zero output after a cold start, the initialization sequence must be programmed **after** the PID call.

**2.2-3 PID programming**

A PID function can be entered in any periodic task (MAST or FAST). The function does not need to have a condition input.

When entering a function, the following window appears allowing the operator to select the required function. A description of the parameters is associated with each function. The parameters required by the function are entered on the right-hand side of the screen.

**Function call**

**Function Information:** Parameters ▼

Family	Lib.Y.	App.Y.		Name	Comment
Movement Command	10	-	▲	PID	Mixed PID controller
Dorpee functions	12	-	▲	PID_MMI	Manage the dedicated man-machine inte>>
Process control	18	-	▲	PwM	Width modulation of a digital value pulse
Single length integers	10	-	▼	SERVO	PID output processing for controlling a >>

**Call Format**

**Parameters of the PROCEDURE :**

Name	Type	Kind	Comment	Entry field	
TAG	STRING	IN	PID name (8 char., for MMI on CCX17	TEMP	▲
UNIT	STRING	IN	Measurement unit (8 char., for MMI on>>	DEGR	▲
PV	WORD	IN	Measurement, format [0; +10000]	SOND1	▼

**Display the Call**

PID ( TEMP,DEGR,SOND1... )

OK
Cancel

The syntax to call the PID function is :

**PID (TAG, UNIT, PV, OUT, AUTO, PARA)**

where :

<b>TAG</b>	char[8]	is a character string input (up to 8 characters) representing the PID name used by the CCX 17.
<b>UNIT</b>	char[6]	is a character string input (up to 6 characters) representing the unit of measurement used by the CCX 17.
<b>PV</b>	integer word	is the input representing the <b>process value</b> for the function.
<b>OUT</b>	integer word	is the function <b>control</b> output,
<b>AUTO</b>	%Mi or %Qi.j bit	is an input/output used both by the CCX 17 and the PID function for the <b>MANU/AUTO</b> operating mode.
<b>PARA</b>	integer table	word table comprising 43 consecutive input/output type words and organized as shown in the table below :

#### Details of the PID parameters : PARA table

Position	Parameter	Function
%MWi	SP	setpoint input,
%MW(i+1)	OUT_MAN	manual control,
%MW(i+2)	KP	serial gain (100 by default),
%MW(i+3)	TI	integral time in 1/10 sec (0 by default),
%MW(i+4)	TD	derivative time in 1/10 sec (0 by default),
%MW(i+5)	TS	sampling period in 1/100 sec,
%MW(i+6)	OUT_MAX	upper control limit,
%MW(i+7)	OUT_MIN	lower control limit,
%MW(i+8):X0 & %MW(i+8):X8	PV_DEV/DEVAL_MMI (bit 2 <sup>0</sup> and 2 <sup>8</sup> of %MW)	selection of the derivative action (bit 0) / PID-MMI inhibit bit (bit 8),
%MD(i+9)	PV_SUP (1 double word : %MD)	upper process value limit,
%MD(i+11)	PV_INF (1 double word : %MD)	lower process value limit,
%MD(i+13)	PV_MMI (1 double word : %MD)	image of the process value for the operator,
%MD(i+15)	SP_MMI (1 double word : %MD)	operator setpoint,

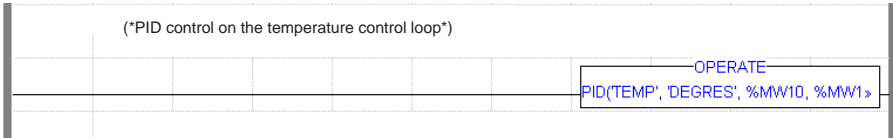
## Important

The other parameters are used for internal PID management and must never be modified by the application.

Call examples :

- **Programmed in Ladder language**

Example in which the process control man-machine interface is used (DEVAL\_MMI = 0)



With PID('TEMP', 'DEGREES', %MW10, %MW11, %M10, %MW20:43)

- **Programmed in List language**

Example in which there is no man-machine interface DEVAL\_MMI = 1 and it is not intended to add one. Note the empty character strings.

! (\*PID correction on the control loop without integrated MMI\*)

```
LD TRUE
  [PID('', '', %IW3.1, %QW4.0, LOOP1_MA, LOOP1_REG:43)]
```

It is important to note the possibility of accessing the PID input variables %IWx.y and output variables %QWx.y (%IW3.1, %QW4.0 in the example above).

- **Programmed in Structured Text language**

! (\*PID correction on the temperature loop\*)

```
PID('TEMP', 'DEGREES', %IW3.1, %QW4.0, LOOP1_MA, LOOP1:43);
```



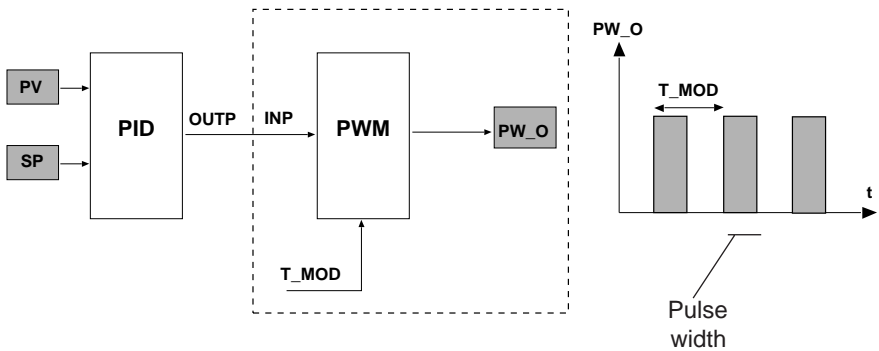
## 2.3 The PWM function

### 2.3-1 Functions

The PWM function is used to perform pulse width control on a discrete output. It is a function which formats the PID output.

The pulse width depends on the PID output (input INP of the PWM function) and the modulation period.

**Use : Discrete control - pulse width modulation**



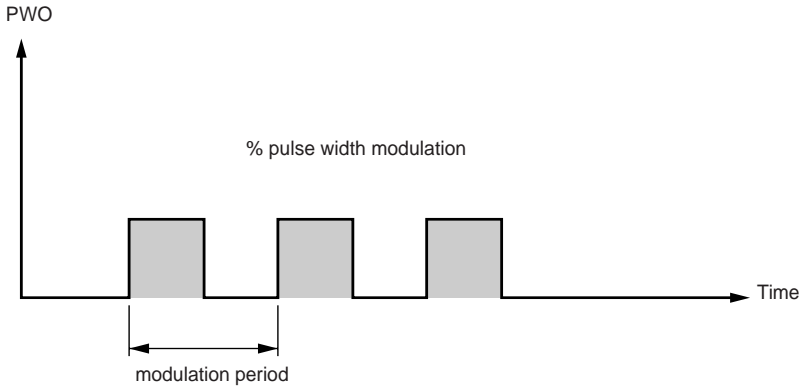
### 2.3-2 Description

The table below describes the user parameters of the PWM function. The syntax to call the function is described in the programming section.

Parameter	Type	Nature	Description
<b>INP</b>	Word	Input	Analog value to be modulated in width (format 0; 10000)
<b>PW_O</b>	Bit	Output	Logic output whose form ratio is the same as that of input INP
<b>T_MOD</b>	Word	I/O	Modulation period expressed in 1/100 second (between 0 and 32767). T_MOD must be greater than or equal to the current task period. It is adjusted by the system in order to form an integer multiple of this period.

At the start of each modulation period  $T\_MOD$ , the activation time, in milliseconds, of output  $PW\_O$  is calculated according to the formula :

State 1 of the pulse =  $INP * T\_MOD / 1000$  (milliseconds)



**In practice :**

- $T\_MOD = TS$  (where  $TS$  is the sampling period for the upstream PID),
- The current task period (in ms) - (Required resolution).  $10 \cdot T\_MOD$ .

**Example :**

The PID is in the MAST task, the MAST period is 50 ms,  $TS = 10$  (ie. 100 ms) and the required resolution is  $1/20$  (time  $T\_MOD$  must contain at least 20 periods of the current task).

$T\_MOD = TS = 10$ .

The task period where the PWM is located must therefore be less than  $10 * 10 / 20 = 5$ .

The PWM function is therefore programmed in the FAST task with a period of 5 ms.

### 2.3-3 PWM programming

A PWM function can be entered in any periodic task (MAST or FAST). The function does not need to have a condition input.

When entering a function, the operator uses the following window to select the required function. A description of the parameters is associated with each function. The parameters required by the function are entered on the right-hand side of the screen.

Function call					
Function Information:			Parameters		
Family	Lib.V	App.V	Name	Comment	
Orphee functions	1.2	-	PID	Mixed PID controller	
Process control	1.8	1.8	PID_MMI	Manage the dedicated man-machine inte>>	
Single length integers	1.0	-	PWM	Width modulation of a digital value pulse	
Single precision reals	1.0	-	SERVO	PID output processing for controlling a>>>	
<b>Call Format</b>					
<b>Parameters of the PROCEDURE :</b>					
Name	Type	Kind	Comment	Entry field	
INP	WORD	IN	Digital value to modulate	%MW11	
PW_0	EBDOL	OUT	Discrete output with a cyclic ratio equal to >>	%Q1.3	
PARA	ARI_W	INOUT	PWM parameters (5 word table)	%MW90:5	
<b>Display the Call</b>					
PWM ( %MW11,%Q1.3,%MW90:5 )					
			OK Cancel		

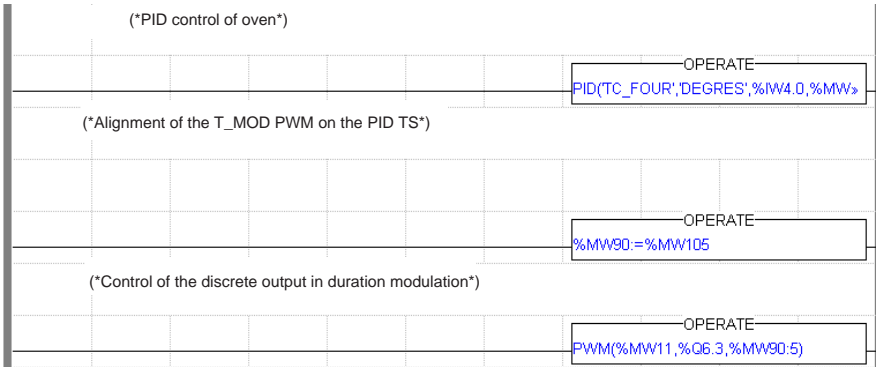
The syntax to call the PWM function is :

**PWM (INP , PW\_0 , PARA)**

<b>INP</b>	Word	Value to be modulated
<b>PW_0</b>	Bit %Q or %M	Modulated discrete output
<b>PARA</b>	Word [5]	5-word table, the first word of which corresponds to the parameter T_MOD. The following words are used internally by the function and must never be modified by the application.

Examples of use :

- **Programmed in Ladder language :**



With PWM(%MW11,%Q6.3,%MW90:5)

- **Programmed in List language :**

```
!
(* Oven control PID *)
LD TRUE
[PID('TC_OVEN','DEGREES',%IW4.0,%MW11,%M10,%MW100:43)]
!
(* Alignment of the T_MOD PWM on the PID TS *)
LD TRUE
[%MW90:=%MW105]
!
(* Control of the discrete output in duration modulation *)
LD TRUE
[PWM(%MW11,%Q6.3,%MW90:5)]
```

- **Programmed in Structured Text language :**

```
! (* Oven control PID*)
PID('TC_OVEN','DEGREES',%IW4.0,%MW11,%M10,%MW100:43);
%MW90:=%MW105;
PWM(%MW11,%Q6.3,%MW90:5);
```

## 2.4 The SERVO function

### 2.4-1 Functions

The SERVO function is used to perform PID control using a motor as an actuator with 2 discrete actions (UP and DOWN). This is an output condition to be connected in cascade with the analog output of a PID.

Where position feedback is used, control of the valve position is performed by means of the INP (setpoint) and POT (position measurement) inputs.

When feedback is not physically present, the algorithm does not use the absolute PID output, but rather the output variation. The output UP (or DOWN, depending on the sign of the variation) is set to 1 for a period of time proportional to the period of opening of the actuator and to the variation value. In addition, the concept of a minimum pulse time is introduced.

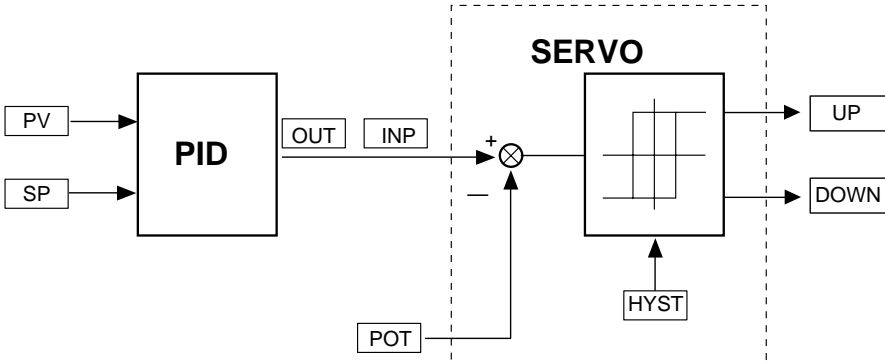
### 2.4-2 Description

The table below describes the user parameters of the SERVO function. The syntax to call the function is described in the programming section.

Parameter	Type	Nature	Description
<b>INP</b>	Word	Input	Position setpoint (format 0/+10000). Must be connected to the PID output.
<b>POT</b>	Word	Input	Position feedback (format : 0/+10000) (0 : valve closed; 10000 : valve open). If there is no feedback, POT must be initialized to -10000. This particular value means "no feedback".
<b>UP</b>	Bit	Output	Output signal for the UP operating direction of the motor.
<b>DOWN</b>	Bit	Output	Output signal for the DOWN operating direction of the motor.
<b>PID</b>	Word table	Input/output	Parameter table of the upstream PID. Used if there is no feedback for synchronization with the upstream PID.
<b>T_MOTOR</b>	Word	Input/output	Valve opening time. Expressed in hundredths of a second. Used if there is no feedback (between 0 and 32767).
<b>T_MINI</b>	Word	Input/output	Minimum pulse time. Expressed in hundredths of a second. Used if there is no feedback (between 0 and 32767).
<b>HYST</b>	Word	Input/output	Value of the hysteresis to be applied to the discrete outputs. Format 0/+10000. Used if feedback exists.

### • SERVO operation with position feedback

The SERVO function performs motor position control as a function of a position setpoint INP from a PID output in the format 0/10000 and a position measurement POT. The servo-control algorithm is a relay with hysteresis.



In this case, parameters PID, T\_MOTOR and T\_MINI are not used.

#### Note :

It is possible to program the SERVO function call in the FAST task to increase the resolution of the motor control.

### • SERVO operation without position feedback (POT = -10000)

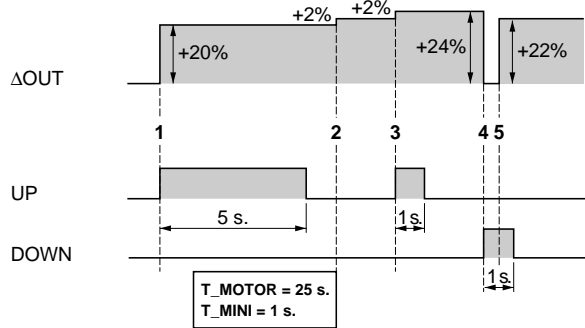
In this case, the SERVO function is synchronized with the upstream PID algorithm using the PID parameter table, transmitted as a parameter to the SERVO function.

The input for the algorithm is the PID output variation, which it converts into the pulse duration, according to the formula :

$$T\_IMP = \Delta OUT \times T\_MOTOR / 1000 \text{ (in ms)}$$

The time obtained is added to the remaining time of the preceding cycles : in fact, what is not "used up" during one cycle is memorized for the following cycles.

This ensures satisfactory operation, in particular when the control is changed abruptly (eg : PID setpoint increment) and in manual mode.

**Example :**

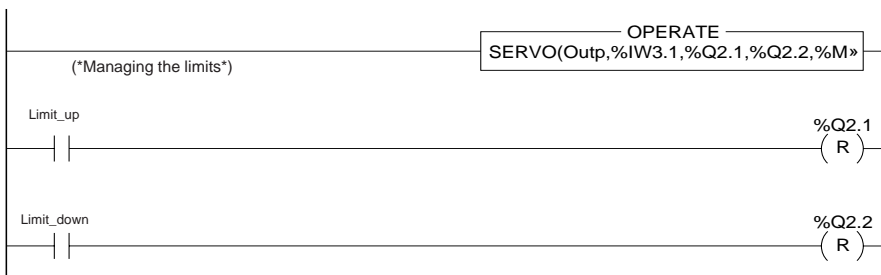
1. Variation of the PID output is +20% (the  $T_{MOTOR}$  pulse = 25 s. for a variation of 100%). In this case, the pulse activates the UP output for a duration of 5 s,
2. Variation of the PID is +2%, which would correspond to a pulse of 0.5 s. This pulse is shorter than  $T_{MINI}$  (=1 s.) and does not activate the outputs,
3. A second variation of +2% appears. The function adds this variation to the preceding one (which corresponded to a variation lower than the minimum value) in order to perform its calculation. This corresponds to a total positive variation of +4%, and therefore to a pulse of 1 s on the UP output,
4. A variation of -24% appears. The pulse activated is therefore 6 s long on the DOWN output,
5. Before the next second has elapsed, another variation of +22% returns the system to a total variation of 2% < than the  $T_{MINI}$  variation (4%). The function completes activation of the minimum pulse of 1 s.

**Note :**

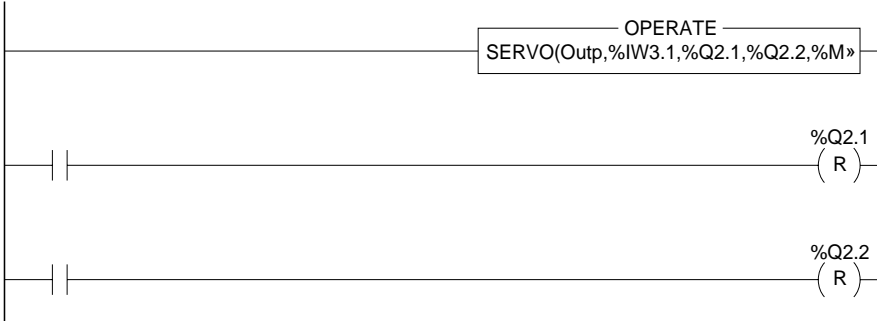
It is possible to program the SERVO function call in the FAST task to increase the resolution of the motor control.

**Notes :**

- The SERVO function does not manage the position limits, but it is easy to manage them within the application : if a limit is detected, the corresponding output must be forced to 0 (UP for the upper limit, DOWN for the lower limit).



- It is possible to change from one operating mode to another (eg : in the event of a feedback error, changing from operation with feedback to operation without feedback).



### 2.4-3 SERVO programming

A SERVO function can be entered in any periodic task (MAST or FAST). The function does not need to have a condition input.

When entering a function, the operator uses the following window to select the required function. A description of the parameters is associated with each function. The parameters required by the function are entered on the right-hand side of the screen.

**Function call**

**Function Information:** Parameters ▾

Family	Lib.V.	App.V.	Name	Comment
Orphee functions	1.2	-	PID	Mixed PID controller
Process control	1.8	1.8	PID_MMI	Manage the dedicated man-machine inte>
Single length integers	1.0	-	PWM	Width modulation of a digital value pulse
Single precision reals	1.0	-	SERVO	PID output processing for controlling >>

**Call Format**

**Parameters of the PROCEDURE :**

Name	Type	Kind	Comment	Entry field
INP	WORD	IN	Position setpoint, format [0;10000] (to >>	%MW11
PDT	WORD	IN	Position copy, format [0;10000] (-10000 >>	%MW12
UP	EBOOL	OUT	Discrete output, UP direction	%Q11

**Display the Call**

SERVO ( %MW11,%MW12,%Q11... )



The syntax to call the SERVO function is :

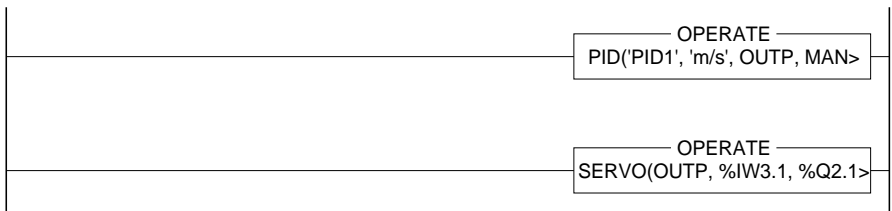
### SERVO (INP, POT, UP, DOWN, PID, PARA)

<b>INP</b>	Word	Corresponds to the INP input designating the position setpoint,
<b>POT</b>	Word	Corresponds to the POT position feedback input,
<b>UP</b>	Bit %Q or %M	Corresponds to the UP output,
<b>DOWN</b>	Bit %Q or %M	Corresponds to the DOWN output,
<b>PID</b>	Word [43]	Table corresponding to the internal upstream PID table,
<b>PARA</b>	Word [10]	10-word table, the first 3 words of which correspond to parameters T_MOTOR, T_MINI and HYST : the others are used internally by the function. They must <b>never be modified by the application</b> .

All parameters **must** be entered, regardless of the operating modes used.

Examples of use :

- **with position feedback : programmed in Ladder language**



```
PID('PID1', 'm/s', PV, OUTP, MAN_AUTO, %MW100:43)
SERVO(OUTP, %IW3.1, %Q2.1, %Q2.2, %MW100:43, %MW180:10)
```

- **without position feedback : programmed in List language**

```
! LD TRUE
  [PID('PID1', 'm/s', PV, OUTP, MAN_AUTO, %MW100:43)]
! LD TRUE
  [SERVO(OUTP, -10000, %Q2.1, %Q2.2, %MW100:43, %MW180:10)]
```

- **without position feedback : programmed in Structured Text language**

```
! PID('PID1', 'm/s', PV, OUTP, MAN_AUTO, %MW100:43);
! SERVO(OUTP, -10000, %Q2.1, %Q2.2, %MW100:43, %MW180:10);
```

---

## 2.5 Behavior of functions in various PLC operating modes

---

This paragraph describes the behavior of functions during different types of startup :

- cold restart (new application, cartridge change, etc),
- warm restart (power return, with no change of application context),
- first execution after addition of a function via modification online.

---

### 2.5-1 Cold restart

On a cold restart, the PLC may start automatically in RUN (depending on the application configuration). The behavior of the PID controller functions is designed with safety in mind : in manual mode, outputs at 0. In addition, this makes it possible to change the PLC to RUN without performing PID adjustment, and then to perform debugging with the CCX 17 (adjustment can only be performed in RUN).

---

### 2.5-2 Warm restart

When power returns after a power break (regardless of its length) and if the application context has not been lost or modified, the functions continue from the state they were in before the power break. If the user wishes the system to behave differently, it is his responsibility to test system bit %S1 and associate the required processing with it (forcing to manual mode, etc).

**Note :**

On Micro/Premium, the PLC real-time clock can be used to determine the length of the last power break.

---

### 2.5-3 Addition of a new function call in online mode

Following the addition of a new process control function call in online mode, initialization identical to the cold restart is performed.

**Note :**

In order to be seen as a new function, the latter must use a new parameter table. The removal of a PID, followed by the addition of a PID using the same parameter table, is therefore not considered to be an addition of a new PID. In this case, the PID is executed in the state and using the parameters of the preceding PID.

### 3.1 Man-machine interface on CCX 17

The CCX 17 is used to display and control all the modifiable parameters of a PID controller without having to program specific PLC application programs.

The man-machine interface function integrates an application program for controlling and adjusting CCX 17 application PIDs on the CCX 17. It manages three types of screen on the CCX 17 enabling the selection of a PID, display and control of this PID and adjustment of the PID parameters. It can easily be inserted into any man-machine interface application on the CCX 17.

**Note:**

Warning : the function is only effective if the PLC is in RUN.

**There is no limit to the number of PIDs** in the application. However, a maximum of 9 PIDs can be accessed by the man-machine interface function on CCX 17-20 and CCX 17-30.

CCX control buttons are used to navigate between screens, while the up and down arrow keys are used to navigate within screens. The navigation offered is "vertical". The user must always return to the loop selection screen to have access to the values of other loop controllers.

The display is 4 lines long (8 lines for CCX 17-30) with 40-character messages.

**Operating mode :**

The MOD key is used to switch from display mode to data entry mode (in this case, the selected value flashes).

On the same screen, the data entry mode remains active for all fields. If MOD is pressed again, the user quits data entry mode (the flashing stops).

In data entry mode, a parameter modification is taken into account by pressing the ENTER key.

The man-machine interface is very simple to use : the PID\_MMI function(s) are executed on each cycle (unconditional call). There is one single call to the PID\_MMI function to manage all the application PIDs. However, the PID\_MMI function is called by the CCX 17 connected to the PLC.

The application PIDs are detected automatically by the PID\_MMI function, including when they are added or removed in RUN mode. No declaration therefore has to be made.

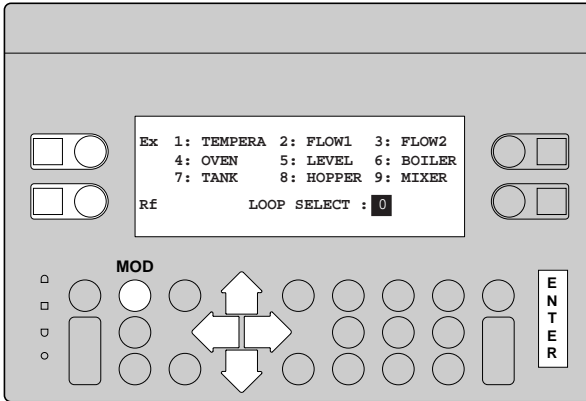
Addressing of the required PID controller is performed by the "TAG" parameter of the PID function and its selection depends on the value of function parameter "DEVAL\_MMI". (The PID\_MMI function only takes into account PIDs whose parameter DEVAL\_MMI = 0).

**Note :**

The maximum number of PIDs which can be used by the CCXs is 9, regardless of the number of CCXs connected.

## 3.2 Description of predefined screens

### 3.2-1 Selecting a loop



The loop selection screen can accommodate up to 9 loops.

This screen displays all the names of the loops installed under PL7 Micro.

A number is associated with each name (from 1 to a maximum of 9).

To control one of the loops, the operator must enter the corresponding number.

Once the loop number is entered, the loop control screen is displayed.

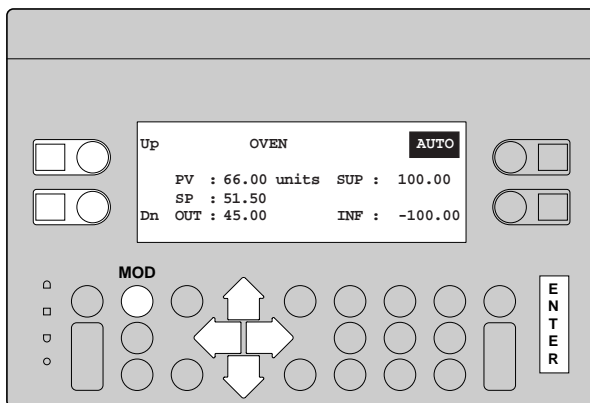
Pressing the Exit (Ex) button quits the PID control screens.

Pressing the Refresh (Rf) button refreshes the screen. This operation is necessary after loops have been deleted or added by PL7 in online mode.

**Note :**

If the application does not have any PIDs which can be accessed by the CCX 17 (either there are no PIDs in the application, or the DEVAL\_MMI parameters of existing PIDs are all at 1), the message "NO PID" is displayed. The Exit and Refresh buttons retain their function.

### 3.2-2 Controlling a loop



This screen is used to control setpoint, control and Manu/Auto mode values. The values PV\_INF and PV\_SUP are also displayed and can be controlled from this screen.

The Manu/Auto field appears in reverse video. By pressing the associated control button, the user switches from one mode to the other. Controlling outputs in automatic mode is not authorized.

The up and down arrows are used to move between data entry fields. The operating mode is the following : once the screen is displayed, the value **SP** is selected (reverse video), and then, in the order of pressing the down key, OUT (if manual), INF and SUP. By pressing MOD, the user switches to data entry mode (press MOD again to quit this mode).

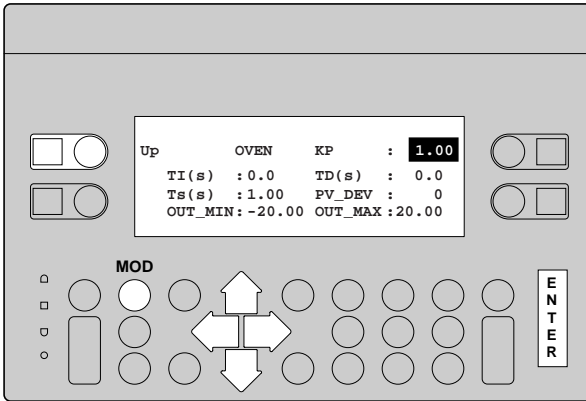
The **Dn** button accesses the adjustment screen. To return to the loop selection screen, press the **Up** button. (The values PV, SP, OUT, INF and SUP are reals with 2 significant digits after the decimal point).

PV, SP, INF and SUP are in physical units. OUT is expressed as a percentage (for an explanation of the mnemonics, see section 2.2.2).

**Note :**

When a field is flashing (data entry mode), the value is not refreshed if it is modified by the application or PL7.

### 3.2-3 Adjusting a loop



Navigation and data entry are performed in a similar way to that described for the preceding screen (navigation using the up and down arrows).

Once the screen is displayed, the value KP is selected (reverse video).

The parameter KP does not have units. TI, TD and TS are in seconds. OUT\_MIN and OUT\_MAX are expressed as a percentage (for an explanation of the mnemonics, see section 2.2.2).

Pressing the **Up** button returns the user to the loop control screen.

### 3.3 The PID\_MMI function

#### 3.3-1 PID\_MMI functions

The PID\_MMI function is used to communicate with the TSX Micro/Premium PLCs to which the CCX 17 is connected.

A PID\_MMI function is required by the CCX 17 to control, display and adjust the application PIDs.

#### 3.3-2 Description of the PID\_MMI

The syntax to call the PID\_MMI function is :

**PID\_MMI (ADDR, EN, BUTT, PARA)**

The role of the various PID\_MMI function parameters is the following :

Parameter	Type	Nature	Description
ADDR	6-word table	Input	CCX 17 address.
EN	Bit	Input/output	Activation of the process control man-machine interface. The application sets this bit to 1, the PID_MMI function resets it to 0 when exiting from the PID control man-machine interface (by pressing EX).
BUTT	5-bit table	Input/output	Bits associated with the CCX 17 buttons. These bits are used to control the various screens as well as Manu/Auto.
PARA	62-word table	Input/output	PID_MMI parameters. The first 4 are communication report words.

**Note :**

The 4 report words are common to all asynchronous communication functions (communication, integrated MMI and PID\_MMI function blocks).

However, the PID\_MMI function block automatically manages these words and the application must never modify them. They are provided for information only.

For more information, see Volume 1, part D, section 3.5, Man-machine interface function.

Example of CCX 17 address :

If the CCX 17 is connected directly to the front panel of TSX Micro/Premium PLCs (UNI-TELWAY), it is at the UNI-TELWAY slave addresses 4-5 or 6-7.

Coding can be performed :

- by assigning an immediate value :
  - PID\_MMI(ADR#{0.254}0.0.4, ....)
  - or simply
  - PID\_MMI(ADR#0.0.4, ....)
- by assigning a 6-word table :
  - %MW10:6 := ADR#0.0.4
  - PID\_MMI(%MW10:6, ...)

For more information, see part K section 3.2, Man-machine interface function.

### 3.3-3 PID\_MMI programming

The call to the PID\_MMI function **must not be conditioned**.

It may be placed anywhere in the period of the slowest task containing PIDs.

**Function call**

Function Information: Parameters

Family	Lib.V.	App.V.	Name	Comment
Orphee functions	1.2	-	PID	Mixed PID controller
Process control	1.8	1.8	PID_MMI	Manage the dedicated man-machine inte>
Single length integers	1.0	-	PWM	Width modulation of a digital value pulse
Single precision reals	1.0	-	SERVO	PID output processing for controlling a >>

**Call Format**

Parameters of the PROCEDURE :

Name	Type	Kind	Comment	Entry field
ADDR	AR_V	IN	CCX17 target address in the network: top>	ADR#0.0.4
EN	EBOOL	IN/OUT	MMI activation on CCX17	%M1
BUTT	AR_X	IN/OUT	5-bit table assigned to the CCX17 control>	%M10:5

**Display the Call**

PID\_MMI ( ADR#0.0.4,%M1,%M10:5,%MW45:62 )

OK Cancel

Example :

FAST task at 10 ms and MAST task at 50 ms, both containing PIDs.

In this case, the PID\_MMI must be programmed in MAST.

#### Note :

This can be the FAST or MAST task.



**Synchronization with a standard man-machine interface application program**

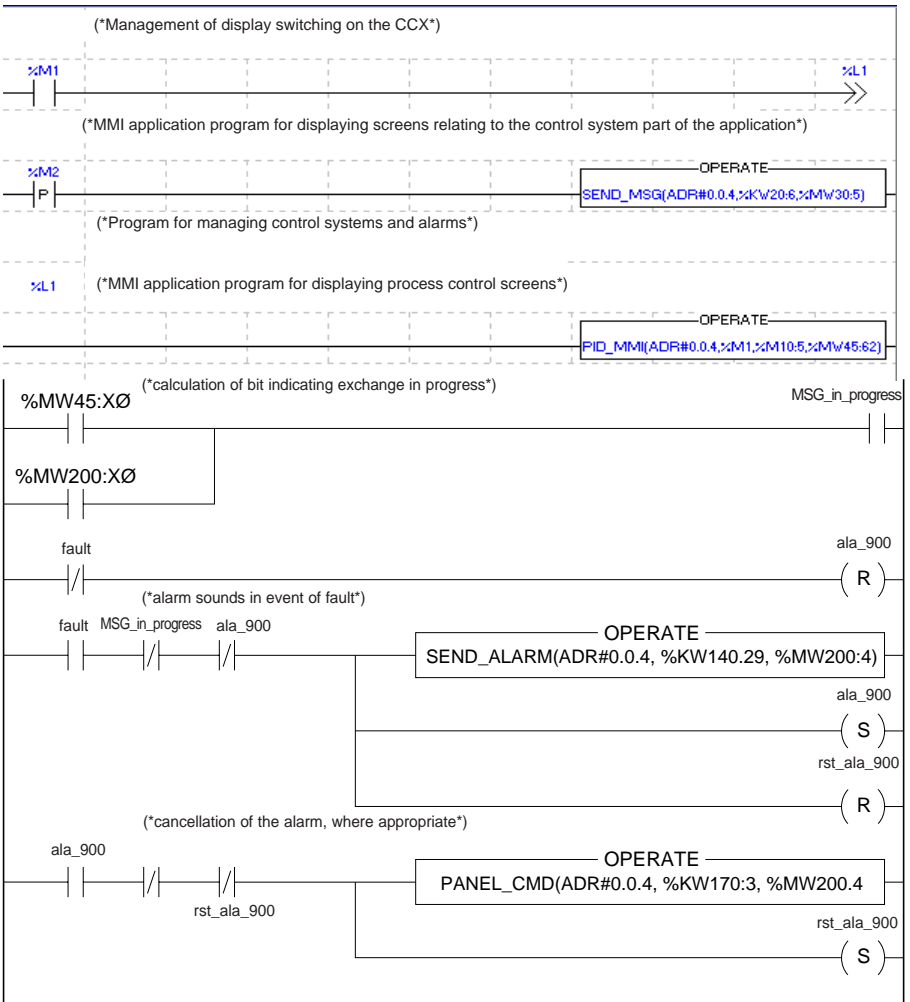
The CCX 17 can be used to display types of screen other than control screens. The EN bit is used to activate/deactivate the process control man-machine interface. Setting EN to 1 activates the process control man-machine interface and displays the PID selection screen.

Example :

%M1 is associated with bit EN (man-machine interface display switch)

Note that the alarm management application program is always activated, as is the process control man-machine interface.

**Programming in Ladder language**



**Note :**

When the PID\_MMI is activated (EN at 1), it assigns the CCX 17 control buttons. If the application program uses these buttons for functions other than process control, they must be reassigned on the falling edge of EN (use the ASSIGN\_KEYS function, see part K).

However, if the CCX 17 is used only for process control, it is recommended to SET bit EN unconditionally in the PL7 application.

**Selecting PIDs managed by the PID\_MMI function**

Reminder : each PID has a bit-type DEVAL\_MMI parameter. If this bit is at 1, the PID is not managed by PID\_MMI. This is the only level of protection available.

In addition, if the application has more than 9 PIDs, this is the way to manage those which are processed by PID\_MMI.

**Alarm management**

It is up to the user to create his own alarm management by program.

It can be superimposed on the process control screens.

---

### Using several PID\_MMI functions

It is possible to connect several CCX 17 terminals to the same PLC. It may therefore be useful to have several PID\_MMIs in the same application.

In this case, the various PID\_MMIs must be executed consecutively (no PID call in between) from the same PL7 task.

---

### 3.3-4 Behavior of the PID\_MMI function depending on the operating modes of the PLC and CCX 17

The PID\_MMI function takes into account the PLC and CCX 17 operating modes :

#### Warm restart

If a problem such as a micro-break on the supply to the PLC occurs when a message is being sent, the command is not repeated. To refresh the screen, simply change screens and then return or refresh the loop selection screen if on that screen.

#### STOP/RUN and RUN/STOP changeover

In STOP mode, the PID\_MMI function is no longer active. Nonetheless, it is still possible to enter parameters for the screen displayed.

On the change from STOP to RUN, the function continues from its current state before the transition to STOP mode.

#### CCX 17 power break or CCX 17 disconnection/reconnection

On a CCX 17 power break or when it is reconnected, it reinitializes communication with the PLC. Periodically, the PID\_MMI reassigns the control buttons of the CCX 17. Therefore, after a maximum of 20 seconds, one of the process control screens will be displayed by pressing one of the first three buttons (preferably the Ref or Dn button, that is, the left-hand button on the second row).

#### Note :

It is also possible to detect whether the CCX 17 is present via the application program, using the language words associated with the communication channels (see part Volume 1, part D, section 4.1).

#### Note

If an alarm (from the man-machine interface application program) is activated during display of one of the three process control screens, the CCX 17 screen is then dedicated to the management of alarm messages.

On returning to the process control man-machine interface, the screen is incomplete. Up/Dn or Refresh can be used to refresh this screen.

#### Cold restart

Process control screens are only reinitialized on a cold restart.



## 4.1 Adjusting loops/debugging the application

### 4.1-1 With CCX 17

The following table lists the PID parameters accessible by the man-machine interface.

Name	Type	Access by CCX 17 (in process control mode)
<b>TAG</b>	Character string (8)	Read, in all screens
<b>UNIT</b>	Character string (6)	Read, control screen
<b>PV</b> (process value)	Single word	Read, control screen
<b>OUT</b> (command)	Single word	Read, control screen
<b>AUTO</b> (operating mode)	Bit	Read/write, control screen
<b>PARA</b>	43-word table	

### Details of PARA

Name	Type	Access by CCX 17 (in process control mode)
<b>SP</b> (setpoint)	Word	Read/write, control screen
<b>OUT MAN</b> (manual control)	Word	Read/write, control screen
<b>KP</b> (gain)	Word	Read/write, adjustment screen
<b>TI</b> (integral)	Word	Read/write, adjustment screen
<b>TD</b> (derivative)	Word	Read/write, adjustment screen
<b>TS</b> (sampling period)	Word	Read/write, adjustment screen
<b>OUT_MAX</b> (upper control value)	Word	Read/write, adjustment screen
<b>OUT_MIN</b> (lower control value)	Word	Read/write, adjustment screen
<b>PV_DEV</b> (derivative on process value (0) or deviation (1))	Byte	Read/write, adjustment screen
<b>DEVAL_MMI</b>	Byte	
<b>PV_HI</b> (upper process value limit)	Double word	Read/write, control screen
<b>PV_LO</b> (lower process value limit)	Double word	Read/write, control screen
<b>PV_MMI</b> (image of process value at the physical scale)	Double word	Read/write, control screen
<b>SP_MMI</b> (image of the setpoint at the physical scale)	Double word	Read/write, control screen
Private variables	26 words	

---

On start-up, the operator goes to the control screen of the loop to be adjusted. He must ensure that the loop controller is in manual mode and the parameters are configured with the required values.

By varying the control output, he steers the process to the required setpoint, monitoring the changes in the process value on-screen. The loop can then be adjusted by selecting one of the methods described in the appendix of part J. The loop adjustment screen is displayed. The MANU/AUTO key is used to switch from AUTO to MANU and vice versa. Each parameter is accessed using the up and down arrow keys and their content can be modified using the MOD key.

---

#### 4.1-2 Without CCX 17

The method is identical, but the parameters are accessed via an animation table where the operator can display and modify the loop controller parameters.

#### Warning

In PL7 Junior, the scale of the parameters is different. They are all in integer format (see section 2.2-1).

In this case, it is strongly recommended to create symbols for the PID parameters in the variables editor.

It is also recommended to set DEVAL\_MMI to 1, which improves PID performance (see the section on function characteristics).

#### Note :

MMI parameters must not be modified.

## 4.2 Diagnostics / Maintenance

### Limitations

The speed of a control loop depends on the interfaces used :

- analog input modules have a scan time of 6 ms or 550 ms, depending on their reference and the number of channels used,
- a counter module used in a FAST task allows scans of 5 to 10 ms.

### Diagnostics

#### • Process control functions (PID, SERVO, PWM)

These functions do not require diagnostics information.

The possible errors are :

Behavior	Diagnostics
All the function parameters are at 0.	- The function has not been called, - The PARA table is too short.
The output remains at 0 even in manual mode.	- The function has not been called, - The PARA table is too short.
The integral action is not functioning (TI of the PID not taken into account)	- The task is configured cyclically instead of periodically.
TS cannot be modified. (TS of the PID remains at 0)	- The task is configured cyclically instead of periodically.
In automatic mode, the PID output remains at 0.	- Check the output limit (OUT_MIN and OUT_MAX)

#### • PID\_MMI function

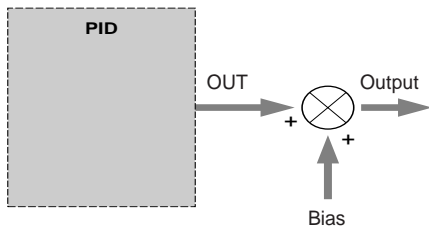
Behavior	Diagnostics
After disconnection or a power break on the CCX 17, the last process control screen is not redisplayed.	- This is normal; press a button (B1, B2 or B3) to redisplay a process control screen (wait a few seconds).
Some of the PIDs are not taken into account.	- These PIDs are not executed, - Bits DEVAL_MMI of these PIDs are at 1, - These PIDs are in a task which is slower than the PID_MMI.
No screens are displayed when EN switches to 1.	- Check whether the application is overwriting words in the PID_MMI 62-word table.

## 4.3 Additional programming

### 4.3-1 Adding a bias to the output

This function is used to cancel the static error at a given operating point when the PID is used in P or PD mode.

The simplified schematic is thus :



#### Programming :

```
PID('level', 'mm', PV, OUT, AUTO, PARA:43)
```

```
Output := OUT + Bias
```

```
OUT_MAX := 10000 - BIAS (where the bias is > 0)
```

```
OUT_MIN := -BIAS (where the bias is < 0)
```

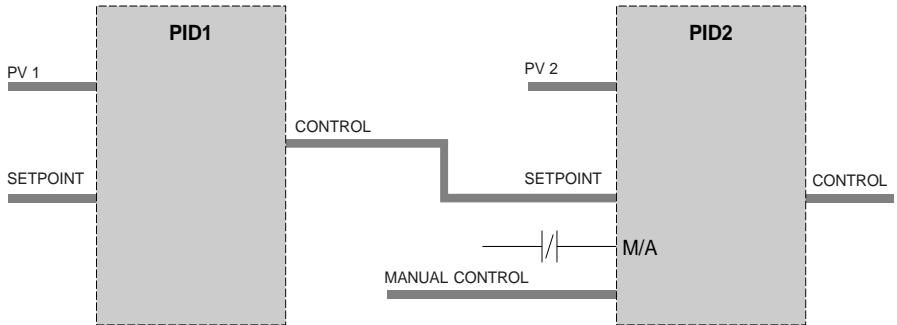
#### Note :

When the PID is executed, OUT\_MAX and OUT\_MIN are included in the range 0-10000. The last two instructions are used specifically to manage desaturation of the PID integral.



### 4.3-2 PIDs in cascade

Two PIDs in cascade are represented schematically as follows :



Generally, the downstream PID must be faster, and therefore the sampling time  $TS$  of PID1 will be shorter than that of PID2.

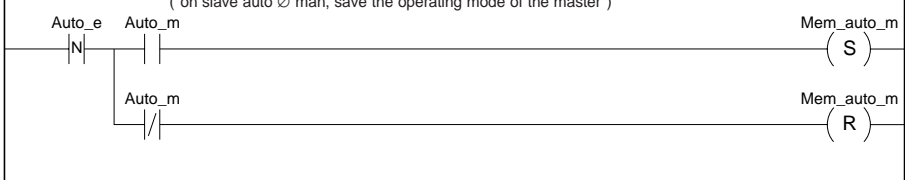
#### Programming :

The control output of PID1 coincides with the setpoint of PID2. The 2 PIDs must simultaneously change to Manual mode. The operator must then act on the manual control of the slave PID (downstream).

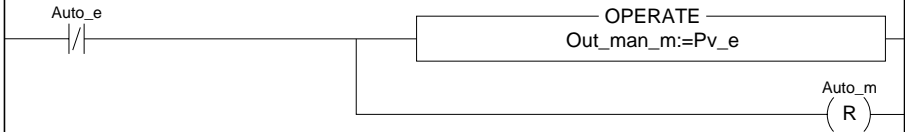
Example (see following page) :

MAST-MAIN

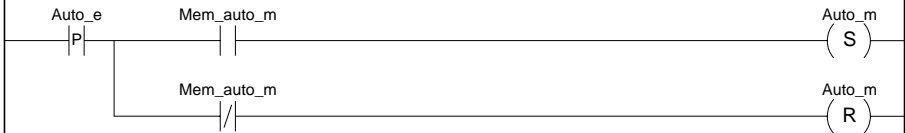
(\*on slave auto  $\emptyset$  man, save the operating mode of the master\*)



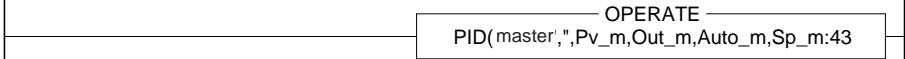
(\*if slave in manual mode, change the master to manual mode and align its output to the slave PV\*)



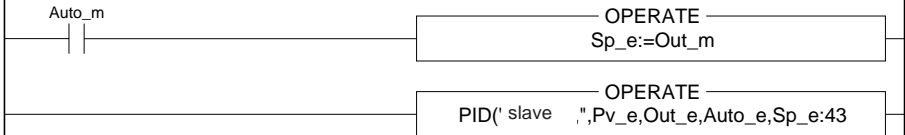
(\*on slave man  $\emptyset$  auto, restore the saved operating mode of the master\*)



(\*master PID\*)



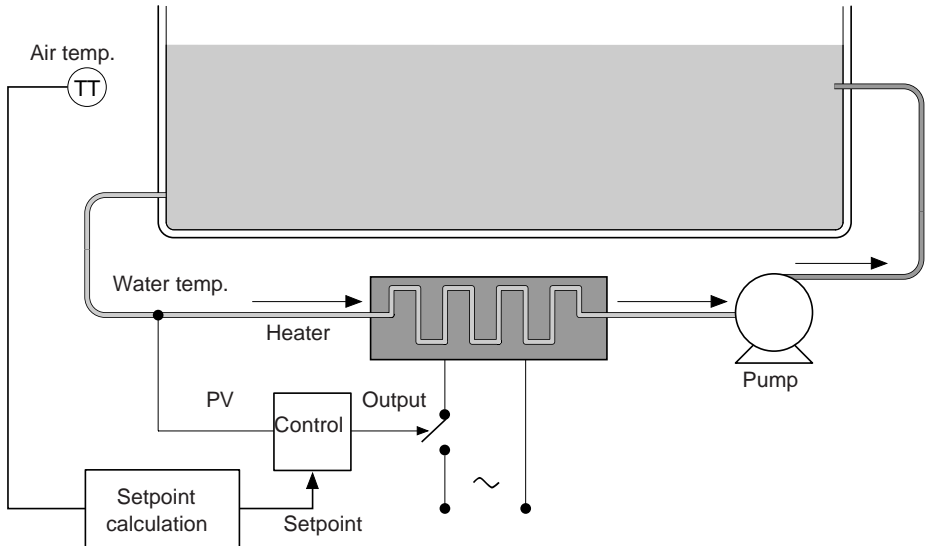
(\*if master is in auto mode, its output is wired to the slave setpoint\*)



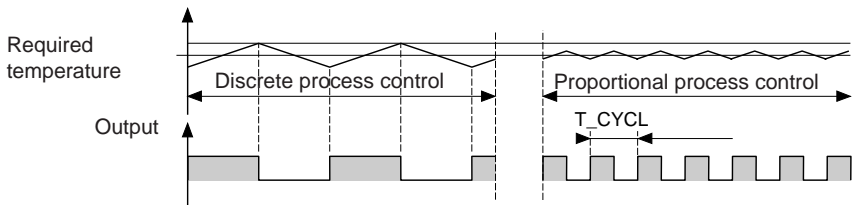
## 5.1 Temperature control

### 5.1-1 Application description

Maintaining the water temperature of an open-air swimming pool at a required value. This value is itself determined by the temperature of the ambient air.



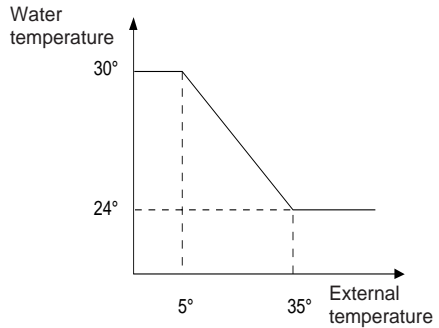
Discrete process control is generally used in this type of installation. In this example, this could be replaced by modulated output proportional control, which should allow reduction of temperature oscillations around the required value.



The water temperature and ambient temperature are measured using Pt 100 type temperature probes.

The water temperature setpoint depends on the external temperature as defined by the following :

- A HIGH TEMPERATURE alarm will be activated if the water temperature exceeds 32°C.
- A LOW TEMPERATURE alarm will be activated if it falls below 22°C.
- A PROCESS CONTROL FAULT alarm will be activated if the SETPOINT/ PROCESS VALUE deviation exceeds 2°C in either direction.
- Process control will be disabled (output at 0) if the pump stops.



### 5.1-2 Hardware configuration

This application requires :

- a TSX 57-10 PLC,
- a TSX AEY 414 analog input module,
- a TSX DEY 32D2K discrete input module,
- a TSX DSY 08R5 discrete output module.

Discrete output %Q2.0 is assigned to control of the heater.

Discrete output %Q2.1 is assigned to control of the pump.

Discrete outputs %Q2.2, %Q2.3 and %Q2.4 are assigned to the alarms.

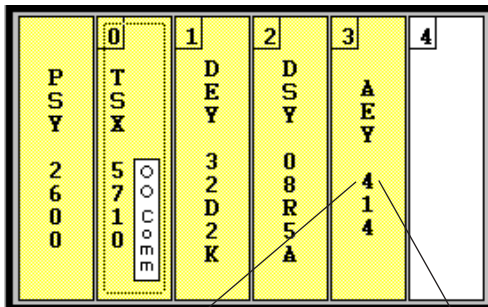
Bit %M0 is used to select the AUTO/MANU operating mode of the controller.

Discrete inputs %I1.1 and %I1.2 are used to modify the setpoint value in AUTO mode and the output value in MANU mode using the following algorithm :

- %I1.1 = 1 increase of 0.1 % per scan,
- %I1.2 = 1 decrease of 0.1 % per scan.

Input %I1.3 provides information on the pump status.

%IW3.0 and %IW3.1 : value of the analog inputs.



Chan.	Symbol	Range	Filter	Scale	Task
0	Water_temp	Pt100	0	%..	MAST
1	Air_temp	Pt100	0	%..	MAST
2	Motor_overheat	Thermo J	0	%..	MAST
3	Heater_overheat	Thermo K	0	%..	MAST

**Channel Parameters 0**

Broken Wire Test      Unit:  C     F

Task :  [v]

Range :  [v]

Temperature range: from -2000 to 8500 1/10 °C

Scale

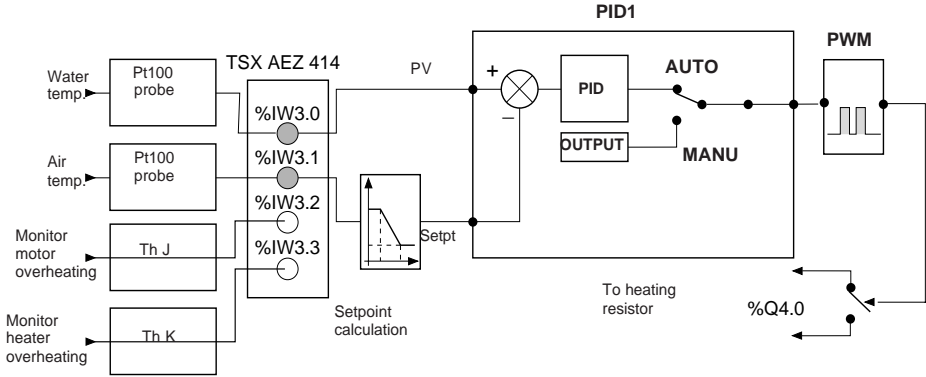
Standardized      Scaling

1/10 °C->0

1/10 °C->10000

Filtering  [v]

### 5.1-3 Simplified schematic of the control loop



The direction of action of the PID controller is INVERSE (an increase in the process value must be matched by a decrease in the output).

### 5.1-4 Programming

#### Suggested method

Block PID1 is assigned to temperature control. The water temperature setpoint is calculated from the air temperature.

On a power return, process control operation is selected and the pump is started.

The state of the controller is determined by the operating status of the pump. If the latter is faulty, the PID changes to MANU mode and the output is forced to 0.

Status word bits (high process value threshold, low process value threshold, high deviation threshold and low deviation threshold) are used to activate the alarms.

The coefficients of the PID loop will be initialized at :

- |            |                 |         |                  |
|------------|-----------------|---------|------------------|
| • Kp = 600 | } Data<br>scale | Kp = 6  | } CCX<br>display |
| • Ti = 300 |                 | Ti = 30 |                  |
| • Td = 50  |                 | TD = 5  |                  |

These values can, of course, be refined during the course of later adjustments.

---

**RACK CONFIGURATION**

Rack number : 0

Rack reference : TSX RKY 6E

Power supply reference : TSX PSY 2600

Slot	Family	Reference
0	Processors	TSX 5710
1	Discrete	TSX DEY 32D2K
2	Discrete	TSX DSY 08R5A
3	Analog	TSX AEY 414

**TSX DEY 32D2K [RACK 0 POSITION 1]**
**Module identification****Product ref.** : TSX DEY 32D2K**Designation** : 32E 24 VDC SINK CONN**Address** : 001**Symbol** :**Channel parameters**

Channel	Address	Symbol	Task
0	%I1.0		MAST
1	%I1.1	Increm_setpt	MAST
2	%I1.2	Decrem_setpt	MAST
3	%I1.3	Pump_status	MAST
4	%I1.4	Act_pump	MAST
5	%I1.5		MAST
6	%I1.6	Enable_mmi_pcl	MAST
7	%I1.7		MAST
.	.	.	.
31	%I1.31		MAST

## TSX DSY 08R5A [RACK 0 POSITION 2]

### Module identification

Product ref. : TSX DSY 08R5A      Designation : 8Q RELAY 100VA, TERM  
Address : 002      Symbol :

### Channel parameters

Channel	Address	Symbol	Task	Fallback mode	Fallback value	Reactivation
0	%Q2.0	Heater_ctrl	MAST	Fallback	Fallback to 0	Programmed
1	%Q2.1	Pump_ctrl	MAST	Fallback	Fallback to 0	Programmed
2	%Q2.2	High_temp_alarm	MAST	Fallback	Fallback to 0	Programmed
3	%Q2.3	Low_temp_alarm	MAST	Fallback	Fallback to 0	Programmed
4	%Q2.4	Pcl_fit_alarm	MAST	Fallback	Fallback to 0	Programmed
5	%Q2.5		MAST	Fallback	Fallback to 0	Programmed
6	%Q2.6		MAST	Fallback	Fallback to 0	Programmed
7	%Q2.7		MAST	Fallback	Fallback to 0	Programmed

## TSX AEY 414 [RACK 0 POSITION 3]

### Module identification

Product ref. : TSX AEY 414      Designation : 4I ANA. MULTIRANGE  
Address : 003      Symbol :

### Common parameters

Type : Inputs      Test for presence of terminal block : Inactive  
Cold junction : Internal

### Channel parameters

Channel	Address	Symbol	Range	Scale	Min	Max	Unit	Filter	Task	Wiringtest
0	%IW3.0	Water_temp	Pt100	User	0	500	°C	0	MAST	Inactive
1	%IW3.1	Air_temp	Pt100	User	-200	800	°C	0	MAST	Inactive
2	%IW3.2	Motor_	Thermo J	User	0	1000	°C	0	MAST	Inactive
		overheat								
3	%IW3.3	Heater_	Thermo K	User	0	1000	°C	0	MAST	Inactive
		overheat								

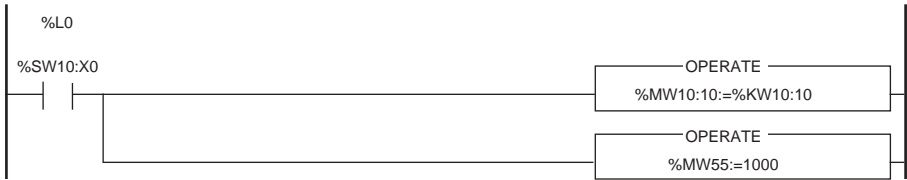
## BIT, WORD AND FUNCTION BLOCK CONFIGURATION

BITS		WORDS		FUNCTION BLOCKS	
Internal (%M)	256	Internal (%MB,%MW,%MD,%MF)	512	Timer(s) series 7 (%T)	0
System (%S)	128	System (%SW,%SD)	128	Timer(s) (%TM)	64
		Common (%NW)	0	Monostable(s) (%MN)	8
		Constant (%KB,%KW,%KD,%KF)	128	Counter(s) (%C)	32
				Register(s) (%R)	4
				Drum(s) (%DR)	8



### MAST-MAIN

(\*Initialization on cold restart of constants -> PID loop buffer and initialization of PWM period at 10 s\*)



List of Variables used in the rung :

ADDRESS	SYMBOL	COMMENT
%SW10:X0		
%KW10		
%MW10	Temp_Setpoint	
%MW55	Period_modul	

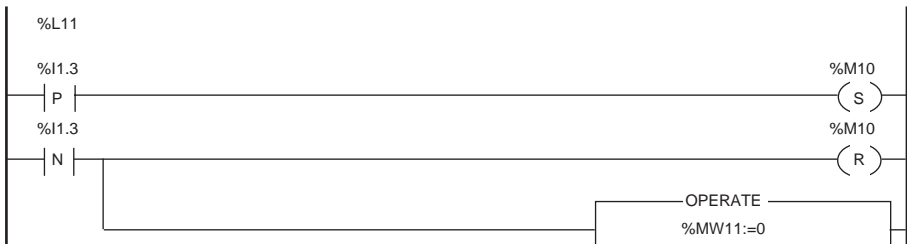
(\*Pump activation\*)



List of Variables used in the rung :

ADDRESS	SYMBOL	COMMENT
%I1.4	Act_pump	
%Q2.5		

(\*Management of PID controller operating mode. This programming allows the CCX17 to modify bit A/M\*)



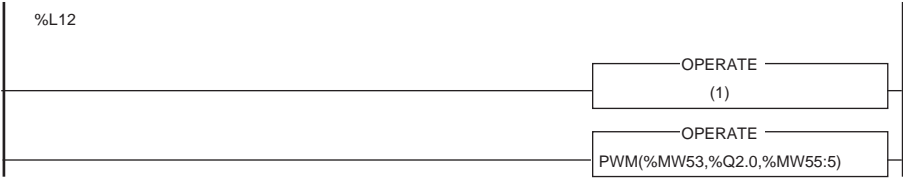
List of Variables used in the rung :

ADDRESS	SYMBOL	COMMENT
%I1.3	Pump_status	
%M10	Auto_manu	
%MW11	PID_output	

(\*Initialization of the water temperature setpoint at 27 °C\*)



(\*Execution of the temperature control loop\*)



List of rung connectors :

(1):PID('TEMP\_WATER','DEGREES',%IW3.0,%MW53,%M10,%MW10:43)

List of Variables used in the rung :

ADDRESS	SYMBOL	COMMENT
%IW3.0	Water_temp	
%MW53		
%M10	Auto_manu	
%MW10	Temp_Setpoint	
%Q2.0	Heater_ctrl	
%MW55	Period_modul	

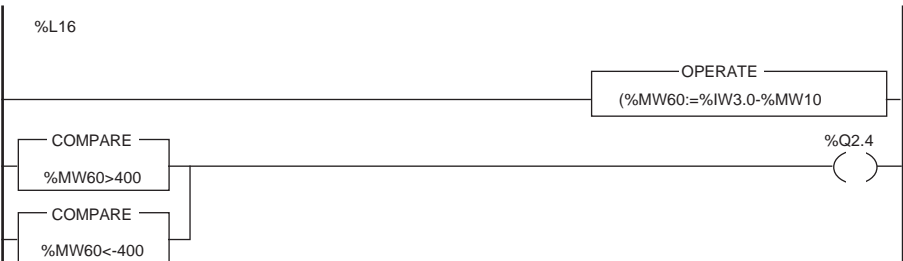
(\*Management of alarms on process value\*)



List of Variables used in the rung :

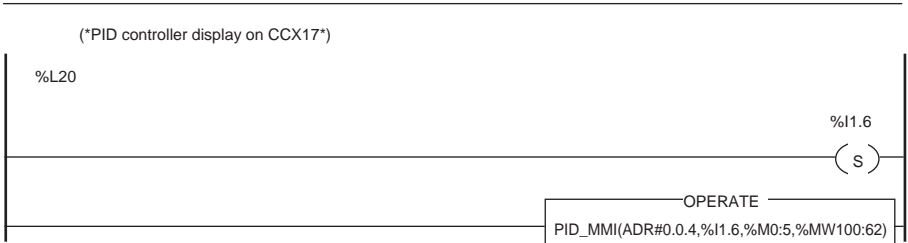
ADDRESS	SYMBOL	COMMENT
%IW3.0	Water_temp	
%Q2.2	High_temp_alarm	
%Q2.3	Low_temp_alarm	

(\*Management of alarms on deviation\*)



List of Variables used in the rung :

ADDRESS	SYMBOL	COMMENT
%IW3.0	Water_temp	
%MW10	Temp_Setpoint	
%MW60		
%Q2.4	Pcl_ft_alarm	



List of Variables used in the rung :

ADDRESS	SYMBOL	COMMENT
%I1.6	Enable_mmi_pcl	
%M0		
%MW100		

**INTERNAL BIT(S)**

ADDRESS	SYMBOL	COMMENT
%M10	Auto_manu	

**INTERNAL WORD(S)**

ADDRESS	SYMBOL	COMMENT
%MW10	PID_buffer	
%MW11	PID_output	
%MW55	Period_modul	

**CONSTANT WORD(S)**

ADDRESS	SYMBOL	VALUE	BASE	COMMENT
%KW0		0	Decimal	
%KW1		0	Decimal	
%KW2		0	Decimal	
%KW3		0	Decimal	
%KW4		0	Decimal	
%KW5		0	Decimal	
%KW6		0	Decimal	
%KW7		0	Decimal	
%KW8		0	Decimal	
%KW9		0	Decimal	
%KW10		0	Decimal	
%KW11	Man_control	0	Decimal	
%KW12	Gain	600	Decimal	
%KW13	T_integral	300	Decimal	
%KW14	T_derivative	50	Decimal	
%KW15	Sample_per	100	Decimal	
%KW16	Control_max	10000	Decimal	
%KW17	Control_min	0	Decimal	
%KW18	PID_config	0	Decimal	
%KW19		?	Double	
%KW20		?	Double	

**DOUBLE(S) MOT(S) CONSTANT(S)**

ADDRESS	SYMBOL	VALUE	BASE	COMMENT
%KD19	High_PV	5000	Double	



## 6.1 Reminder of process control

### 6.1-1 PID parameter adjustment

There are a number of ways to adjust the parameters of a PID control loop. The Ziegler and Nichols method is described here with two variations :

- closed loop adjustment,
- open loop adjustment.

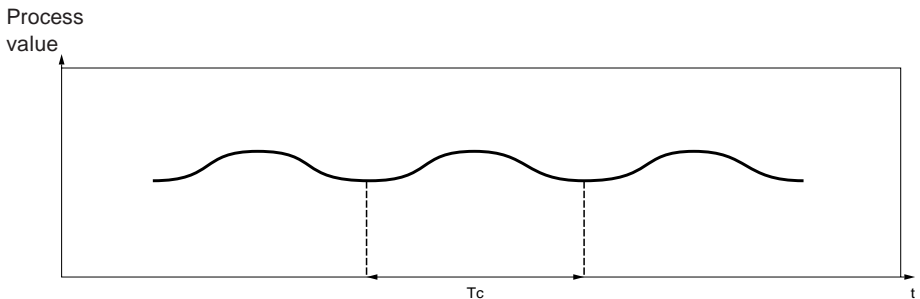
Before using one of these methods, the PID direction of action must be determined :

- If an increase in the OUT output causes an increase in the PV process value, set the PID to indirect mode ( $K_P > 0$ ).
- In the opposite case, if it causes a decrease in the PV, set the PID to direct mode ( $K_P < 0$ ).

### Closed loop adjustment

The principle requires the use of proportional control ( $T_I = 0$ ,  $T_D = 0$ ) to excite the process by increasing the gain until it starts to oscillate after applying a step function to the setpoint used by the PID controller.

Once this is achieved, simply increase the value of the critical gain ( $K_{pc}$ ) which caused the undamped oscillation and the oscillation period ( $T_c$ ) to deduce the values required for an optimal adjustment of the control system.



Depending on the type of process control used (PID or PI), adjustment of the coefficients is carried out with the following values :

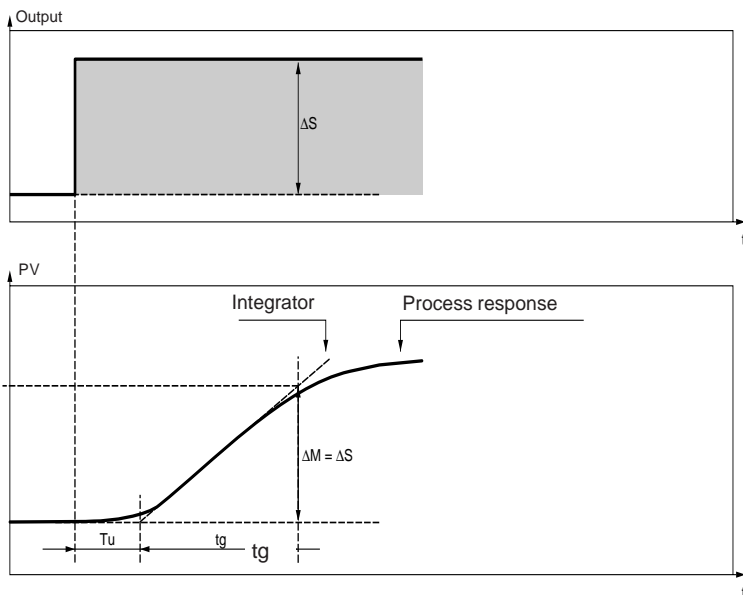
	$K_p$	$T_i$	$T_d$
PID	$\frac{K_{pc}}{1.7}$	$\frac{T_c}{2}$	$\frac{T_c}{8}$
PI	$\frac{K_{pc}}{2.22}$	$0.83 * T_c$	X

where  $K_p$  = proportional gain,  $T_i$  = integral time and  $T_d$  = derivative time.

This type of adjustment produces a highly dynamic control which may result in range overrun when a setpoint is changed. If this occurs, reduce the gain value until the system reacts as required.

### Open loop adjustment

With the process controller in manual mode, apply an increment to the output and consider the initial reaction of the process to that of a pure delay integrator.



The intersection of the line representing the integrator and the time axis determines the time value  $T_u$ .

The time value  $T_g$  is then defined as the time taken by the controlled variable (process value) to vary by the same amplitude (as a % of the full scale range) as the process controller output.

Depending on the type of process controller used (PID or PI), adjustment of the coefficients is carried out with the values opposite.

#### Note :

Care should be taken with units. If the adjustment is made in PL7, multiply the value obtained for  $K_P$  by 100.

	$K_p$	$T_i$	$T_d$
<b>PID</b>	$- 1.2 T_g/T_u$	$\bullet 2 * T_u$	$0.5 * T_u$
<b>PI</b>	$- 0.9 T_g/T_u$	$3.3 * T_u$	X

This type of adjustment also produces a highly dynamic control which may result in range overrun when a setpoint is changed. If this occurs, reduce the gain value until the system reacts as required.

The benefit of using this method is that no assumptions need be made about the type or order of the process to control. It applies equally to stable or true integrator processes. It is especially useful when using slow processes (such as those found in glass manufacturing applications, etc) as the user need only have the start of the answer in order to set the values of coefficients  $K_p$ ,  $T_i$  and  $T_d$ .

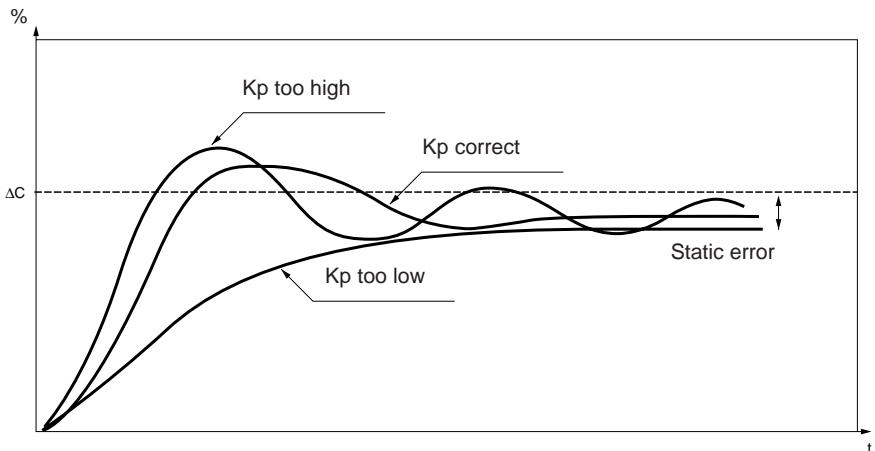
## 6.1-2 PID parameter influence and effects

### Proportional action

Proportional action lets the user affect the response time of the process. The higher the gain, the faster the response time, and the lower the static error (in purely proportional terms), but the more the stability is reduced.

Therefore a compromise must be found between speed and stability.

### Influence of proportional action on process response to a step function

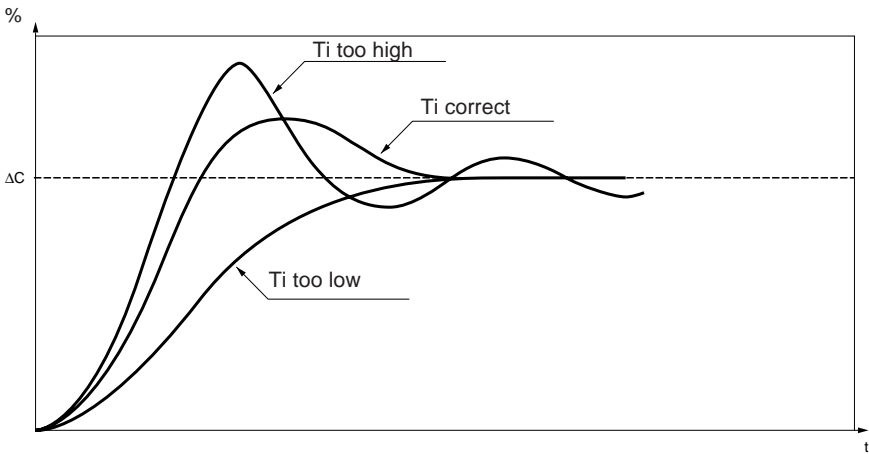


## Integral action

Integral action is used to cancel the static error (difference between the process value and the setpoint). The higher the integral action ( $T_i$  low), the faster the response and the lower the stability.

Therefore a compromise must be found between speed and stability.

## Influence of integral action on process response to a step function



Reminder : a low  $T_i$  implies a higher integral action.

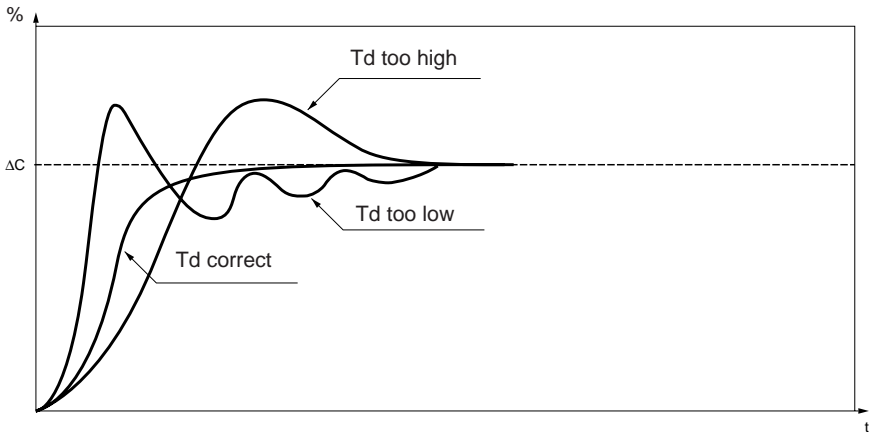


### Derivative action

Derivative action anticipates process response. It adds a term to the equation which takes into account the speed of error variation. This is used to anticipate by accelerating the response of the process when the error increases and slowing the response when the error diminishes. The higher the derivative action ( $T_d$  high), the faster the response.

Again, a compromise must be found between speed and stability.

### Influence of derivative action on process response to a step function



---

## PID control limits

If the process is compared to a first order pure delay of a transfer function :

$$H(p) = \frac{Ke^{-\tau p}}{1 + \theta p}$$

where :

- $\tau$  = model delay,
- $\theta$  = model time constant,

The performance level of the PID control is a function of the ratio  $\tau / \theta$ .

PID control is perfectly suited to the following condition :

$$2 \leq \frac{\tau}{\theta} \leq 20$$

For  $\tau / \theta < 2$ , ie. fast process control loops ( $\theta$  low) or for processes with a long delay ( $\tau$  high), PID control is unsuitable. More sophisticated algorithms are required.

For  $\tau / \theta > 20$ , threshold control plus hysteresis is adequate.

---

## 6.2 Function characteristics

---

### 6.2-1 Memory usage

Function	Generated code volume
PID	2.2 K words
PWM	0.6 K words
SERVO	1.2 K words
PID_MMI	4.4 K words

---

### 6.2-2 Function execution time

Function	Execution time		
	TSX 37	TSX 57-10	TSX 57-20
PID (TI=0 and TD=0)	1.2 ms (1ms without PID_MMI)	1.7ms (1.5ms)	1.1 ms (0.9ms)
PWM	0.6 ms	0.7 ms	0.5 ms
SERVO	0.6 ms	0.8 ms	0.6 ms
PID_MMI (EN=1)	1.3 ms	1.4 ms	1 ms



**A**

Adding a new function call in online mode	2/20
ADDR	3/5
Adjusting a loop	3/4
Adjusting loops	4/1
Alarm management	3/8
AUTO	2/6, 2/9, 4/1
Automatic	2/6

**B**

Bias	4/4
BUTT	3/5

**C**

Call the PID function	2/9
Call the PID_MMI function	3/5
Call the PWM function	2/13
Call the SERVO function	2/19
Cascade	4/5
CCX 17 power break	3/9
Change to RUN	3/9
Change to STOP	3/9
Closed loop adjustment	6/1
Cold restart	2/20, 3/9
Control loop	1/3
Controlling a loop	3/3
Controlling values	3/3

**D**

Derivative	2/6
Derivative action	2/7, 6/5
DEVAL_MMI	2/7, 4/1
Diagnostics	4/3
Disconnection of the CCX 17	3/9
Dn button	3/3
DOWN	2/15

**E**

EN	3/5
Execution time	6/7
Exit button	3/2

**G**

Gain	2/6
------	-----

**H**

HYST	2/15
Hysteresis	2/15

**I**

Inhibit	2/7
INP	2/11, 2/15
Integral	2/6
Integral action	6/4

**K**

KP	2/6, 4/1
----	----------

**L**

LADDER	2/1
Limitations	4/3
LIST	2/2
Lower limit	2/7

**M**

Man-machine interface	3/1
Manual	2/6
Measurement	2/6
Memory usage	6/7
Methodology	1/4
Modulation period	2/11

**N**

Number of PIDs	3/1
----------------	-----

**O**

Open loop adjustment	6/2
Operating modes	2/1
Operating modes of the PLC	3/9
Operator setpoint	2/7
OUT	2/6, 2/9, 4/1
OUT_MAN	2/6, 4/1
OUT_MAX	2/6, 4/1
OUT_MIN	2/7, 4/1

**P**

PARA	2/9, 3/5, 4/1
Parameters of the PID_MMI function	3/5
PID	1/2, 2/4, 2/9, 2/15
PID analog output	2/6
PID function	2/4
PID parameters	2/9
PID programming	2/8
PID_MMI	1/2, 3/5
PID_MMI function	3/5, 4/3
PID_MMI programming	3/6
PLC operating modes	2/20
Position feedback	2/15
Position setpoint	2/15
POT	2/15
Process control functions	4/3
Proportional action	6/3
Pulse width modulation	2/11
PV	2/6, 2/9, 4/1
PV_DEV	2/7, 4/1
PV_HI	2/7
PV_INF	4/1
PV_LO	2/7
PV_MMI	2/7, 4/1
PV_SUP	4/1
PW_O	2/11
PWM	1/2, 2/11, 2/13
PWM function	2/11
PWM programming	2/13

**R**

Reconnecting the CCX 17	3/9
Refresh button	3/2
Rf	3/2

**S**

Sampling period	2/6
Selecting a loop	3/2
Selecting PIDs	3/8
SERVO	1/2, 2/15, 2/19
SERVO function	2/15
SERVO programming	2/18
SERVO with position feedback	2/16
SERVO without position feedback	2/16
Setpoint	2/6

SP	2/6, 4/1
SP_MMI	2/7, 4/1
STRUCTUREDTEXT	2/2

**T**

T_MINI	2/15
T_MOD	2/11
T_MOTOR	2/15
TAG	2/6, 2/9
TD	2/6, 4/1
Temperature control	5/1
TI	2/6, 4/1
TS	2/6, 4/1

**U**

UNIT	2/6, 2/9, 4/1
UP	2/15
Up button	3/3, 3/4
User parameters of the PID function	2/6
User parameters of the PWM function	2/11
User parameters of the SERVO function	2/15

**W**

Warm restart	2/20, 3/9
--------------	-----------

<b>Section</b>	<b>Page</b>
<b>1 Presentation</b>	<b>1/1</b>
1.1 Situation with regard to the control system	1/1
1.2 Characteristics of PMX CPUs	1/3
1.3 User services and functions	1/4
1.3-1 Configuration editor	1/4
1.3-2 Selecting the processor	1/4
1.3-3 Access to parameter entry for the process control application	1/5
1.3-4 Tuning tools in PL7-PRO and PL7-PRO-DYN	1/7
1.3-5 Using control loops with XBT terminals	1/8
1.3-6 Autotuning control loops	1/10
1.3-7 Setpoint programmer	1/11
1.4 Operating modes, software setup	1/12
1.5 Compatibility	1/14
1.6 Processing performance of the PMX	1/14
1.7 Memory occupation	1/14
<b>2 Configuring the process control application</b>	<b>2/1</b>
2.1 Hierarchical structure	2/1
2.2 Configuring each control loop	2/1
2.2-1 Configuring the type of process control	2/2
2.2-2 Description of the control loops	2/4

<b>Section</b>	<b>Page</b>
2.3 Description of the processing branches	2/6
2.3-1 Integrated functions	2/6
2.3-2 Process value processing branch	2/7
2.3-3 Setpoint processing branch	2/9
2.3-4 Feedforward processing branch	2/11
2.3-5 Command and loop controller branch	2/12
2.3-6 Output processing branch	2/17
2.3-7 Summary table	2/20
2.4 The setpoint programmer	2/21
2.5 Global loop parameters	2/28
2.5-1 Execution parameters	2/28
2.5-2 Instrumentation parameters	2/28
2.6 Detailed description of the math and logic functions	2/30
2.6-1 Process value processing functions	2/30
2.6-2 Setpoint branch functions	2/42
2.6-3 Feedforward branch functions	2/52
2.6-4 Loop controller branch functions	2/58
2.6-5 Output branch functions	2/93
2.7 Configuring the I/O	2/103
2.7-1 Assignment	2/103
2.7-2 Associated checks and functions	2/103
2.7-3 Types of interface	2/104
2.8 Symbolization of language objects	2/104
<b>3 Debugging</b>	<b>3/1</b>
3.1 Control loop debug screens	3/1
3.2 Modifying the parameters of each loop	3/2

---



<b>Section</b>	<b>Page</b>	
3.3	Modifying the functions of each loop	3/3
3.4	Debugging the setpoint programmer	3/4
3.5	Optimization of the loop	3/5
3.6	Saving data	3/5
3.6-1	Saving tuning parameters	3/5
3.6-2	Application backup	3/5
<hr/>		
<b>4 Operation</b>	<b>4/1</b>	
<hr/>		
4.1	Configuration under PL7	4/1
4.1-1	Selecting the loops to use	4/1
4.1-2	Exchange zones (%MW)	4/1
4.1-3	Method for configuring the man-machine interface	4/3
4.2	Process control runtime applications on XBT-F terminals	4/4
4.2-1	Applications provided	4/4
4.2-2	Runtime page formats	4/5
4.2-3	Moving around the various views	4/7
4.2-4	Method for loading XBT-F applications	4/8
4.3	XBT-F01 process control runtime screens	4/9
4.3-1	Monitoring screen	4/9
4.3-2	Front panel screen	4/10
4.3-3	Dynamic trending screen	4/11
4.3-4	Parameter tuning screen	4/12
4.3-5	Autotuning screen	4/13
4.3-6	Screen for selecting and tuning the setpoint profile	4/14
4.3-7	Using the alarm pages	4/14

<b>Section</b>	<b>Page</b>
4.4 XBT-F02 and TXBT-F02 process control runtime screens	4/15
4.4-1 Monitoring screen	4/15
4.4-2 Supervisory control screen	4/16
4.4-3 Tuning screen	4/17
4.4-4 Screen for selecting and tuning the setpoint profile	4/18
4.4-5 Using the alarm pages	4/18
4.4-6 Error messages	4/18
<hr/> <b>5 Operating modes</b>	<hr/> <b>5/1</b>
5.1 Executing the control channels	5/1
5.1-1 Distribution of the process control processing	5/1
5.1-2 Synchronizing the pre- and post-processing	5/1
5.1-3 Multitask applications	5/2
5.2 PLC operating modes	5/2
5.2-1 On PLC power-up	5/3
5.2-2 CPU in RUN mode	5/3
5.2-3 On a CPU change from RUN to STOP	5/3
5.2-4 On a cold restart	5/4
5.2-5 On a warm restart	5/4
5.3 Control loop operating modes	5/4
5.3-1 Manual control	5/4
5.3-2 Automatic execution	5/5
5.3-3 Starting an autotuning operation	5/5
5.3-4 Execution in tracking mode	5/6
5.3-5 Auto/manu changeover	5/6
5.3-6 Manu/Auto changeover (apart from ONOFF loop controller)	5/6
5.3-7 Behavior of the loops on an I/O fault	5/7
5.4 Process loop operating mode	5/8
5.5 Single loop operating mode (3 single loops)	5/9
5.6 Cascaded loop operating mode	5/10

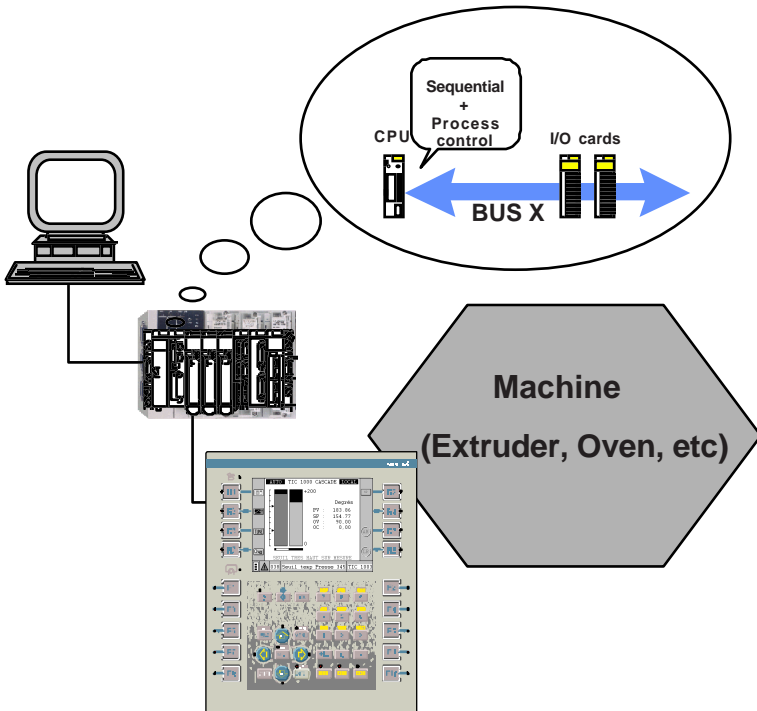
---

<b>Section</b>	<b>Page</b>
5.7 Autoselective loop operating modes	5/12
<hr/>	
<b>6 Process control language objects</b>	<b>6/1</b>
<hr/>	
6.1 Object language addressing	6/1
<hr/>	
6.2 Language objects associated with process control channels	6/1
6.2-1 Sending commands	6/1
6.2-2 Command parameter values (%MDxy.i.j)	6/2
6.2-3 Control loop word command values (%MWxy.i.11)	6/3
6.2-4 Setpoint programmer command values (%MWxy.i.7)	6/4
<hr/>	
6.3 Language objects associated with the process loop channel	6/5
6.3-1 Configuration language objects	6/5
6.3-2 Default and diagnostic language objects	6/9
6.3-3 Process control language objects	6/14
<hr/>	
6.4 Language objects associated with a 3 single loop channel	6/19
6.4-1 Configuration language objects	6/19
6.4-2 Diagnostic and default language objects	6/23
6.4-3 Process control language objects	6/31
<hr/>	
6.5 Language objects associated with a cascaded loop channel	6/37
6.5-1 Configuration language objects	6/37
6.5-2 Default and diagnostic language objects	6/43
6.5-3 Process control language objects	6/49
<hr/>	
6.6 Language objects associated with the autoselective loop channel	6/55
6.6-1 Configuration language objects	6/55
6.6-2 Fault and diagnostics language objects	6/61
6.6-3 Process control language objects	6/68
<hr/>	
6.7 Language objects associated with the setpoint programmer	6/74
6.7-1 Configuration language objects	6/74
6.7-2 Fault and diagnostic language objects	6/81
6.7-3 Process control language objects	6/84

---

<b>Section</b>	<b>Page</b>	
<b>6.8</b>	<b>Tables of exchanges for operation</b>	<b>6/90</b>
6.8-1	Table of multiplexed parameters for a loop	6/90
6.8-2	Table of periodic data	6/95
6.8-3	Table of alarms (loop only)	6/95
6.8-4	XBT special table	6/96
6.8-5	Default addresses	6/98
<hr/>		
<b>7</b>	<b>Appendix</b>	<b>7/1</b>
<hr/>		
<b>7.1</b>	<b>Debugging the feedforward</b>	<b>7/1</b>
7.1-1	Adjusting the gain	7/1
7.1-2	Adjusting the lead-lag	7/1
<hr/>		
<b>7.2</b>	<b>Debugging the PID function</b>	<b>7/4</b>
7.2-1	Closed loop adjustment	7/4
7.2-2	Open loop adjustment	7/5
7.2-3	Roles and effects of the parameters of a PID controller	7/6
<hr/>		
<b>7.3</b>	<b>Debugging the model-based controller</b>	<b>7/9</b>
7.3-1	Adjusting the static gain (Ks)	7/9
7.3-2	Adjusting the dead time or delay (T_DELAY)	7/10
7.3-3	Adjusting the time constant	7/11
<hr/>		
<b>8</b>	<b>Index</b>	<b>8/1</b>

## 1.1 Situation with regard to the control system



PMX57 process control processors are identical to TSX57 CPUs and have, in addition to their two communication channels, **ten control channels known as "loop controllers"**. These channels can be optionally configured to execute the control algorithms of industrial processes. To do this, the following types of processing :

- process loop,
- cascaded loop,
- autoselective loop (also called "secondary loop"),
- single process control loop,
- setpoint programmer,

are available and can be downloaded with the PL7 application.

As with any PLC processor, PMX57 PLCs manage an entire station consisting of discrete and analog I/O modules, and application-specific modules distributed over one or more racks connected on Bus X or a remote bus. The I/O interfaces necessary for process control are the standard channels of the PLC discrete or analog modules.

PMX57 process control processors have processing tasks called Mast and Fast, as well as 64 event-triggered tasks. The user must therefore assign the I/O interfaces and processing of the process control loops to these processing tasks.

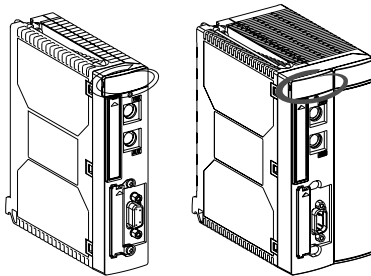
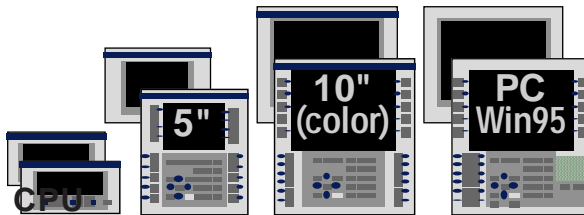
The parameters of the process control loops are entered during the configuration of the CPU, using PL7 Junior or PL7 Pro, via the application-specific screen. Debugging screens are available to the user in online mode. Programming of the control system application follows the same rules as those used for TSX57 CPUs, with all the PL7 tools available to the programmer.

XBT F and TXBT Windows man-machine interfaces provide application-specific process control screens, as well as implicit navigation between these screens to assist with operation.

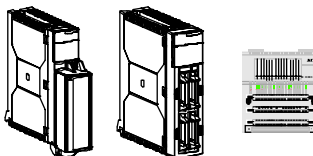
These screens are integrated in the global runtime application. They do not require any special programming.

From an XBTL1000 application which integrates the process control part, the user can add to his runtime application using XBTL1000 software, the programming tool for the XBT range.

## MMI

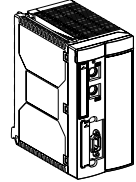
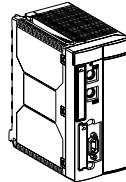
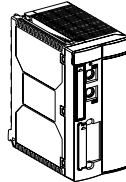
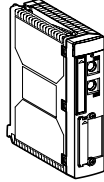
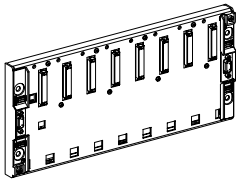


I/O



## 1.2 Characteristics of PMX CPUs

Type of PMX processor which can be integrated in TSXRKY ●●racks



<b>Station characteristics</b> (1) <i>TSX RKY..EX racks</i>	2		8	
<i>Module positions</i> (2)	21		87	
<i>I/O profile</i>	Fixed			Flexible
<i>No. of discrete I/O</i> (3)	512	1024	2048 maximum	
<i>No. of analog I/O</i>	24	80	128	256 maximum
<i>No. of expert channels</i>	8	24	32	64 maximum
<i>No. of control channels</i>	10			
<i>Process control functions</i>	Process loop (*)		Process loop Setpoint programmer 3 single loops Cascaded loop Autoselective loop	
<i>Network connection</i>	1 (FIPWAY,ETHWAY/TCP_IP,Modbus +)		3	4 (FIPWAY,ETHWAY/TCP_IP,Modbus+)
<i>FIPIO master connection</i>	1 (integrated)			
<i>Connection to other manufacturer's fieldbuses</i>	1		2	
<i>Connection to AS-i sensor/actuator bus</i>	2	4	8	
<b>Memory characteristics</b>				
<i>Internal memory</i>	32 K16	48 K16	80 K16	112 K16
<i>Memory extension</i>	64 K16	128 K16	256 K16	
<b>References</b>	<b>TPMX P57 102</b>	<b>TPMX P57 202</b>	<b>TPMX P57 352</b>	<b>T PMX P57 52</b>

(\*) Without model-based controller

(1) Maximum characteristics of the station managed by the processor.

(2) For standard format modules, excluding power supply modules and processor

(3) Fixed I/O profile: the number of discrete I/O, analog and application-specific channels can be summed.

Flexible I/O profile: the number of discrete I/O, analog and application-specific channels cannot be summed, the distribution is defined by a formula.

C

---

## 1.3 User services and functions

---

### 1.3-1 Configuration editor

The generic software setup of all the application-specific functions is described in the "Basic functions" manual, in the PL7 Junior or Pro documentation. For more detailed information, please refer to this manual.

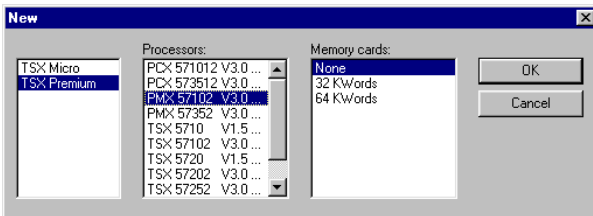
---

### 1.3-2 Selecting the processor

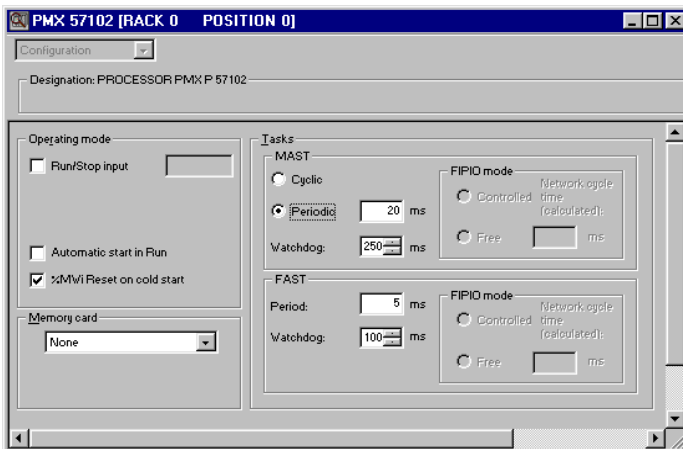
The processor is selected by accessing the selection list, when an application is created. It is also given at the top left hand side of the configuration screen.

The validation of this selection is displayed by the corresponding graphic representation.

The associated configuration is updated automatically.



The characteristics defined in the configuration menu of the PMX processors (name of the application, task scans, etc) are identical to those of TSX57 CPUs. Please refer to the user manual for these modules.

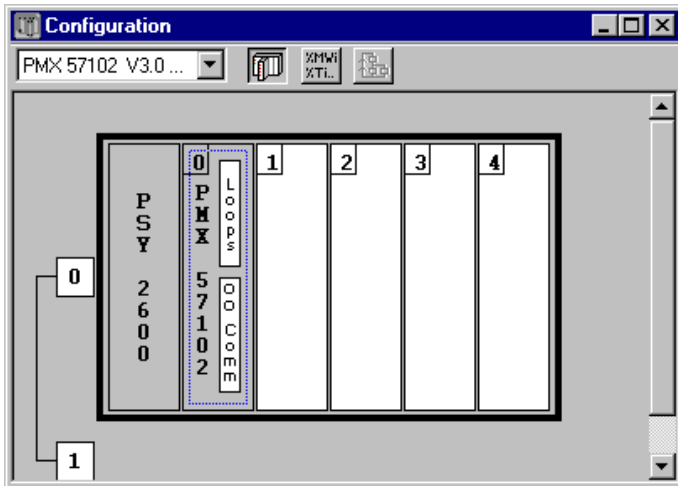


Note : The control loops are processed in periodic tasks.



### 1.3-3 Access to parameter entry for the process control application

The software setup of the control channels is identical to that used for any PL7 application-specific function. It is performed using an interface integrated in the processor. From position 0 or 1 in the configuration editor, the screens for the various channels of the CPU are accessed by double-clicking on the “LOOPS” interface.



Two types of screen are used for the software setup of an application-specific function. They are called :

- Configuration screens (offline and online mode)
- Debug / tuning screen (online mode only).

The configuration screens are the initial entry point to the process control application. They are used to define the use of the 10 control channels.

These application-specific screens are divided into three zones :

Zone A : Module zone.

Zone B : The channel zone then characterizes the strategy adopted for each loop controller. The designer can choose between :

- A process loop,
- A cascaded loop,
- An autoselective loop,
- 3 single loops,
- A setpoint programmer.

These choices cannot be modified in online mode.

**Zone C :** Each loop can then be configured. The user determines the various calculation functions defined in each processing branch. To do this, he adds to his algorithm or reduces it by selecting/deselecting functions.

The screenshot shows the configuration window for 'Loop Controller 4 - LOOP0'. The 'Functions' table in Zone B is as follows:

Loop	Functions	Parameters
Loop parameters	Instrumentation	Loop name: LOOP0
Measurement: Standard	Execution	Unit:
Setpoint: Simple		Low scale (phg): 0.0
Loop Controller: PID		High scale (phg): 100.0
Feed Forward: No		
Output 1: Analog		

The schematic in Zone C shows a control loop with inputs PV and SP1, a controller block, and outputs OUT1 and OUT2.

The loop configuration can be modified in online or offline mode. It cannot be modified by the program.

The debug screen in online mode provides the following services :

- Simulation of the input values (measurement, feedforward)
- Animation of the schematic,
- Modification of the math and logic function tuning parameters
- Modification and saving of all parameters
- Sending autotune, manu etc, commands.

The debug screen shows the 'Loop Controller 4 - LOOP0' configuration. The 'Functions' table in Zone B is as follows:

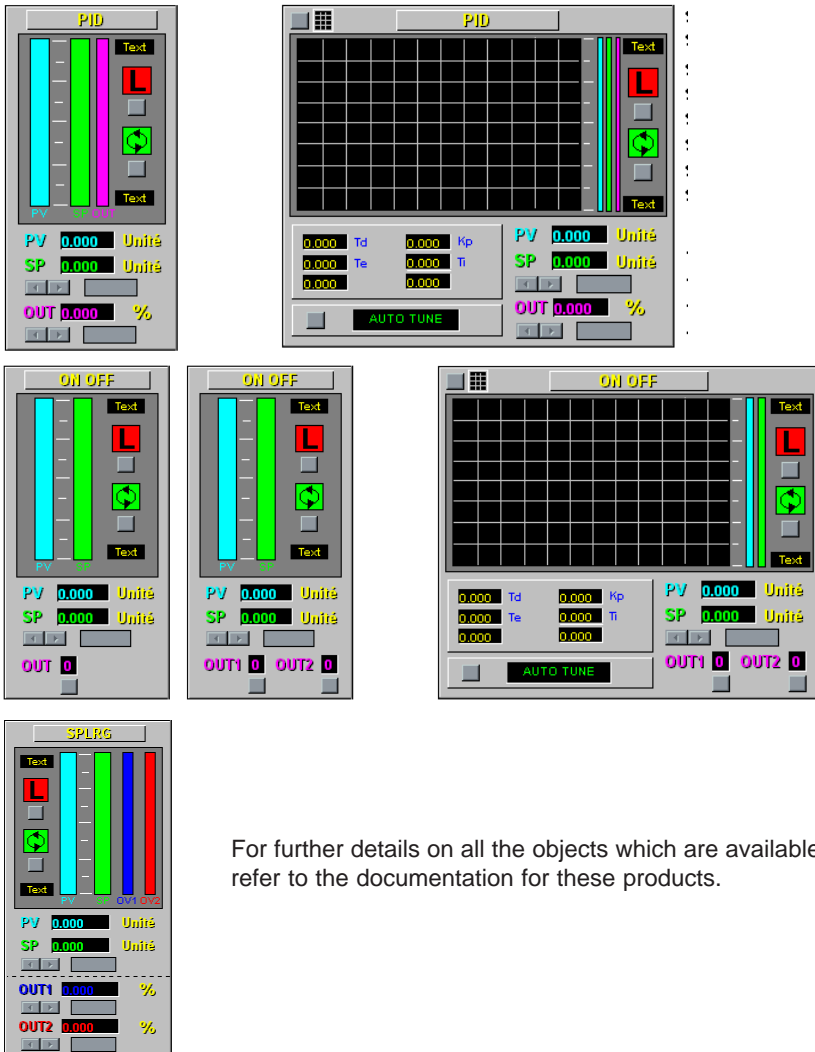
Loop	Functions	Parameters
Loop parameters	Format	Time constant (s): 10.0
Measurement: Standard	Filtering	Gain: 1.0
Setpoint: Simple	Fct Generator	Output: 4036.995
Loop Controller: Hot/Cold	Alarms	
Feed Forward: Yes	Simulation	
Output 1: Analog		

The schematic in Zone C shows a control loop with inputs PV (5000), SP1 (630), and FF (3745). The output OUT1 is 15.608 and OUT2 is 0.0. The current output value is 140.37.



### 1.3-4 Tuning tools in PL7-PRO and PL7-PRO-DYN

Bargraphs, trend charts and customized runtime pages can be used for tuning and using control loops. These functions can be be set up using PL7-PRO-DYN runtime software or with the "Runtime Screens" tool integrated in the PL7-PRO software workshop. Their object libraries contain front panel views and trend page views, making it easy to animate the process control application.



For further details on all the objects which are available, please refer to the documentation for these products.

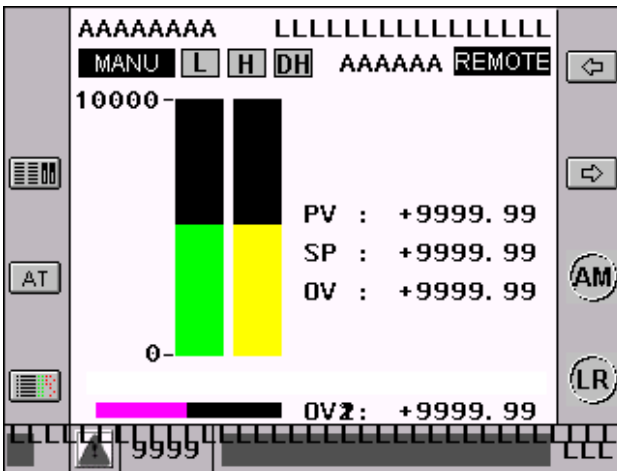
### 1.3-5 Using control loops with XBT terminals

Control loops can be used with the application provided for certain Magelis XBT-F terminals, **in point-to-point mode**. If this method is used, and to make the programming of the communication transparent, the interface uses memory zone %MW3200 to %MW3235 and %MW3350 to %MW4090 by default to perform exchanges with the PLC (see section 4.1).

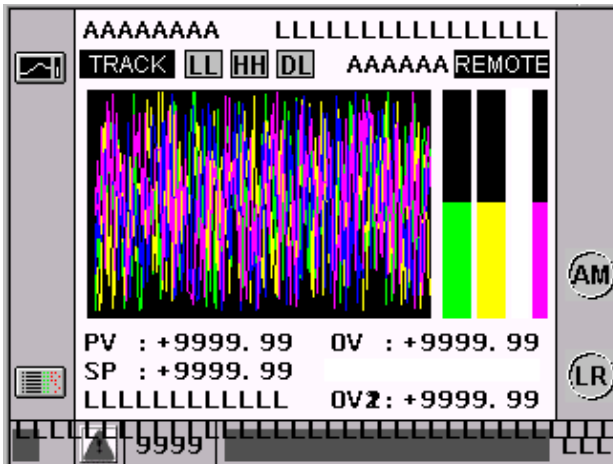
In addition to the services offered by XBTL1000 and the Magelis terminals, XBT-F man-machine interfaces provide the following, for controlling **each** control loop :

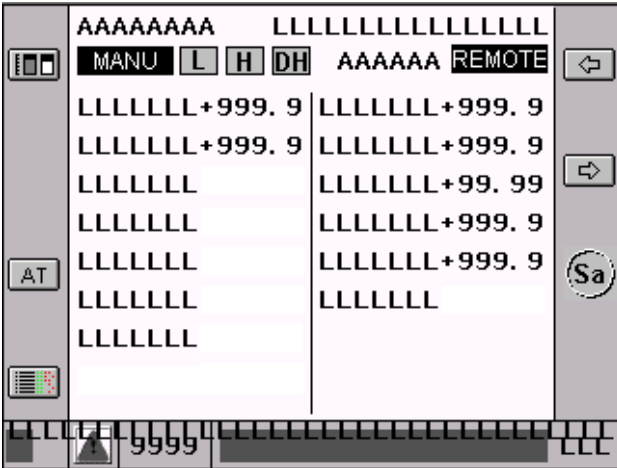
A "front panel" page, a "supervisory control" page, a "tuning" page.

These pages are identical for all control loops.

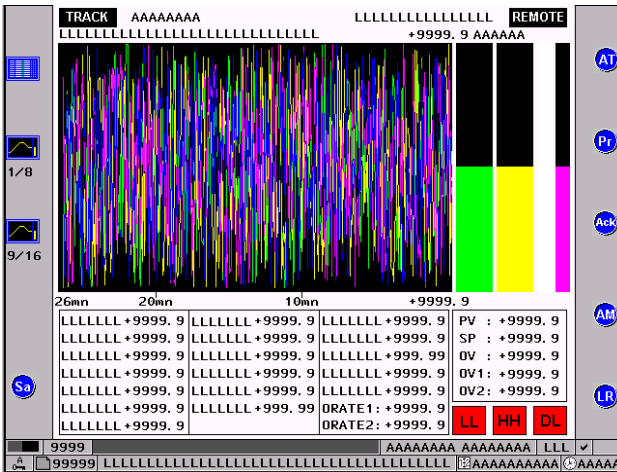


The control functions, as well as manual mode, automatic mode, autotuning, etc, can all be performed from these runtime screens.

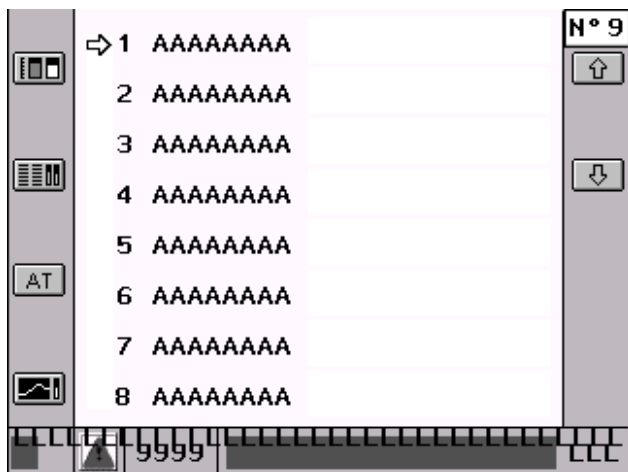




XBT-F02 man-machine interfaces offer a runtime page and a complete tuning page for each control loop.



- To monitor the whole process :
  - A monitoring page : This page brings together the main data on all the control loops used. It is the entry point for running the process control application. It is also used for managing access to the runtime pages for a specific loop.
  - Alarm pages associated with each loop (HH,H,L,LL, deviation) : These pages are integrated in the alarm supervisors of the XBT terminals.



### 1.3-6 Autotuning control loops

Autotuning is applied to most processes, such as the control of temperature, flow rate, pressure, etc.

The controllers integrated in these control loops calculate a set of tuning parameters ( $K_p, T_i, T_d$ ) when autotuning is requested.

These parameters can be accessed from the debug screens in PL7 and from a specific tuning screen in the XBT terminals.

These variables can also be used in PL7 animation tables.



### 1.3-7 Setpoint programmer

All the control channels can be configured as setpoint programmers. In this case they each define a maximum of 48 segments. These segments can be divided into a maximum of 6 profiles. These profiles can be assigned to one or more control loops. They are assigned in the configuration screen of the control loop.

In this case, when the programmer is interfaced with a single loop, the process value (PV) tracking function can then be used. One guaranteed dwell time function can be used.

PMX 57352 [RACK 0 POSITION 0]

Configuration

Designation: PROCESSEUR PMX P 57352

Symbol:

Loop Controller:  Function:  Task:  Distribution of segments:

Loop Controller 5 - SPP\_1  Prog. de consigne  MAST  8-8-8-8-8

Name

PROFILE\_1 PROFILE\_2 PROFILE\_3 PROFILE\_4 PROFILE\_5 PROFILE\_6

Segments Execution Number of segments:  on

Plateau ensured

Segment #	SP #	VAL #	Unité	Pq	S0	S1	S2	S3	S4	S5	S6	S7
1 Ramp	0.0	0.0	Seconds	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2 Plateau	0.0	0.0	Seconds	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3 Ramp	0.0	0.0	Seconds	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4 Plateau	0.0	0.0	Seconds	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

OFFLINE U:SYS MODIF. OVR

On the XBT, a tuning screen and a runtime screen specific to the setpoint programmer are used for continuous tuning of the required setpoint profiles.

## 1.4 Operating modes, software setup

Setup of the process control application comprises a number of steps which can be divided up as follows.

### PL7 in offline mode

Configura-  
tion of the  
CPU

Configura-  
tion of the  
control chan-  
nels

### PL7 in offline or online mode

Symboliza-  
tion

Program-  
ming

Transfer

### PL7 in online mode

Debug the  
loops

Tune the loop  
parameters

### XBT

Configuration of the PMX57 processor :

- Hardware configuration of the application
- Software configuration of the application

Configuration of the 10 control channels :

- Choice of the type of loop
- Choice of the math and logic functions
- Entry of the configuration parameters
- Etc

### XBT

Symbolization if necessary (optional) of the associated variables, using the data editor.

Programming of the PLC and XBT applications using PL7 and XBTL1000 (sequential part).

Transfer of the program and each application-specific function (I/O, process control, communication, etc) to the PLC for debugging, and transfer to the XBTs.

Program-  
ming

Transfer

### XBT

Debug the  
application

### XBT

The configuration editor then offers :

- debug screens for modifying the values of the tuning parameters and saving them
- diagnostics screens for identifying faults on modules.



**PL7 in offline or  
online mode**

Documentation

**XBT**

**XBT  
(Operation)**

Operation

Printing out of the application documentation containing information concerning the various application-specific functions including those for the process control application :

- Config parameters
- Tuning parameters
- Etc

On the XBT, the predefined process control screens are used to control the process loops of the machine.

The PL7 application can always be altered at a later date by adding, removing or modifying the control channels. The XBT application automatically recognizes the changes.

Setup remains identical to that described above.

---

## 1.5 Compatibility

---

There is upwards compatibility between TSX and PMX processors.

Example :

A PMX57-20 CPU can execute any application designed for TSX57-10 and TSX57-20 processors.

However, a PMX57 application, even if it does not contain any control loops, cannot be downloaded to a TSX processor.

---

## 1.6 Processing performance of the PMX

---

### Processing times for the algorithms

The control algorithms use single precision floating point format calculation (IEEE format).

The processing performances are summarized in the following table. The times are given in ms, for one loop.

	Single loop	Process loop	Cascaded loop	Autoselective loop	Setpoint programmer
PMX57-10		3.0 to 6.5			
PMX57-20 PMX57-35 PMX57-45	0.5	(*) 0.5 to 1.0	(*) 1.0 to 2.0	(*) 1.0 to 2.0	

(\*) x to y :    x = minimum loop profile  
                  y = maximum loop profile

---

## 1.7 Memory occupation

---

The code is only loaded once for each type of loop. However, the volume of data depends on the number of loops.

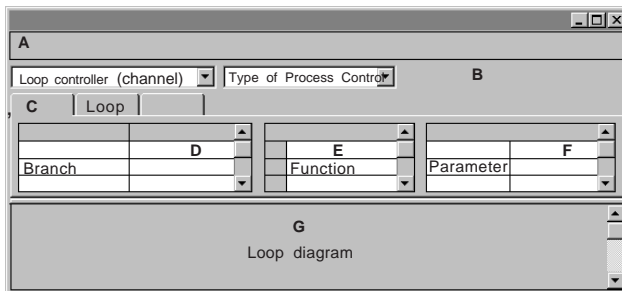
Note : Only channels which are defined occupy memory in the processor.

## 2.1 Hierarchical structure

The PMX **module** has 10 integrated control channels (known generically as **Loop controllers**). Each loop controller can be made up of several **control loops** (eg : master loop, slave loop)

A control loop is, itself, made up of **Branches** or Blocks (Process Value processing branch, Setpoint processing branch, etc). These branches integrate calculation **Functions** (gain, filtering, square root). They are each described by a certain number of **Parameters**.

The configuration of the process control application in PL7 is based on this hierarchical structure represented from top to bottom by objects (zones), ie :



A-> Module zone (CPU),  
 B-> Channel zone,  
 C-> Loops tab,  
 D-> Branches grid,  
 E-> Functions grid,  
 F->Parameters grid,  
 G-> Overall graphic.

## 2.2 Configuring each control loop

The following must be performed for each of the control channels :

1. Define the control structure (single loop, process loop, cascaded loop, autoselective loop).
2. Define the algorithm for the various processing branches (process value, setpoint, loop controller, etc).
3. Choose the functions and configuration parameters for these processing branches.
4. Enter the input and output interfaces.
5. Set the initial values of the tuning parameters.
6. Symbolize the language objects associated with these channels (optional).
7. Configure the exchanges for level 2 (optional).
8. Validate the overall configuration.

These steps are described in the following sections.

Note : Only channels which are defined occupy memory in the processor.

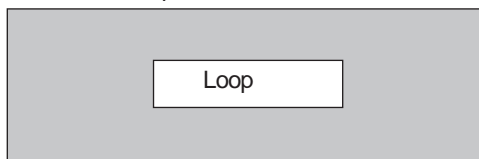
## 2.2-1 Configuring the type of process control

For a control channel, one strategy can be selected from the 5 following predefined profiles :

- 1 process loop,
- 3 single loops,
- 1 cascaded loop,
- 1 autoselective loop,
- 1 setpoint programmer.

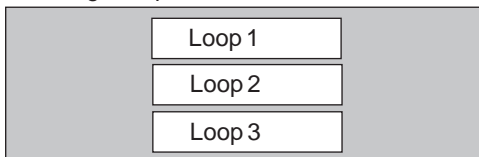
The **process** loop  
(loop with a single loop controller).

Process loop



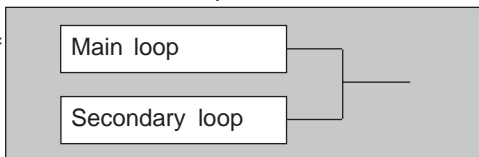
The loop controller with **3 single loops**, consisting of 3 single loops, is provided for increasing the capacity of the number of loops.

3 single loops



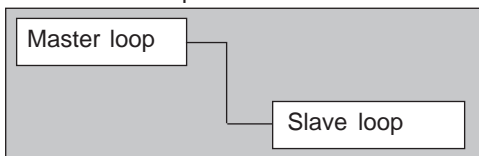
The **autoselective loop**, also called the **secondary loop**, consists of 2 parallel loops with an output selection algorithm. The secondary loop is a single type loop.

Autoselective loop

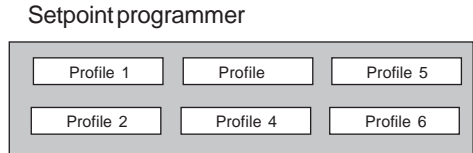


The **cascaded** loop consists of 2 dependent loops (the output of the master loop is the setpoint for the slave loop).

Cascaded loop



The **setpoint programmer** comprises a maximum of 6 profiles, which are themselves made up of 48 segments which are executed exclusively.



The 10 channels are independent in terms of choice. They can have, for example :

- 30 single loops,
- or
- 5 setpoint programmers, each associated with 5 control loops,
- or
- 10 cascaded loops,
- or
- 2 setpoint programmers and 8 process loops,
- or
- etc.

### Predefined strategy

Each strategy has default parameters, except for the setpoint programmer. The use of the various functions integrated in the algorithms (square root, function generator, etc) is predefined, as is the initial value of each parameter.

These channels are assigned by default to the MAST task. It is possible to change this assignment for higher priority processing (Fast task).

The **loop sampling period** is preset at 300 ms. This defines the processing period for the loop controller in automatic mode.

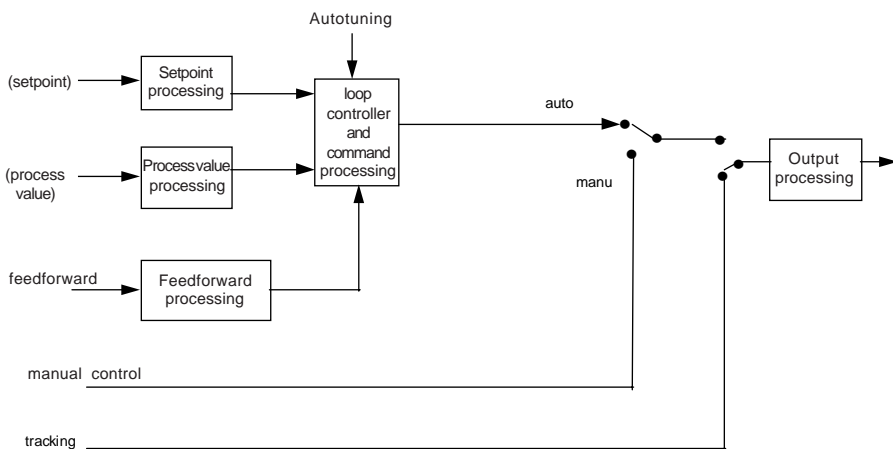
## 2.2-2 Description of the control loops

5 processing branches are used in the control loop algorithms :

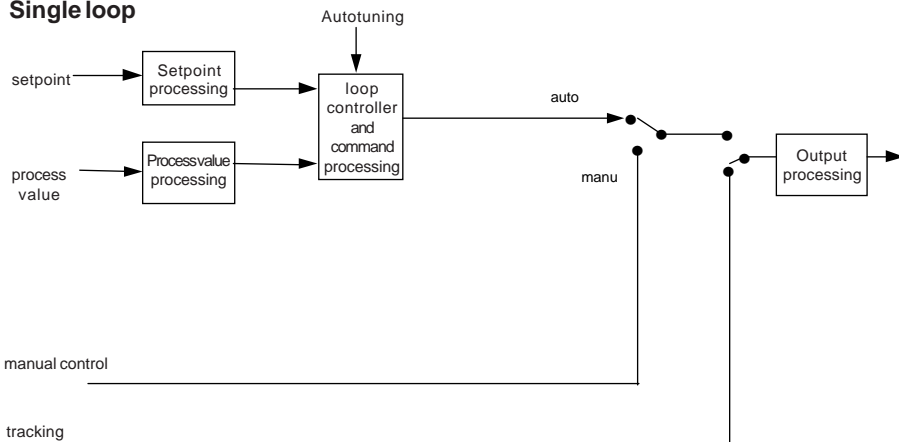
- The process value processing branch,
- The Feedforward processing branch,
- The setpoint processing branch,
- The loop controller branch,
- The output(s) processing branch.

The predefined algorithms for the 4 types of process control can be represented in the following way :

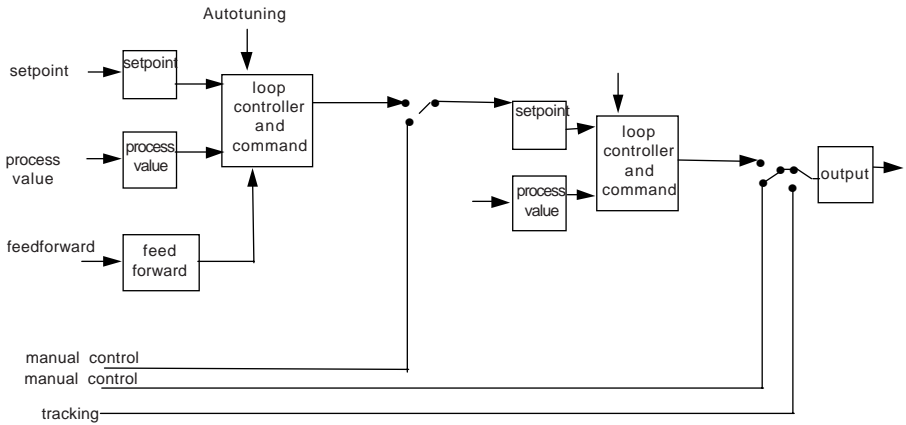
### Process loop



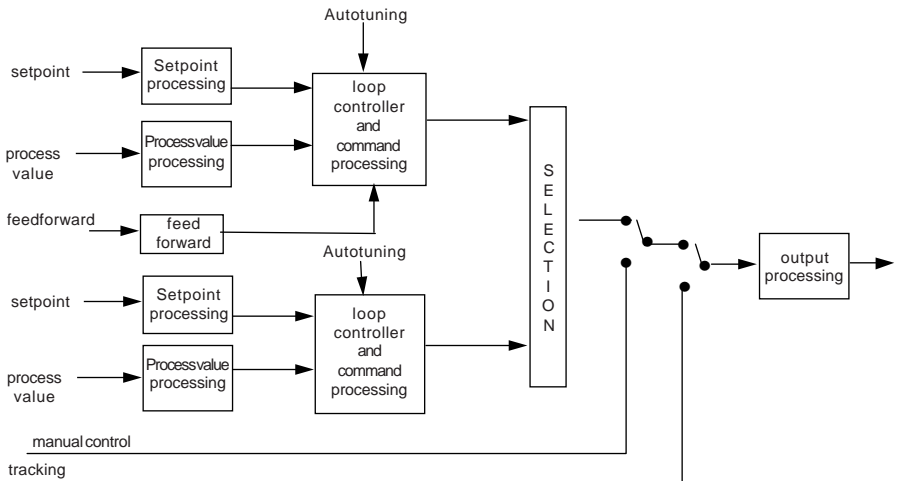
### Single loop



**Cascaded loop**



**Autoselective loop**



The operation of each processing branch (process value, setpoint, etc) is identical, whatever type of process control is chosen (cascaded, single, etc). The behavior of these branches is described in the following specific sections.

## 2.3 Description of the processing branches

### 2.3-1 Integrated functions

Each of the processing branches integrates math and logic functions :

- Process value branch



1st order filtering



Square root



Function generator



Threshold limiter



Alarms on level



Totalizing



Scaling

- Setpoint branch



Selection



Ratio



Setpoint programmer



Tracking the PV



Speed limiter



Scaling



Threshold limiter

- Loop controllers



OnOff 2 states



OnOff 3 states



PID



Internal model-based controller



Hot/Cool



Split range

- Feedforward branch



Scaling



Lead-lag

- Output branch



Scaling



Servo servomotor



Analog output



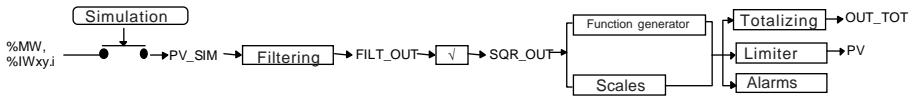
Pulsed output



### 2.3-2 Process value processing branch

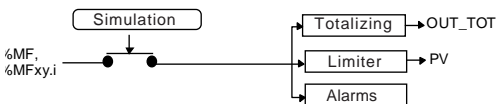
There are 2 types of process value : the standard process value, and the external process value, for which the schematics are as follows :

#### Standard process value



- The first order filtering function has a gain coefficient,
- There are four thresholds for the alarm block on the process value with a hysteresis whose value is 1% of the full scale,
- Two input formats can be used (unipolar or bipolar),
- There is a smooth transition to simulated mode, and the initial simulation value used is the last process value read,
- The function generator includes scaling,
- The value of the process value can be restricted to the scale limits.

#### External process value



The external process value is used to obtain, at the loop controller input, a process value, PV, which was processed outside the control loop.

This solution is provided for cases in which calculation of the process value requires specific or customized functions not offered in the processing of standard process values.

#### Initialization

On startup, the associated data is first updated before running the first processing operation of this branch.

If the process value input address is not entered, processing is performed on the simulated value initially set to zero.

On initialization, the consistency of the configuration which has been entered is checked. If the configuration is incorrect, the loop remains in an initialization state.

---

### Execution check

There are 2 types of fault :

- serious faults,
- warnings.

Two serious faults are tested for during processing of the process value :

- Parameter error (it is not written in floating point format).
- Internal calculation error (division by zero, overflow, etc).

If a serious error is detected, processing of the loop changes to fallback state and the value of the calculated PV is frozen. The outputs of the control loop are frozen.

When the error disappears, the control loop returns to normal state. The loop restarts in the preceding mode without jerks at the outputs.

During a cold start, if a serious error occurs during processing of the process value or if the scale values are incorrect (non-floating point value, or lower limit greater than upper limit), the loop remains in its initialization position and it does not start. The loop starts when the error disappears.

During normal operation, if there is an error on the scale values, the process value is processed with the old correct scale values which are replaced in the scale parameters. The scale parameters are updated when the result of the check is correct.

### 2.3-3 Setpoint processing branch

Four types of setpoint can be used on a process loop, the master loop of a cascade or the main loop of an autoselective loop. These are :

- a ratio setpoint
- a selection setpoint
- a simple setpoint (remote with scaling)
- a setpoint programmer

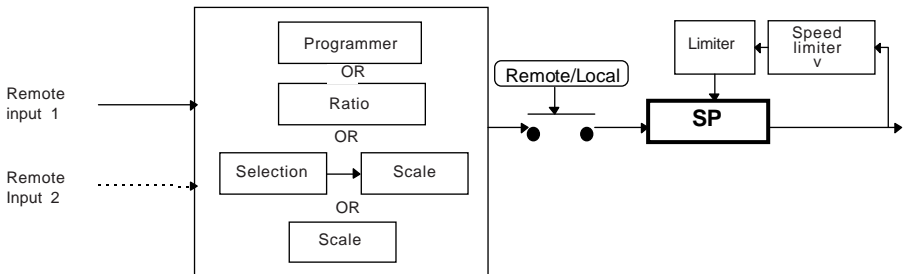
In the case of 3 single loops or the secondary loop in an autoselective loop, only the simple setpoint and the setpoint from a setpoint programmer can be used.

#### Definition

Local setpoint : setpoint written by a man-machine interface

Remote setpoint : setpoint produced by a processing operation.

#### General schematic



Generally :

- The Local value tracks the remote setpoint value in order to ensure a smooth transition when changing operating mode.
- If the address of the Remote setpoint is not filled in, offline mode is “ forced ”.
- To avoid sudden changes, the speed of the setpoint can be limited.
- By default, the setpoint is limited to the scale of the loop. A more restrictive limit can be set.
- When the loop controller is in manual mode, the setpoint can track the process value.

#### Initialization

On initialization (cold start), the consistency of the configuration which has been entered is checked. If the configuration is incorrect, the loop remains in an initialization state and the error is indicated in the status words.

---

On a cold start, if the input address of the setpoint is not entered, the setpoint remains on the local setpoint, whose initial value parameters can be entered (0.0 by default). It is not possible to change the setpoint to remote mode. The behavior of the selection setpoints, with two input addresses, is identical. The remote1/remote2 and R/L commands are not accepted if the addresses are not entered.

The initial state of the Remote or Local and R1 or R2 setpoints can be configured by entering the parameters in the configuration screen or by sending commands from a sequential program.

#### Execution check

An error occurs during processing of the setpoint if :

- the value of a parameter is not written in floating point format,
- an internal calculation is not performed correctly (division by zero, overflow, etc).

In this case, the result of processing of the setpoint, SP, is frozen.

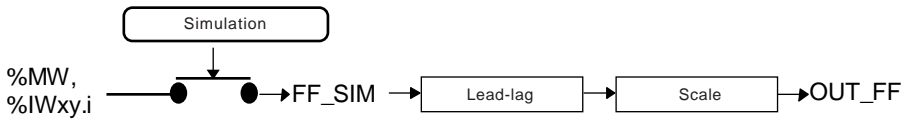
Warnings are displayed. These errors are not considered serious in terms of the control loop, and the loop controller and the output values continue to be calculated with the frozen setpoint value.

Calculation of the setpoint, SP, restarts as soon as the fault disappears.

Other warnings associated with the functions integrated in the setpoint branch are also displayed. Details of these are given in the description of each function (WARN button in the channel zone).

### 2.3-4 Feedforward processing branch

#### General schematic



#### Feedforward action to compensate for disturbance

In standard PID control applications, the loop controller reacts to variations in the process output (closed loop control). As a result, if disturbance occurs, the loop controller only starts to react when the process value deviates from the setpoint. The Feedforward function is used to compensate for measurable disturbance as soon as it appears. This function, in open loop mode, anticipates the effect of the disturbance : it is thus referred to as anticipative (or Feedforward) action.

#### Initialization

If the address of the Feedforward branch is not entered, processing is performed from the simulated value initially set to zero.

#### Execution check

An error occurs during processing of the Feedforward if :

- the value of a parameter is not written in floating point format,
- an internal calculation is not performed correctly (division by zero, overflow, etc).

In this case, the result of processing of this branch is frozen.

These errors are not considered serious in terms of the control loop, and the loop controller continues to calculate the output values with the frozen Feedforward value. Specific warnings are displayed.

The value `OUTFF` at the loop controller input is updated as soon as the fault disappears. Other warnings associated with the functions of the Feedforward branch are used to indicate errors in the parameters. Details of these are given in the description of each function (WARN button in the channel zone).

### 2.3-5 Command and loop controller branch

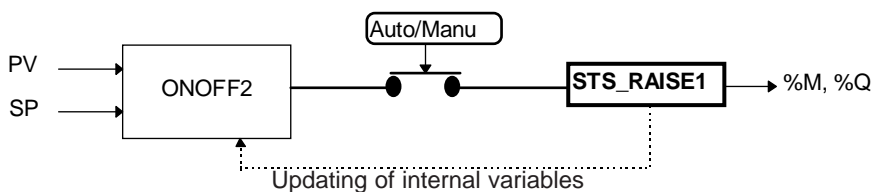
There are 8 types of loop controller:

- Autotuning PID
- Discrete loop controller :
  - 2-state,
  - 3-state,
- Hot/Cool PID controller,
- Split Range PID controller,
- Internal model-based controller,
- Hot/Cool internal model-based controller,
- Split Range internal model-based controller.

#### The 2-state ONOFF controller

This type of branch is made up of the single function, 2-state ONOFF. It is available in process loops and 3 single loops. When this type of branch is selected, there is no output branch or Feedforward branch.

##### General schematic



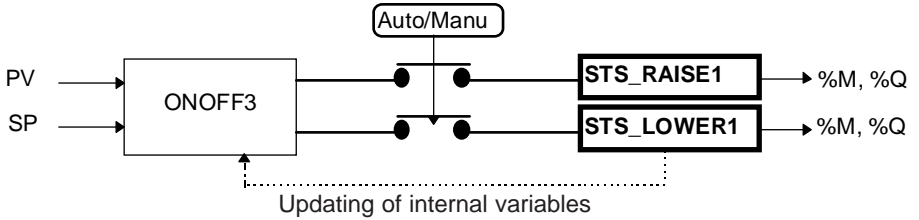
The controller output is copied to status bit STS\_RAISE1.

Updating the internal variables consists of taking the previous value of the command into account.

**The 3-state ONOFF controller**

This type of branch is made up of the single function, 3-state ONOFF. It is available in process loops and 3 single loops. When this type of branch is selected, there is no output branch or Feedforward branch.

General schematic



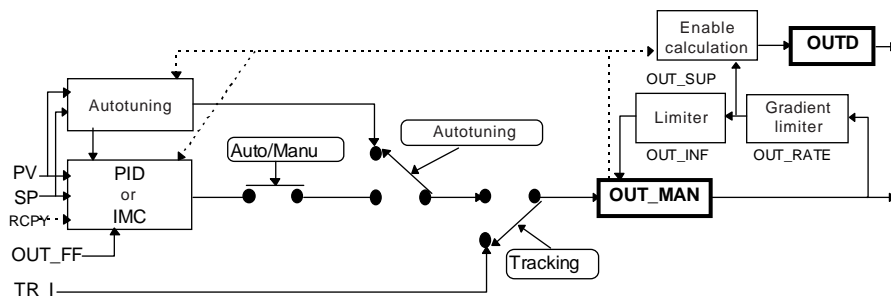
The controller output is copied to 2 status bits STS\_RAISE1 and STS\_LOWER1.

Updating the internal variables consists of taking the previous value of the command into account.

## The PID or IMC controller

The diagram below illustrates the basic PID controller branch. There are several variations, depending on the loops. Each variation is covered in the description of the various loops (see "Control loop operating mode").

### General schematic



For some functions, updating the internal variables consists of taking the previous value of the command into account. This makes smooth transitions to other modes possible and avoids saturation of the integral action by taking account of the limits set on the output.

The limits on the output apply in all operating modes of the controller.

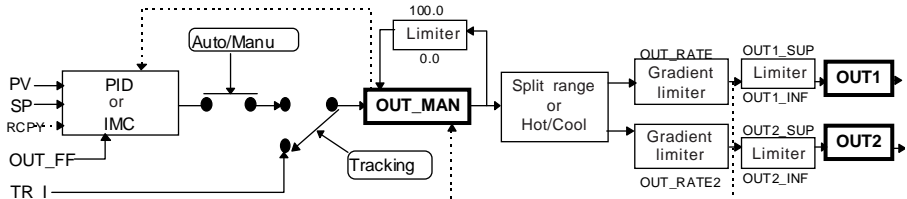
Note : If gradient limitation is used in manual mode, the value of OUT\_MAN (target value before limitation) can temporarily differ from the command applied to the output.

The RCPY input (external input address) only exists on the model-based controller.



## The split range or hot-cool controller (PID or IMC)

### General schematic



The IMC function is identical to that of the single IMC controller.

The PID function is identical to that of the single PID controller. The only differences are :

- no autotuning,
- OUT\_MAN limited to between 0 and 100,
- OUT\_BIAS = 0 (parameters cannot be set).

The OUT\_MAN command is reset according to the limits applied (as with single PID), to avoid problems associated with saturation of the integral action and operation of the split range or hot-cool function.

Each output in the split range or hot-cool function has its own level and gradient limits.

The operating mode acts on the controller output OUT\_MAN.

### Initialization

The consistency of the configuration entered is checked. If the configuration is not correct, the loop remains in an initialization state and the error is indicated in the status words. On a cold start, the parameters and the input values PV, SP, etc, associated with this branch are first updated before running the first processing operation of the controller. The initial operating modes of the controller can be selected by setting parameters in the process control configuration screen or by sending commands to a sequential program. Thus the loop can start in automatic or manual mode.

### Execution check

There are 2 types of fault :

- serious faults,
- warnings.

---

Two serious faults are tested for :

- Parameter error (it is not written in floating point format),
- Internal calculation error (division by zero, overflow, etc).

If a serious error is detected, processing of the controller changes to fallback state and the calculated value of the OUT command is frozen. The outputs of the control loop are maintained.

The state returns to normal when the error disappears. The loop then restarts smoothly on the outputs.

The input scale and output scale values are checked.

If there is an inconsistency (lower limit  $\geq$  upper limit) at a cold start, the loop changes to serious error mode.

Otherwise, the old (correct) values are restored.

Warnings are given in the status words.

### 2.3-6 Output processing branch

There are 3 types of output :

- analog output
- servomotor output
- PWM output

For all types of output, the OUT\_MAN command calculated by the controller crosses a limiter whose lower limits OUTi\_INF and upper limits OUTi\_SUP are used to define the output variation range. These limits define the output scale.

#### Analog output

Apart from the limitation aspect, there is no function specific to this processing. The calculated floating point value is converted to an integer for transmission on an analog channel (%QW) or to a memory word (%MW).

There are two conversion formats :

- unipolar (0 / 10000), default format,
- bipolar (-10000 / 10000).

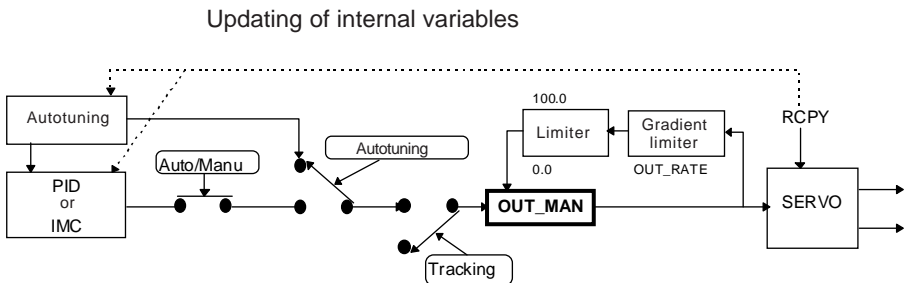
#### Servomotor output

This output consists of a SERVO function with or without position feedback from the actuator. After a split range or hot-cool PID, only the SERVO function with position feedback is available.

With this type of output, the scale of the controller output must be OUTi\_INF and OUTi\_SUP [0, 100].

Its processing period is the period of the task. If the controller is in automatic mode, the SERVO output takes into account a new output value from the controller at each sampling period. In other modes, this is performed on each task scan.

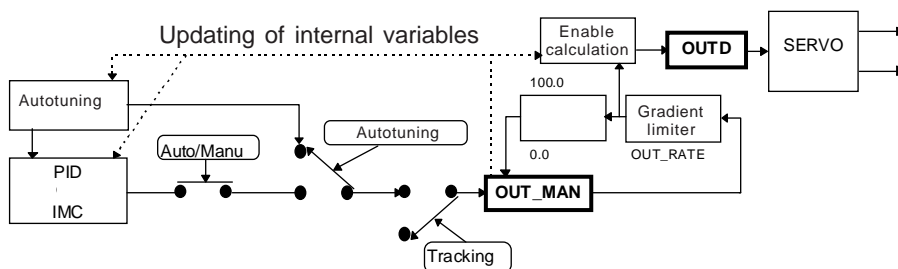
General schematic of a Servo output with position feedback (RCPY)



The schematic on the previous page illustrates the case of a PID followed by a SERVO function with position feedback (RCPY). The inputs of the SERVO function are thus the OUT\_MAN output of the controller and the position feedback of the actuator RCPY.

When the SERVO output follows a split range or hot-cool function, the feedback input is essential. The inputs of the SERVO function are then OUT1 or OUT2 and RCPY.

### General schematic of a Servo output without position feedback



The above schematic illustrates the case of a PID followed by a SERVO function without position feedback.

The input of the SERVO function is then the variation of the PID OUTD command of the PID. It should be noted that this is not affected by the output limitation on OUT\_MANU. This is used for floating point control, and the command calculated by the PID, OUT\_MAN, has no direct link with the actual position of the actuator. In particular, it is possible to continue opening and closing the motorized valve when OUT\_MAN is saturated.

### **PWM output**

This output consists of one PWM function, whose input is the OUT\_MAN command for PID controllers, and OUT1 or OUT2 for hot-cool or split range controllers.

With this type of output branch, the scale of the controller output must be  $OUT_{i\_INF}$  and  $OUT_{i\_SUP}$  [0, 100].

Its processing period is the period of the task. It is independent of the controller operating mode.

### Initialization

On startup, the parameters and the input value of the output branch are updated before the first processing operation.

If the output address is not filled in, processing is performed, but the output is not converted.

The consistency of the configuration entered is checked. If the configuration is incorrect, the loop remains in an initialization state.

#### Execution check

An error occurs during the processing of the output if :

- the value of a parameter is not written in floating point format,
- an internal calculation is not performed correctly (division by zero, overflow, etc).

In this case, the result of the output is frozen. When the error disappears, the state returns to normal and the output is recalculated smoothly.

During a cold start, if the scale values are not correct (non-floating point value or lower limit greater than upper limit), the loop remains in its initialization position and does not start. The outputs then retain their initial value. The loop starts when the error disappears.

## 2.3-7 Summary table

Branch	Single loop	Process loop	Cascaded loop		Autoselective loop	
			Master	Slave	Main	Secondary
Standard process value process	Yes	Yes	No	Yes	No	No
Standard process value single	Yes	Yes	Yes	Yes	Yes	Yes
Feedforward	No	Yes	Yes	No	Yes	No
Simple setpoint	Yes	Yes	Yes	-	Yes	Yes
Profile setpoint SPP	Yes	Yes	Yes	-	Yes	Yes
Selection setpoint	No	Yes	Yes	-	Yes	No
Setpoint with ratio	No	Yes	Yes	-	Yes	No
OnOff 2-state controller	Yes	Yes	No	No	-	-
OnOff 3-state controller	Yes	Yes	No	No	-	-
PID controller	Yes	Yes	Yes	Yes	Yes	Yes
Hot/Cool PID controller	No	Yes	-	Yes	No	-
Split range PID controller	No	Yes	-	Yes	No	-
IMC controller	Yes	Yes	Yes (*)	Yes (*)	Yes (*)	Yes (*)
Hot/Cool IMC controller	No	Yes	Yes (*)	Yes (*)	Yes (*)	Yes (*)
Split range IMC controller	No	Yes	Yes (*)	Yes (*)	Yes (*)	Yes (*)
Analog output	Yes	Yes	-	Yes	Yes	-
Servo output	Yes	Yes	-	Yes	Yes	-
PWM output	Yes	Yes	-	Yes	Yes	-

(\*) A single controller (either master or slave)

## 2.4 The setpoint programmer

### Description

The setpoint programmer offers a maximum of 6 profiles comprising 48 segments in total. The segments are numbered from 1 to 48. They are defined by the following parameters:

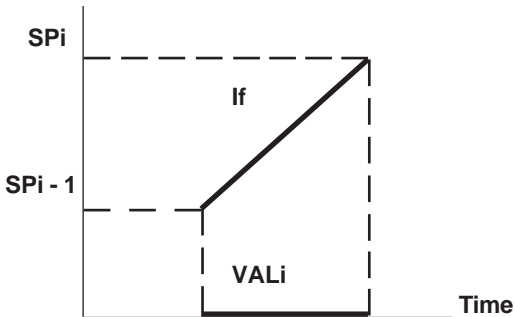
- $SP_i$  (%MF) : Setpoint to be reached
- $VAL_i$  (%MF) : Duration of the segment or segment slope (if it is a ramp).

A segment can be configured as a :

- ramp
- dwell step (in this case  $SP_i = SP_{i-1}$ ).

Each segment can be configured in seconds, minutes or hours.

### Setpoint

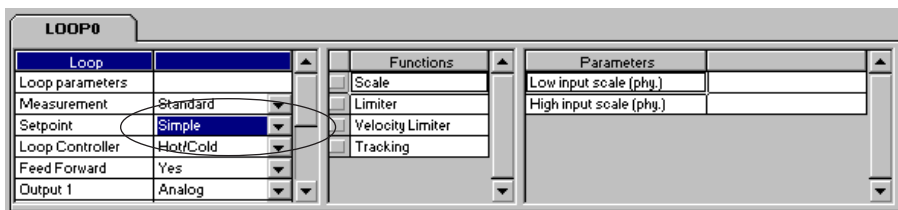


More precisely, it is possible to configure :

- One profile with a maximum of 48 segments
- One profile with a maximum of 32 segments and one profile with a maximum of 16 segments
- Two profiles with a maximum of 24 segments
- Three profiles with a maximum of 16 segments
- Four profiles with a maximum of 12 segments
- Six profiles with a maximum of 8 segments
- One profile with 24 segments, one profile with 16 segments and one profile with 8 segments.

### Link with a control loop

In the associated control loop(s), simply select a programmer as the type of setpoint and enter as the setpoint input address, the output address of the setpoint programmer, ie. %MFxy.i.20 (SP).

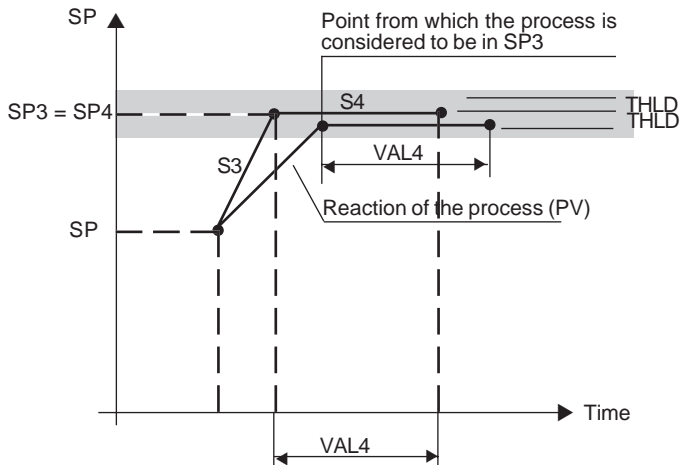


### Guaranteed dwell time

Since the reaction of a process to a change of setpoint varies in speed, it does not necessarily follow the variation in the setpoint calculated by the programmer. It is therefore possible to follow the evolution of a process value and guarantee a dwell time at the chosen setpoint : downcounting of the dwell time starts when the deviation between the setpoint and the process value is less than a defined threshold, THLD.

This guarantee can also be obtained on :

- high deviation overshoot
- low deviation overshoot
- high and low setpoint deviation overshoot. In this case, downcounting of the dwell time is frozen during each overshoot.



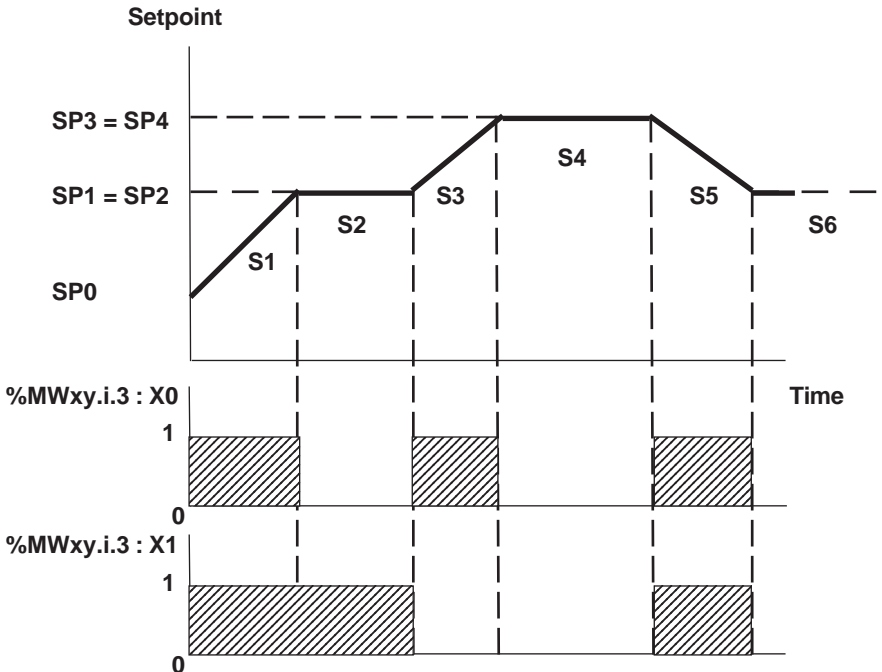


**Control outputs**

The programmer has 8 bit-type logic outputs (%MWxy.i.3 : X0 to X7), which can be associated with the segments in order to generate discrete actions.

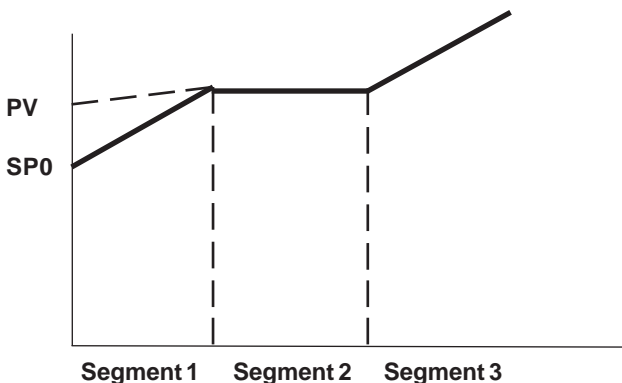
The following PL7 configuration screen is used to define the state of these logic outputs on each of the segments.

Segment #	SP #	VAL #	Unit	Pe	O0	O1	O2	O3	O4	O5	O6	O7
1 Ramp	0.5	0.4	Seconds		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>						
2 Plateau	0.5	0.3	Seconds									
3 Ramp	0.8	0.2	Seconds			<input checked="" type="checkbox"/>						
4 Plateau	0.8	0.8	Seconds				<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>		
5 Ramp	0.8	0.01	Seconds				<input checked="" type="checkbox"/>					
6 Plateau	0.8	0.0	Seconds									
7 Ramp	0.0	0.0	Seconds									
8 Plateau	0.0	0.0	Seconds									



### Bumpless start

A startpoint profile starts on an initial setpoint value SP0. For a smooth start, the profile can start from the process value PV and rejoin the setpoint SP1 depending on the characteristics of the first segment.



This function can also be used when looping back the profiles.

### Execution of a profile

PMX 57352 [RACK 0 POSITION 0]

Debug  Designation: PROCESSEUR PMX P 57352

Symbol: Loop Controller: Function: Task: Distribution of segments:

Loop Controller 5 - SPP\_1 Prog. de consigne MAST 8-8-8-8-8-8

Name SPP\_1 0 1 2 3 4 5 6 7

PROFILE\_1 PROFILE\_2 PROFILE\_3 PROFILE\_4 PROFILE\_5 PROFILE\_6

Segments Execution      Number of segments: 3 on 8

Plateau ensured at 5.0 on Deviation at input

Segment x	SP x	VAL x	Unité	Pq	S0	S1	S2	S3	S4	S5	S6	S7
1	Ramp	50.0	40.0	Seconds	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2	Plateau	50.0	40.0	Seconds	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Ramp	80.0	30.0	Seconds	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Current Profil 1 Repetition 1 Segment 2

PV SP 50.0 Total time elapsed 0 h. 0 m. 50 s. Time elapsed in segment 0 h. 0 m. 10 s.



A profile can be executed once, reproduced a number of times or continuously looped. The number of iterations is defined in a word (NBRE\_CYCLi) whose limits range from 1 to 32767.

If initialization segments are required, the first loop segment of a profile is not necessarily the first one. It can be chosen in the PL7 configuration screen. The first repeated segment is therefore not necessarily segment 1 of the profile.

To execute a profile, the following commands are used :

- START (%MWxy.i.7 = 16#0002) : Starts execution of the selected profile.
- STOP (%MWxy.i.7 = 16#0003) : Stops execution of the selected profile.
- RESET (%MWxy.i.7 = 16#0001) : Reinitializes the programmer and readies it for the START command.
- NEXT (%MWxy.i.7 = 16#0006) : Jumps to the next segment.
- BACK (%MWxy.i.7 = 16#0007) : Jumps to the previous segment.
- HOLD (%MWxy.i.7 = 16#0004) : Freezes changes in the setpoint and the calculation of time.
- DEHOLD (%MWxy.i.7 = 16#0005) : Unfreezes the current profile
- HOLD\_PG (%MWxy.i.7 = 16#0008) : Disables the guaranteed dwell time function on the current profile
- DEHOLD\_PG (%MWxy.i.7 = 16#0009) : Enables the guaranteed dwell time function on the current profile

The RESET command is always accepted.

The START command is only accepted if the programmer is in initialization mode

The NEXT and BACK commands are refused if the profile is not frozen

The STOP command is refused if the programmer is in initialization mode

The HOLD\_PG and DEHOLD\_PG commands are refused if the function is not being used

Each profile can be controlled from its application-specific screen.

### Setpoint programmer parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Process value input	-	real	-32768 / 32767	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	
Number of reiterations of profile	NB_RT_PFi	word	0 / 32767	1	R/W
Value of guar. dwell time thresh. of the profile	THLD_PFi	real	0.0 / 3.4 E38	0.0	R/W
Value of the initial setpoint of the profile	SPO_PFi	real	-3.4 E38 / 3.4 E38	0.0	R/W
Setpoint to be reached by the segment	SPI	real	-3.4 E38 / 3.4 E38	0.0	R/W
Value of the time or speed for the segment	VALi	real	-3.4 E38 / 3.4 E38	0.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	
No. of current profile	CUR_PF	word	0 / 32767	0	R
No. of current segment	SEG_OUT	word	0 / 32767	0	R
No. of current iteration	CUR_ITER	word	0 / 32767	0	R
Value of the calculated setpoint	SP	real	-3.4 E38 / 3.4 E38	-	R
Value of the total elapsed time (inc. freezes)	TOTAL_TIME	real	0.0 / 3.4 E38	-	R
Value of the time elapsed on the current segment (inc. freezes)	CUR_TIME	real	0.0 / 3.4 E38	0.0 / 3.4 E38	R

### Initialization

On initialization, the consistency of the configuration entered is checked. If the configuration is not correct, the setpoint programmer indicates the error and remains in an initialization state.

---

Execution check

An error occurs in the calculation of the setpoint if :

- The value of a parameter used is not written in floating point format,
- The internal calculation is not performed correctly (division by zero, overflow, etc).

In this case, the result of the setpoint calculation is frozen. The state returns to normal when the error disappears.

Other checks are used to indicate possible programming errors :

- A ramp with two identical setpoints,
- A ramp with a rise or fall speed of 0.0,
- A dwell step with two different setpoints.

In this case a warning is given, and the setpoint continues to be calculated. If the programmer is in the faulty segment, these errors cause :

- An immediate move to the next segment for the first error,
- Freezing of the calculated setpoint for the second error,
- Ascending or descending a ramp for the third error.

An additional warning is immediately activated. The text of this warning is "Error on the current segment ".

---

## 2.5 Global loop parameters

---

A number of general parameters associated with the control channels can be classified into two categories :

- Parameters linked to execution of the loop,
- Parameters which characterize the control loop.

---

### 2.5-1 Execution parameters

#### Task

The control channels must be assigned to a CPU processing task (MAST or FAST). The MAST task is selected by default, but if the processing must have a higher priority, the FAST task can be configured.

#### Sampling period (in seconds)

This parameter defines the processing period for the loop controller in automatic mode. The predefined default value is 0.3s. This value must be a modulo of the task period. If this is not the case, periodic processing of the process control application is performed on the closest modulo.

Example :  $T(\text{MAST}) = 0.1\text{s}$   $T\text{-ECH} = 0.124\text{s}$      $\Rightarrow T\text{-ECH} = 0.1\text{s}$

Similarly,  $T(\text{ECH})$  must not be shorter than the task period. If this is not the case, the sampling period will take the value of the task period.

Periodic calculations are performed taking this value,  $T\text{-ECH}$ , into account. If the task processing time exceeds the theoretical period, bit %S19 indicates this fault.

---

### 2.5-2 Instrumentation parameters

#### Name

A name integrated in the constants (%KW) can be given to each loop. This name, with a maximum of 8 characters, is automatically imported into the XBT process control runtime screens.

#### Unit

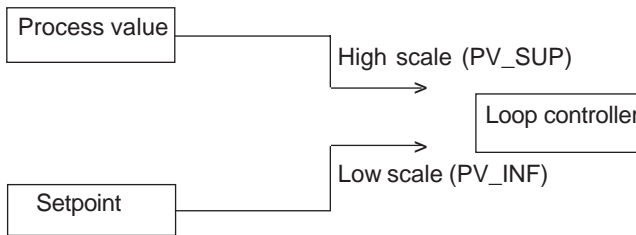
A maximum of 6 characters, integrated in the constants (%KW), defines the unit of the control loop (eg. DEGREE). This parameter is also automatically imported into the XBT process control runtime screens.

## Application ID

This parameter is used to identify the control loop configuration. These application IDs can be compared with one another for the purposes of authentication. They are the numerical representation of the loop configuration constants (%KW).

## Low scale, high scale

These thresholds define the physical scale in which the loop performs its process control operations. Calculations for the upstream branches (process value and setpoint) are both placed in the same scale.



**Note** : In the setpoint branch, there is a scale function which gives the scale range of the variable defined at the setpoint input. This function is useful, for example, when process loops are linked together to form a cascade. By default, this scale must be identical to the physical scale defined by the loop.

**Note** : It is also possible to apply specific scaling to the output branch (for further details, see the description of this function).

## 2.6 Detailed description of the math and logic functions

### 2.6-1 Process value processing functions

#### The "input format" function

##### Description

Only used for entering a Standard type process value, this function produces the raw value of the analog input of the loop. To do this, the format must be configured to be consistent with the type of the corresponding analog input channel.

There are two possible range formats :

- Unipolar : 0-10000 (default selection)
- Bipolar : -10000, +10000.

##### Assigning the input address :

The process value input address is entered in the graphic part of the PL7 process control configuration screen. It must be a word for a Standard type process value, that is :

- A %IW for an analog input
- A %MW memory word

##### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Process value input	-	%mW %IW	-32768 / 32767	-	R



- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Range	PV_UNI_BIP	bit of %KW	-	0 (unipolar)	R

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Effective input	PV_SIM	word	-32768 / 32767	0	R/W

#### Execution check

If no address is entered at the input of this function, the value read is the simulation value initially set to 0.

**External process value:** This function is not used for External process values. In fact, the input format for this process value is directly real in type. This input is then copied to the process value (PV) floating point variable at the loop controller input.

## The "1st order filtering" function

### Description

This function performs first order filtering with a time constant T.  
Its transfer function is :

$$\text{FILT\_OUT} = \text{GAIN\_FILT} \cdot \frac{1}{1 + pT\_FILT} \cdot \text{PV\_SIM}$$

where :

PV\_SIM : Function input value

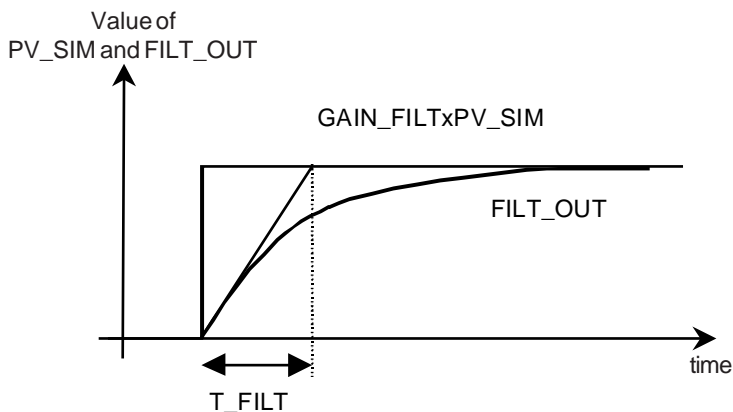
T\_FILT : Time constant

FILT\_OUT : Result of the function

p : Laplace operator

GAIN\_FILT : proportion coefficient

This first order filter is applied directly to the process value input.



### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Effective input	PV_SIM	word	-32768 / 32767	0	R/W

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Filtering time (in s)	T_FILT	real	0.0 / 3.4 E38	0.0	R/W
Gain	GAIN_FILT	real	-3.4 E38 / 3.4 E38	1.0	R/W



- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Output value	FILT_OUT	real	-3.4 E38 / 3.4 E38	-	R W

Note : If the function is not selected, the value of its output is a copy of the value of its input.

Execution check

The parameter check for this function is integrated in the process value branch error handling. If the time constant is negative, its value is rewritten as 0.0.

**The "Square root" function**

Description

This function calculates the square root of a numerical value. The extraction of the square root is typically used to linearize a flow rate measurement taken by a constricting device.

The function performs the following calculation :

$$\begin{aligned}
 \text{OUT} &= 100 \cdot \sqrt{\text{FILT\_OUT}} && \text{if } \text{FILT\_OUT} \geq 0 \\
 \text{OUT} &= 0 && \text{if } \text{FILT\_OUT} < 0
 \end{aligned}$$

Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Function input	FILT_OUT	real	-3.4 E38 / 3.4 E38	-	R

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Output value	SQRT_OUT	real	-3.4 E38 / 3.4 E38	-	R

Note : If the function is not selected, the value of its output is a copy of the value of its input.

Execution check

There is no special check for this function. The parameter check is integrated in the process value branch error handling.

## The "Function generator" function

### Description

The function generator corrects non-linearity of the process value input signal. The non-linearity is corrected using 7 contiguous linear segments, at variable intervals, defined by the coordinates of their points.

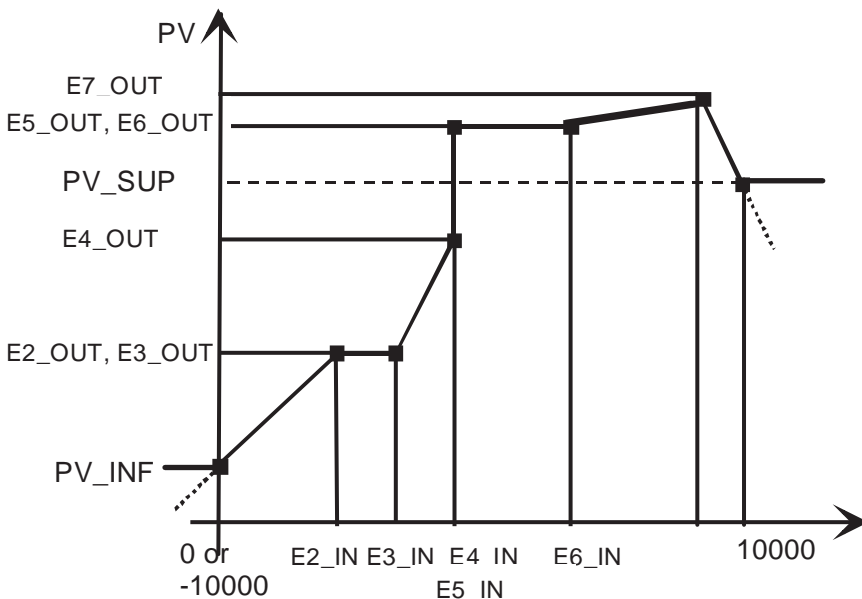
The function also performs a scaling operation. It is therefore exclusive with the scaling function described below. The output is calculated by linear interpolation between 2 points whose abscissae are either side of the input parameter value.

$$PV = f(x) = \{ (X1, Y1), \dots, (X7, Y7) \}$$

where :

X1 = 0 or -10000 and Y1= PV\_INF (lower limit of the loop scale)

X7 = 10000 and Y7= PV\_SUP (upper limit of the loop scale)



Outside the input scale, it is possible, by configuration, to perform an extrapolation or to limit the calculated PV value to within the scale of the process value.

**Parameters**

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Function input	SQRT_OUT	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Extrapolation	EXTRAPOL	bit constant	-	0 (no)	R
Abscissa 1	-	real	-3.4 E38 / 3.4 E38	0 or -10000	-
Abscissa 2	E2_IN	real	-3.4 E38 / 3.4 E38	1428	R/W
Abscissa 3	E3_IN	real	-3.4 E38 / 3.4 E38	2857	R/W
Abscissa 4	E4_IN	real	-3.4 E38 / 3.4 E38	4285	R/W
Abscissa 5	E5_IN	real	-3.4 E38 / 3.4 E38	5714	R/W
Abscissa 6	E6_IN	real	-3.4 E38 / 3.4 E38	7143	R/W
Abscissa 7	E7_IN	real	-3.4 E38 / 3.4 E38	8571	R/W
Abscissa 8	-	real	-3.4 E38 / 3.4 E38	10000	-
Ordinate 1	PV_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Ordinate 2	E2_OUT(phy)	real	-3.4 E38 / 3.4 E38	14.28	R/W
Ordinate 3	E3_OUT(phy)	real	-3.4 E38 / 3.4 E38	28.57	R/W
Ordinate 4	E4_OUT(phy)	real	-3.4 E38 / 3.4 E38	42.85	R/W
Ordinate 5	E5_OUT(phy)	real	-3.4 E38 / 3.4 E38	57.14	R/W
Ordinate 6	E6_OUT(phy)	real	-3.4 E38 / 3.4 E38	71.43	R/W
Ordinate 7	E7_OUT(phy)	real	-3.4 E38 / 3.4 E38	85.71	R/W
Ordinate 8	PV_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the process value	PV(phy)	real	-3.4 E38 / 3.4 E38	-	R

**Note** : The parameters PV\_INF and PV\_SUP are defined in the global loop parameters, see this section for further details on these parameters.

---

### Execution check

The parameter check is integrated in the process value branch error handling. The coordinates of the abscissae should normally be increasing. If  $E_{j+1\_IN} < E_{j\_IN}$ , a Warning is given.

The calculation continues to be performed even with the current parameters.

### The "Scaling" function

The process value branch is scaled automatically based on the global loop parameters PV\_INF and PV\_SUP (see section 2.5).

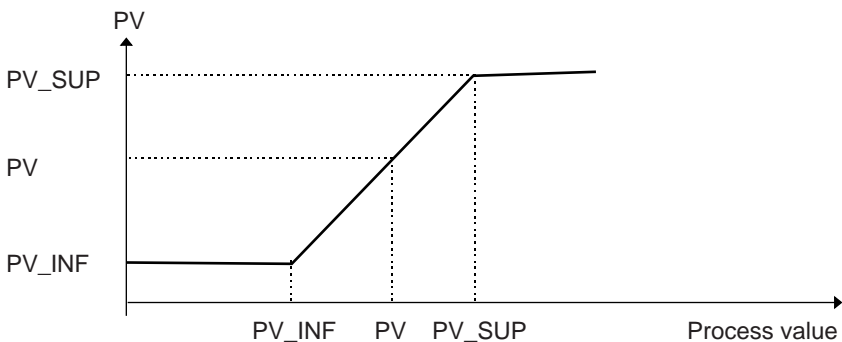
### The "Scale limiter" function

#### Description

This function is used to limit the process value with respect to the physical scale defined for the control loop.

If this function is activated, scaling is only performed within the limits of the range (PV\_INF, PV\_SUP).

Outside this range, the output is limited to the scale values.



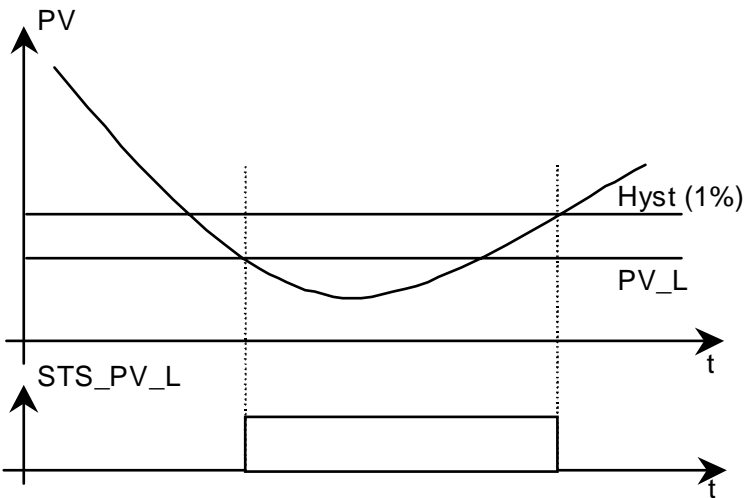
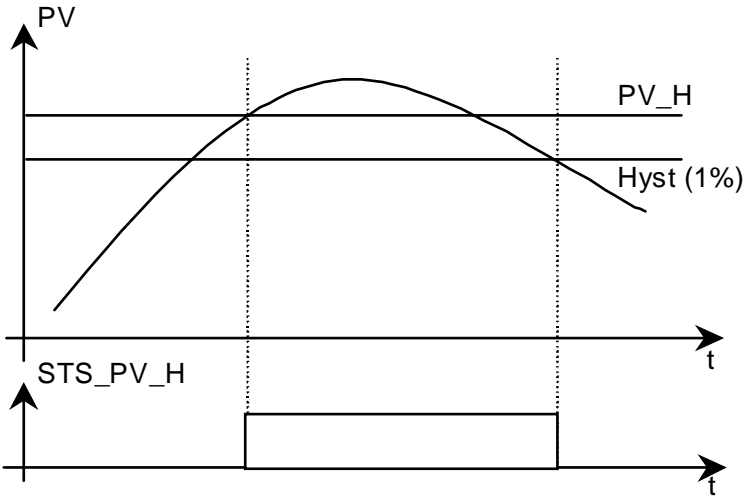
**The "Alarm on level" function**

Description

This function monitors the evolution of the process value by comparing its value with 4 thresholds traditionally known as LL, L, H, HH.

Each alarm feeds back an associated status bit.

These alarms are monitored with a fixed hysteresis of 1% of the scale defined in the global loop parameters.



## Parameters

### • Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the process value (phy)	PV	real	-3.4 E38 / 3.4 E38	-	R

### • Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Very low threshold (phy)	PV_LL	real	-3.4 E38 / 3.4 E38	5.0	R/W
Low thresh. (phy)	PV_L	real	-3.4 E38 / 3.4 E38	5.0	R/W
High thresh. (phy)	PV_H	real	-3.4 E38 / 3.4 E38	95.0	R/W
Very high threshold (phy)	PV_HH	real	-3.4 E38 / 3.4 E38	95.0	R/W

### • Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Very low limit	STS_PV_LL_LIM	bit	-	-	R
Low limit	STS_PV_L_LIM	bit	-	-	R
High limit	STS_PV_H_LIM	bit	-	-	R
Very high limit	STS_PV_HH_LIM	bit	-	-	R
(*) OR of the alarms	STS_ALARMS	bit	-	-	R

(\*) The OR of the alarms is the logical sum of the level alarms and the deviation alarms.

### Execution check

The parameter check is integrated in the process value branch error handling.



## The "Totalizing" function

### Description

This function integrates the value of the input (typically a flow rate) as a function of time, and sends back a total (typically a volume).

It uses an internal partial accumulator Acc which integrates the PV value and is automatically reinitialized to 0 each time it reaches an adjustable threshold THLD. The number of reinitializations CptlNit is also stored so that the overall total OUT\_TOT can be produced.

The principle of the function is as follows :

Each time the function is executed the accumulator Acc and the total OUT\_TOT are calculated using the following algorithm :

```

Acc(new) = Acc(old) + PV . DT
IF Acc(new) >= THLD THEN
    Acc(new) = Acc(new) - THLD
    CptlNit = CptlNit + 1
ENDIF
OUT_TOT = CptlNit x THLD + Acc(new)

```

where : DT = task period

ACC(old) = value of the accumulator ACC in the previous scan.

### Adjusting the integration threshold THLD

The value of the integration threshold generally corresponds to a characteristic of the process which is easy to determine (for example the capacity of a tank).

A status bit is set during a cycle each time the partial accumulator reaches the integration threshold.

The function can also be used to integrate small values from an input, even when the result of the integration is very large.

In this case, the risk is that the values to be integrated become negligible in relation to the accumulated value, and are therefore no longer taken into account.

The proposed solution is to limit the accumulator to a threshold THLD, so that the value to be integrated is never negligible in relation to this partial accumulator.

When threshold THLD is 0, the function does not integrate any value, and the function output remains frozen.

### Time base

The ratio between the cumulative value and the PV value must be less than  $10^9$  so that the new process value can be integrated.

### Associated commands

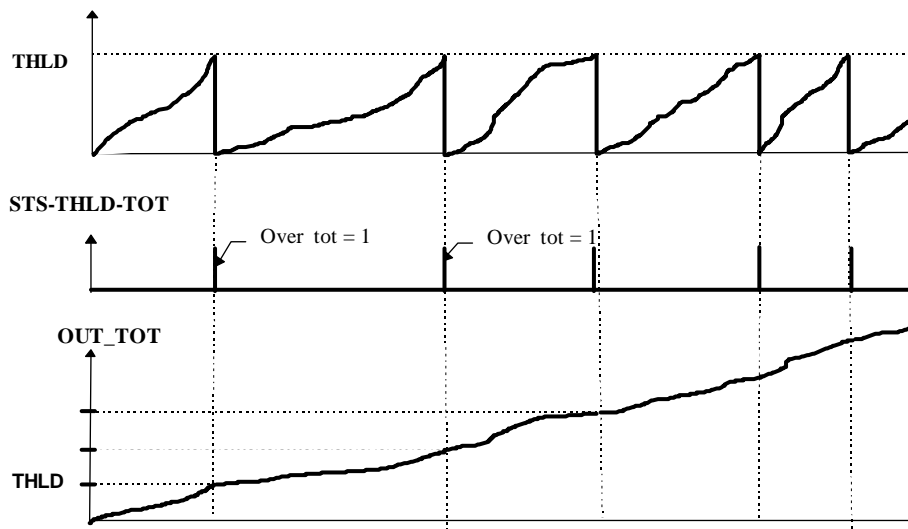
The function has two specific control methods :

- the reset action (reset to 0),
- hold mode (output frozen).

In reset mode, the OUT\_TOT output of the function changes to 0, as do all the internal variables. Reinitialization by resetting is also used to restart from zero (for example following a change of phase in manufacturing).

In hold mode (output frozen), integration is suspended. The function output retains its old value. In this mode, the user can modify the total value, OUT\_TOT, in which case the internal variables are recalculated. This enables the total value to be reset, for example after a control system stoppage.

### Time diagram



### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the process value	PV	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Totalizing threshold	THLD	real	0 / 3.4 E38	1.E8	R/W
Time base (hour)	-	bit of %KW	-	-	R

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Totalizing value	OUT_TOT	real	0 / 3.4 E38	0.0	R
Thresh. reached	STS_THLD_TOT	bit	-	-	R

Execution check

The parameter check is integrated in the process value branch error handling.

C

## 2.6-2 Setpoint branch functions

### The "Ratio" function

#### Description

The function is used to control a ratio.

The purpose of ratio control is to link a controlled value with a declared input (control value). The role of this function is to calculate the setpoint of the loop controller as a function of the control value using the following formula :

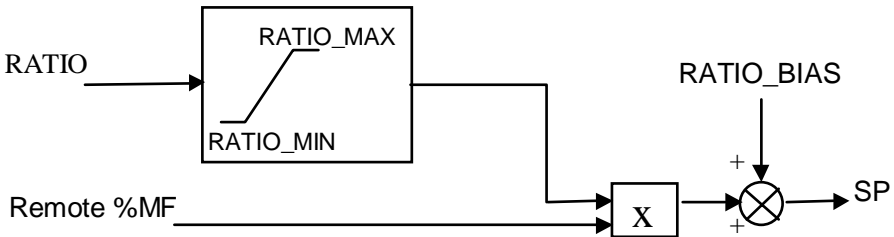
$$SP = \text{RATIO} \cdot (\text{SP Remote1}) + \text{RATIO\_BIAS}$$

SP Remote is the control value

High and low limits can be given for the ratios.

Note : Within the framework of this function, the type of value connected to the SP Remote1 input is an external process value rather than a setpoint.

#### Block diagram



#### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Setpoint input	-	%MFi	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Ratio value	RATIO	real	-3.4 E38 / 3.4 E38	1.0	R/W
Min. ratio value	RATIO_MIN	real	-3.4 E38 / 3.4 E38	0.0	R/W
Max. ratio value	RATIO_MAX	real	-3.4 E38 / 3.4 E38	100.0	R/W
Bias of the ratio	RATIO_BIAS	real	-3.4 E38 / 3.4 E38	0.0	R/W

---

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the setpoint	SP	real	-3.4 E38 / 3.4 E38	-	R
Scale warning	RATIO_WARN	bit	-	-	R

#### Execution check

The parameter check is integrated in the setpoint branch error handling.

---

## The "Selection" function

### Description

The selection function is used to select a setpoint by comparing two numeric inputs. This selection can be :

- Max. selection (the Remote 1 setpoint input is greater than the Remote2 setpoint input),
- Min. selection (the Remote 1 setpoint input is less than the Remote2 setpoint input)
- “ Switch ” selection (input selected by explicit command).

Switching is instantaneous, performed without hysteresis.

### Execution check

The check for this function is integrated in the setpoint branch error handling.

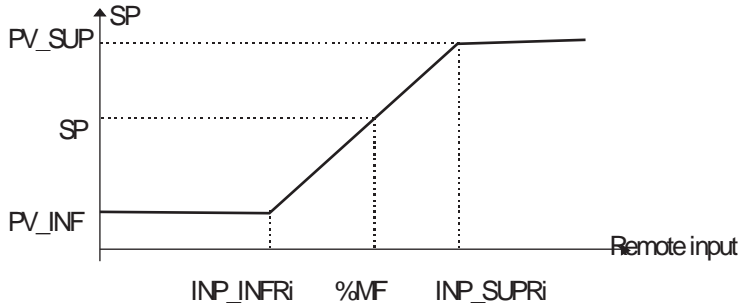
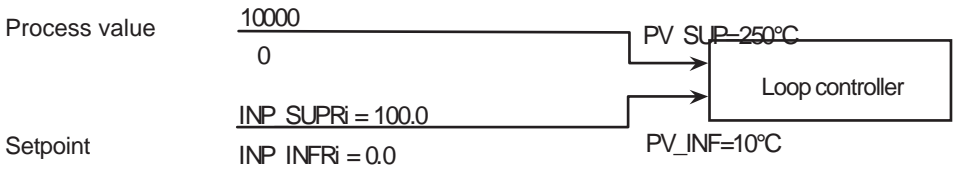
**The "Scale" function**

Description

This function is used to express the value at the setpoint branch input within the process value scale defined by PV\_INF and PV\_SUP (loop parameters). It takes account of the range within which the setpoint input address is located (INR\_INF<sub>Ri</sub>, INR\_SUP<sub>Ri</sub>). It is applied to setpoints Remote1 and Remote2. It performs the following calculation :

$$SP = (IN - INP\_INF<sub>Ri</sub>) \cdot \frac{PV\_SUP - PV\_INF}{INP\_SUP<sub>Ri</sub> - INP\_INF<sub>Ri</sub>} + PV\_INF$$

This function is optional, and is useful for linking 2 loops, for example to create a cascade with 2 process loops.  
By default, INR\_INF<sub>Ri</sub> = PV\_INF, INR\_SUP<sub>Ri</sub> = PV\_SUP.



## Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Setpoint input	-	%MFi	-3.4 E38 / 3.4 E38	-	R/W

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Low input scale (phy)	INP_INFRi	real	-3.4 E38 / 3.4 E38	0.0	R/W
High input scale (phy)	INP_SUPRi	real	-3.4 E38 / 3.4 E38	100.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the setpoint (phy)	SP	real	-3.4 E38 / 3.4 E38	-	R

### Execution check

The parameter check for this function is integrated in the setpoint branch error handling. If  $In\_MINRi \geq In\_MAXRi$ , the setpoint output remains unchanged. A warning is given in the status words.



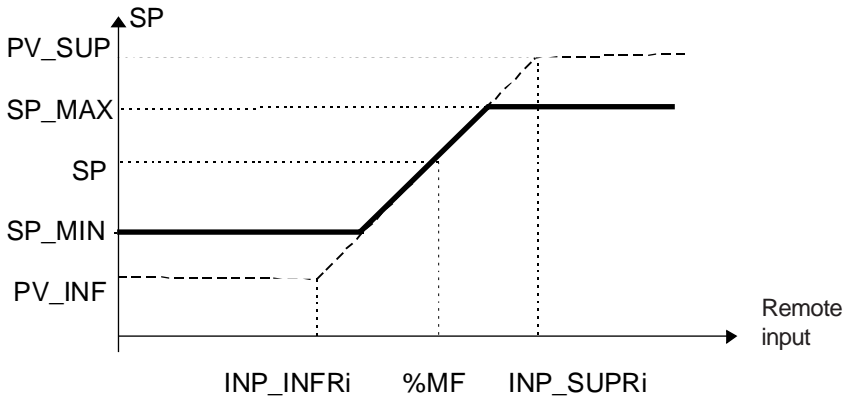
**The "Setpoint limiter" function**

Description

When this function is activated, scaling is only performed within the limits of the range defined by its parameters SP\_MIN and SP\_MAX.

When it is not activated the value of the setpoint is limited to the physical scales of the control loop.

The interval (SP\_MIN / SP\_MAX) must be within the interval (PV\_INF / PV\_SUP).



Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the setpoint	SP	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Lower setpoint limit (phy)	SP_MIN	real	-3.4 E38 / 3.4 E38	0.0	R/W
Upper setpoint limit (phy)	SP_MAX	real	-3.4 E38 / 3.4 E38	100.0	R/W

---

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the setpoint (phy)	SP	real	-3.4 E38 / 3.4 E38	-	R

---

#### Execution check

The parameter check for this function is integrated in the setpoint branch error handling.

If  $SP\_MIN \geq SP\_MAX$ .

$SP\_MIN < PV\_INF$ .

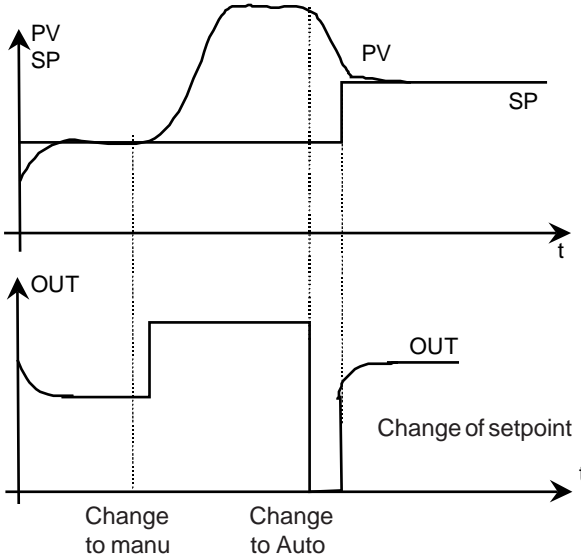
$SP\_MAX > PV\_SUP$ .

The setpoint output remains unchanged. A warning is given in the status words.

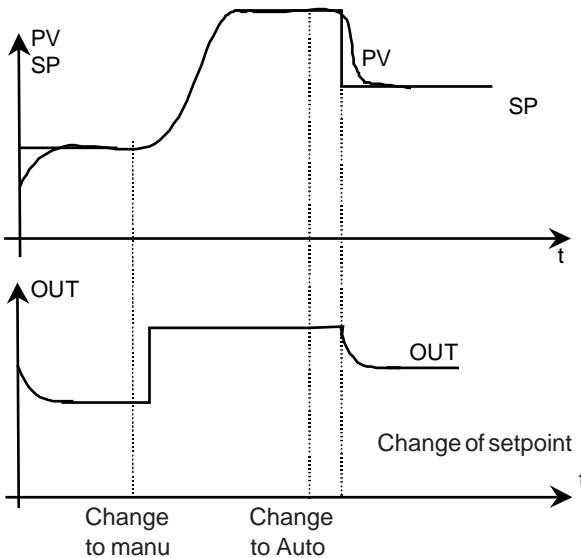
**The "Tracking" function**

Description

This function is used to make the local setpoint value track the PV value when the loop controller is not in automatic mode. Thus, on returning to automatic mode, there are no jerks at the loop controller output.



Function not configured



Function configured

## The "Speed limiter" function

### Description

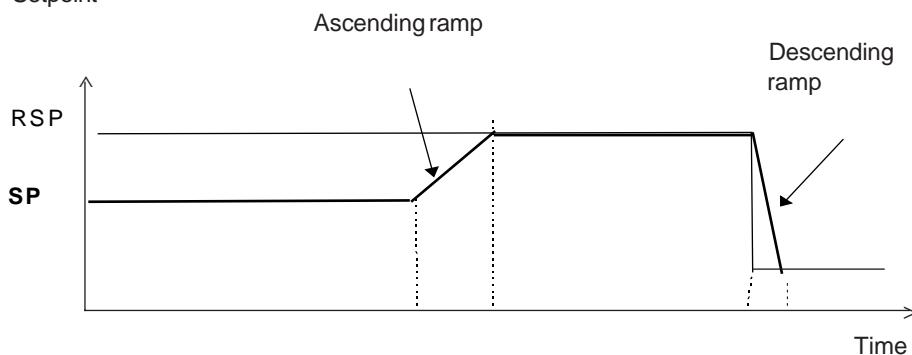
The function is used to wait for the new value, observing a speed limit, when the setpoint changes. The speed increase and speed decrease limits can be different.

When the requested input value is greater than the current value of the SP output, the function increases the value of this output at speed  $R\_rate$ , until the SP value reaches the required level. If the value of  $R\_rate$  is zero, no ramp is implemented. SP is a direct copy of the input value.

When the input value changes during generation of the ramp, the function attempts to reach this new target.

### Block diagram

Setpoint



### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the setpoint (phy)	SP	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Speed increase limit (phy/s)	R_RATE	real	0 / 3.4 E38	0.0	R/W
Speed decrease limit (phy/s)	D_RATE	real	0 / 3,4 E38	0.0	R/W

• Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the limited setpoint (phy)	SPEED_LM_OUTreal		-3.4 E38 / 3.4 E38	-	R/W

Note : R\_RATE and D\_RATE at 0.0 means that there is no limit.

This function can be applied to the Remote setpoint and the local setpoint, or to the local setpoint only, depending on the selected configuration.

Execution check

The parameter check for this function is integrated in the setpoint branch error handling.

## 2.6-3 Feedforward branch functions

### The "Scale" function

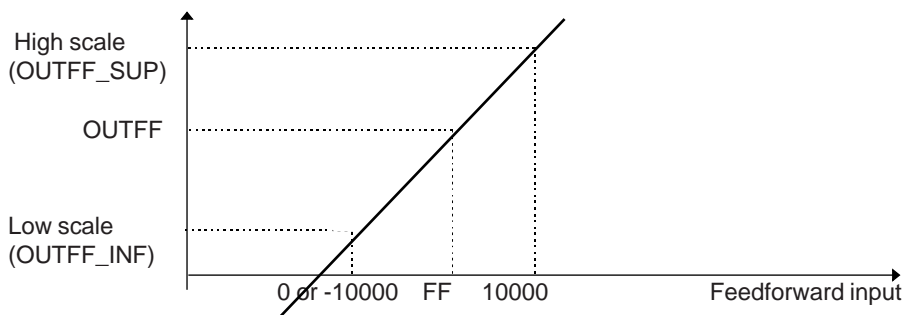
#### Description

This function is used to change the scale of the numeric input value of the Feedforward.

$$\text{OUTFF} = (\text{FF} - \text{in\_min}) \cdot \frac{(\text{OUTFF\_SUP} - \text{OUTFF\_INF})}{(\text{in\_max} - \text{in\_min})} + \text{OUTFF\_INF}$$

where :  $\text{in\_min} = 0$  or  $-10000$

and :  $\text{in\_max} = 10000$



#### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Feedforward input	-	%MW %IW	-32768,32767	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Low scale (phy)	OUT_FF_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
High scale (phy)	OUT_FF_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Feedforward value (phy)	OUTFF	real	-3.4 E38 / 3.4 E38	-	R/W

Execution check

The parameter check for this function is integrated in the Feedforward branch error handling.

There is no check on the order of the scale parameters. The lower limit can have a value greater than the upper limit.

## The "Lead-lag" function

Lead-lag performs a phase lead-lag transfer function. Used in this way, it models the effect of disturbance and thus performs open loop control by feedforward.

The lead-lag function performs the following transfer function :

$$\text{OUTFF} = \frac{1 + p \cdot T1\_FF}{1 + p \cdot T2\_FF} \cdot \text{FF}$$

where :

FF : analog input value : internal variable (measurement of the disturbance)

T1\_FF : time constant corresponding to the phase lead

T2\_FF : time constant corresponding to the phase lag

p : Laplace operator

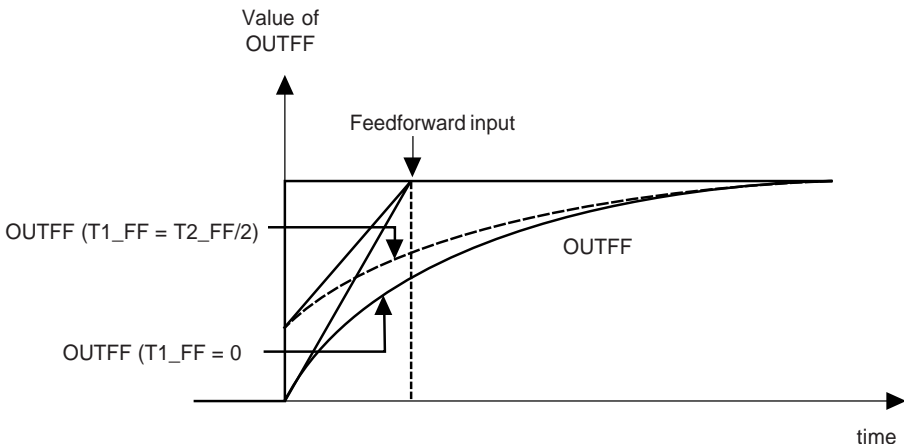
OUTFF : calculated value

The response of the OUTFF output to a step function at the input depends on T1\_FF and T2\_FF (phase lead or phase lag).

If  $T1\_FF > T2\_FF$  : phase lead

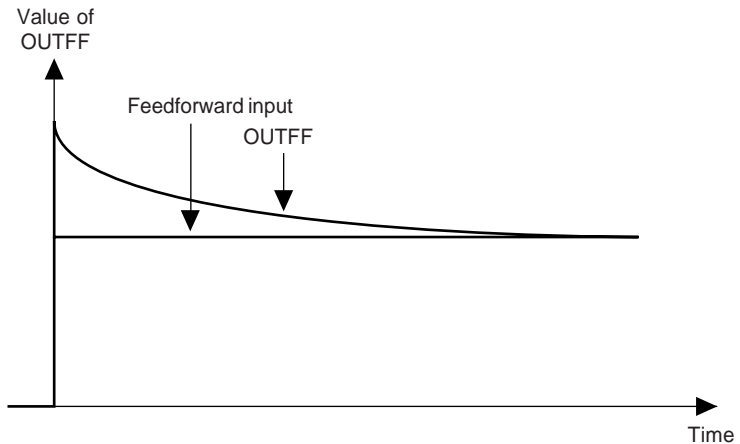
If  $T1\_FF < T2\_FF$  : phase lag

- Lead-lag configured for phase lag ( $T1\_FF < T2\_FF$  : output OUTFF lags in relation to the input).





- Lead-lag configured for phase lead ( $T1\_FF > T2\_FF$  : output OUTFF leads in relation to the input).



Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Feedforward input	-	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Time 1 (s)	T1_FF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Time 2 (s)	T2_FF	real	-3.4 E38 / 3.4 E38	0.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Feedforward value (phy)	OUTFF	real	-3.4 E38 / 3.4 E38	-	R

Execution check

The parameter check for this function is integrated in the Feedforward branch error handling.

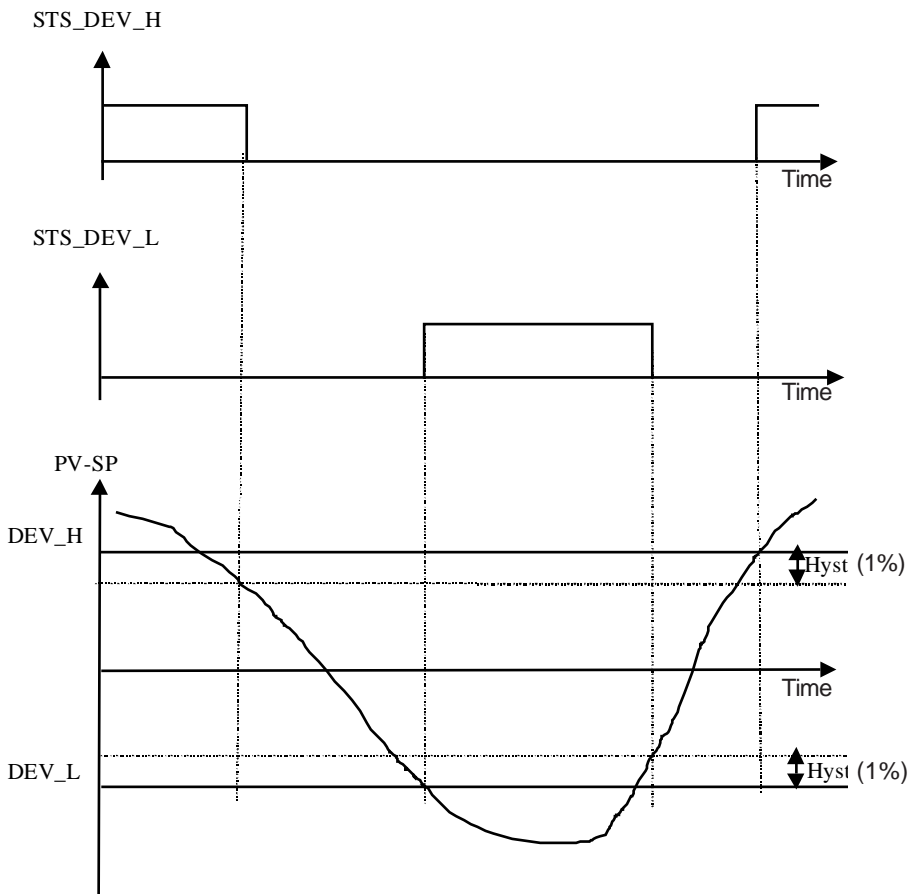
## The "Alarm on deviation" function

### Description

This function monitors the evolution of the deviation between the process value (PV) and the setpoint (SP) by comparing these values with 2 thresholds traditionally known as  $\Delta H$  (high deviation threshold) and  $\Delta L$  (low deviation threshold).

These alarms are checked with a fixed hysteresis of 1% of the full scale of the loop.

Note: The value of the thresholds must exceed the hysteresis (1%), otherwise the alarms are always active.



Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the process value	PV	real	-3.4 E38 / 3.4 E38	-	R
Value of the setpoint	SP	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
High dev. thresh.	DEV_H	real	0 / 3.4 E38	5.0	R/W
Low dev. thresh.	DEV_L	real	-3.4 E38 / 0	- 5.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
High limit	STS_DEV_H	bit	-	-	R
Low limit	STS_DEV_L	bit	-	-	R
(*) OR of the alarms	STS_ALARMS	bit	-	-	R

(\*) The OR of the alarms is the logical sum of the level alarms and the deviation alarms.

Execution check

The parameter check is integrated in the process value branch error handling.

## 2.6-4 Loop controller branch functions

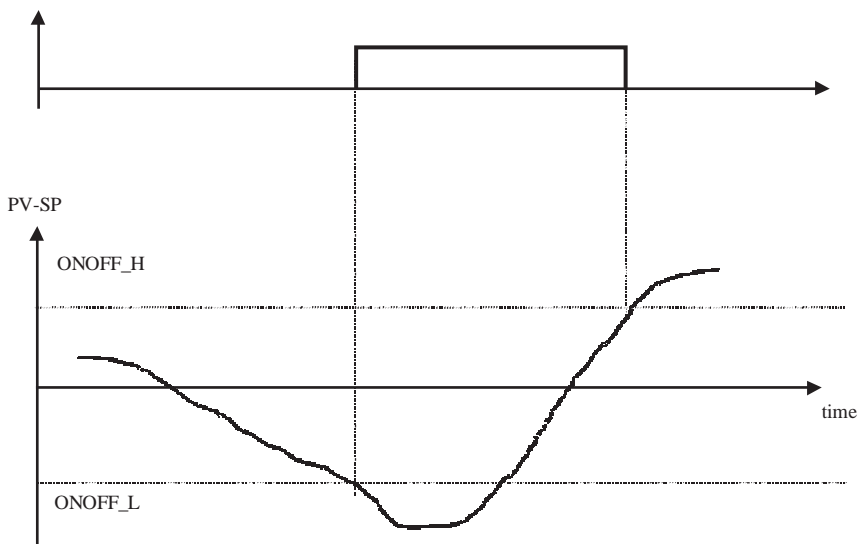
### The 2-state ONOFF loop controller

This loop controller is used for simple process control, for which 2-position discrete control is adequate.

The actuator is controlled according to the position of the process value-setpoint deviation in relation to two thresholds (one high and one low).

#### Block diagram

STS\_RAISE1



#### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the process value	PV	real	-3.4 E38 / 3.4 E38	-	R
Value of the setpoint	SP	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

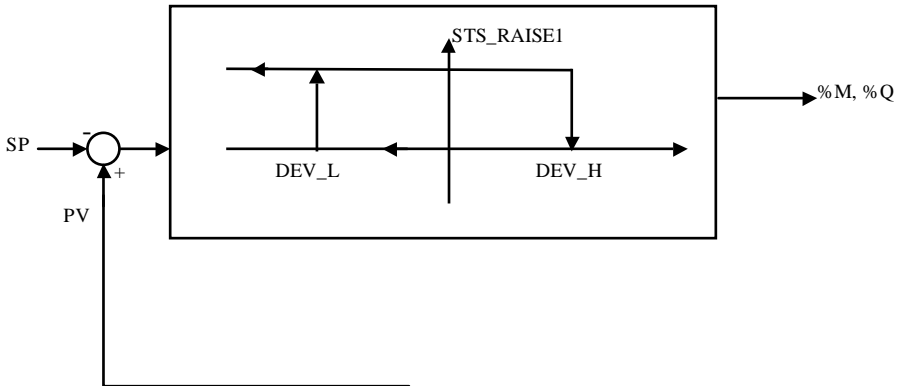
Meaning	Symbolization	Data type	Variation range	Default value	R/W
Low threshold	ONOFF_L	real	-3.4 E38 / 3.4 E38	-5.0	R/W
High threshold	ONOFF_H	real	-3.4 E38 / 3.4 E38	5.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Command status	STS_RAISE1	bit	-	-	R
Auto_man status	STS_M_A	bit	-	-	R
Command	-	bit	-	-	R
Setpoint-process value deviation	DEV	real	-3.4 E38 / 3.4 E38	-	R

As soon as the deviation ( $DEV = PV - SP$ ) becomes less than the low threshold `ONOFF_L`, the logic output changes to 1.

However, when the deviation increases again, the output only returns to 0 when the deviation exceeds `ONOFF_H`.



Operating modes

This loop controller has two operating modes :

- Automatic mode, the output is calculated by the loop controller itself.
- Manual mode, the loop controller does not set the output, the operator can directly modify the value of the variable connected on the output.  
On a cold start, the state of the output in manual mode is 0.

---

### Execution check

An execution error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters
- A problem occurs in a floating point calculation,
- If the low threshold  $> 0$ ,
- If the high threshold  $< 0$ .

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

### Operating modes and associated commands

For further details, please refer to section x.y-z, which describes the operating modes of the loop controller.

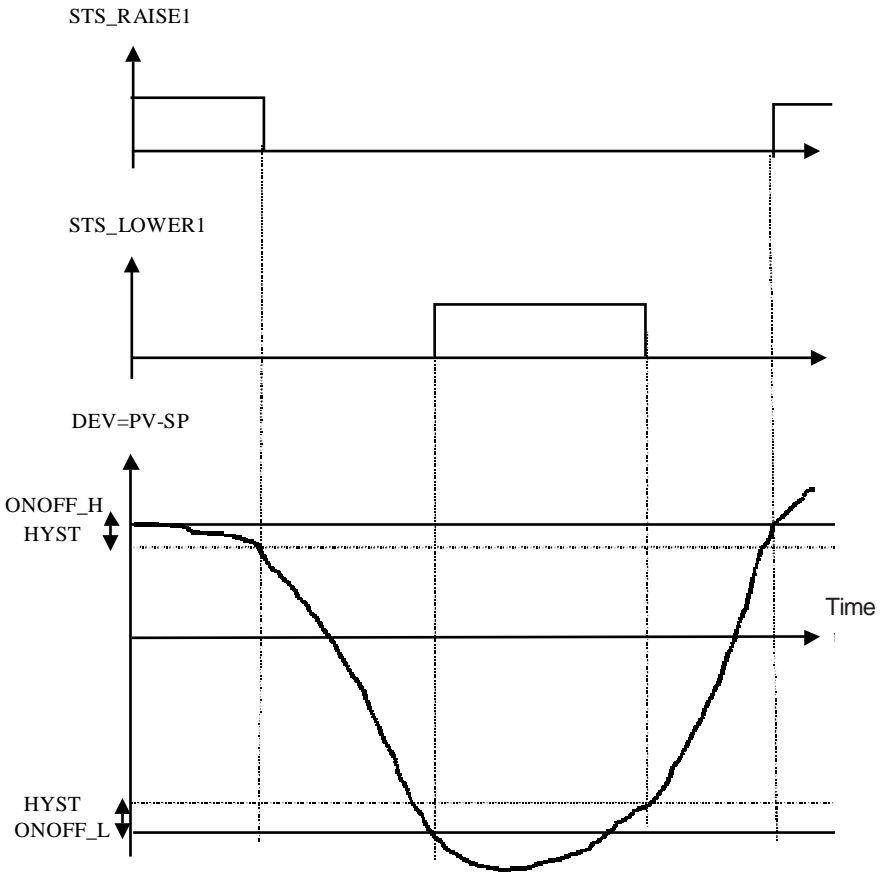
**The 3-state ONOFF controller**

This loop controller is used for simple process control, for which 3-position discrete control is adequate. Two actuators are controlled according to the position of the process value-setpoint deviation in relation to two thresholds (one high and one low).

This threshold management integrates a configurable hysteresis. This loop controller can, for example, be used for discrete control of a hot/cool process.

For more complex process control, it is preferable to use a standard PID controller.

Block diagram



## Parameters

Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the process value	PV	real	-3.4 E38 / 3.4 E38	-	R
Value of the setpoint	SP	real	-3.4 E38 / 3.4 E38	-	R

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Low threshold	ONOFF_L	real	-3.4 E38 / 3.4 E38	-5.0	R/W
High threshold	ONOFF_H	real	-3.4 E38 / 3.4 E38	5.0	R/W
Hysteresis	HYST	real	ONOFF_L, ONOFF_H	0.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Process value-setpoint deviation	DEV	real	-3.4 E38 / 3.4 E38	-	R
Status of command OUT1	STS_RAISE1	bit	-	-	R
Status of command OUT2	STS_LOWER1	bit	-	-	R
Auto_manu	STS_M_A	bit	-	-	R
Value of the command	OUT1	bit	-	-	R
Value of the command	OUT2	bit	-	-	R

## Operating modes

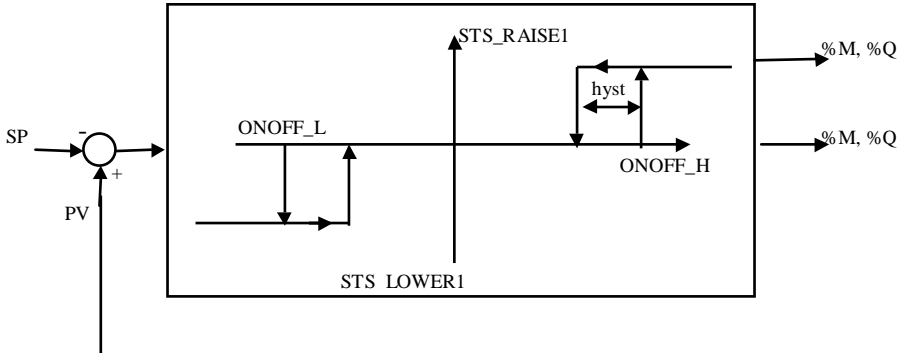
The 3-state OnOff controller has 2 operating modes :

- Automatic mode, outputs STS\_LOWER1 and STS\_RAISE1 are calculated by the block itself.
- Manual mode, the loop controller does not set the logic outputs, the operator can directly modify the value of the variables connected on the STS\_LOWER1 and STS\_RAISE1 outputs.



Operating modes and associated commands

For further details, please refer to section x.y-z, which describes the operating modes of the loop controller.



Execution check

An error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters
- A problem occurs in a floating point calculation
- If the low threshold > 0,
- If the high threshold < 0.

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

## The "PID" function

The PID function executes a PID algorithm with parallel or mixed structure (serial / parallel).

It has a large number of functions, including :

- Calculation of proportional, integral and derivative actions in incremental or absolute form,
- Anti-saturation of the integral action,
- Direct or inverse action,
- Derivative on process value or deviation,
- Configuration of transient derivative gain,
- Integral band
- Feedforward action to compensate for disturbance
- Dead band on the deviation
- High and low limits of the output signal
- Limitation of the output signal gradient
- Output bias, also known as manual integral
- Selection of Manual / Automatic operating mode
- Tracking mode
- Autotuning of the main coefficients

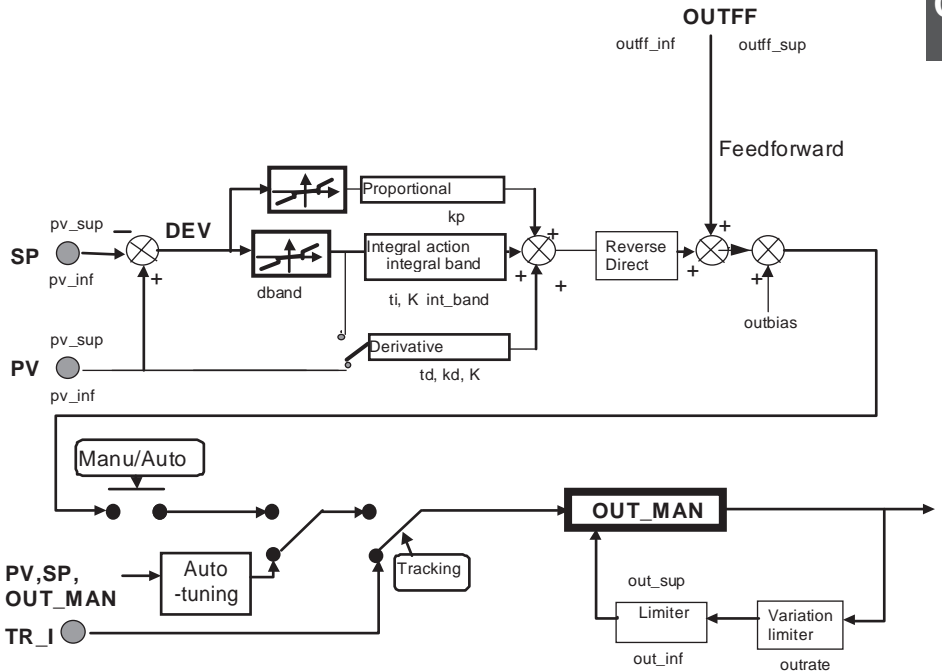
Depending on whether the mixed or parallel structure is used, the transfer function is as follows :

$$\text{Mixed structure : } \quad OUT = kp \left( 1 + \frac{1}{ti \times p} + \frac{td \times p}{1 + \left( \frac{td}{kd} \right) \times p} \right) \times IN$$

$$\text{Parallel structure : } \quad OUT = \left( kp + \alpha \times \frac{1}{ti \times p} + \alpha \times \frac{td \times p}{1 + \left( \frac{td}{kd} \right) \times p} \right) \times IN$$

$$\text{where } \alpha = \text{scale factor} = \frac{out\_sup - out\_inf}{pv\_sup - pv\_inf}$$

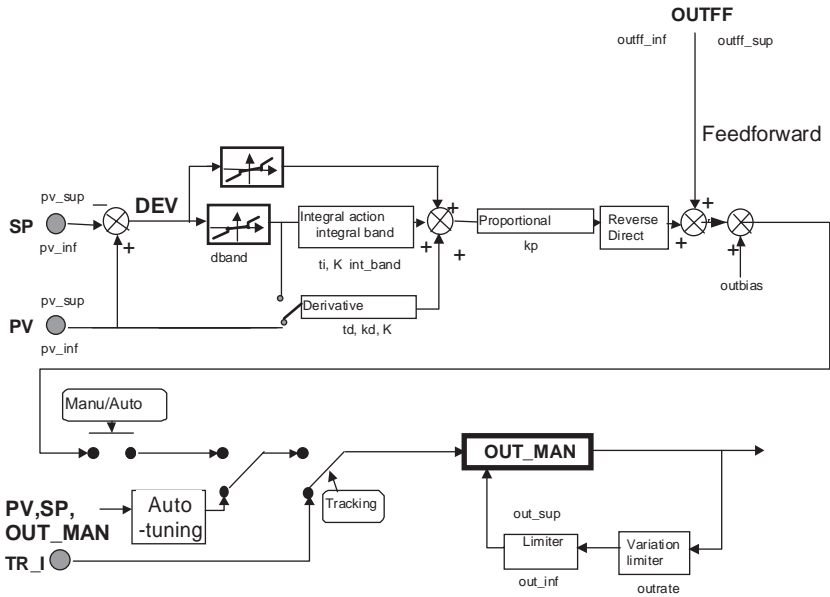
Block diagram of parallel PID



This diagram illustrates the principle of parallel PID. It does not represent implementation of the algorithm in incremental form.



Block diagram of mixed PID



This diagram illustrates the principle of mixed PID. It does not represent implementation of the algorithm in incremental form.

Description of the configuration parameters

Mixed/Parallel structure :

- When it is a mixed type (default type), the proportional action is applied downstream of the integral and derivative actions. The gain K applied to them (see block diagram) is thus equal to  $k_p$ .
- When the structure is parallel, the proportional action is applied in parallel with the integral and derivative actions. In this case, gain  $k_p$  is not applied to the integral and derivative actions. The gain K is thus simply equal to the ratio of the output scale to the process value scale.

Inverse action / Direct action :

The direction of the PID controller can be adapted to that of the actuator / process pair. The action can be defined as inverse (default direction) or direct.

If the action is direct, a positive deviation (PV-SP) increases the output.

If the action is inverse, a negative deviation (PV-SP) decreases the output.

Derivative action :

Derivative action can be used either on the process value or on the deviation.

Smooth manu/auto switching with the absolute form of the algorithm :

see the section on operating modes.

Description of the tuning parameters

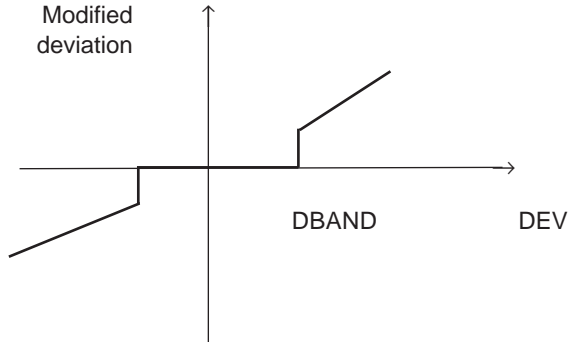
- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Lower limit of the process value scale (phy)	PV_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Upper limit of the process value scale (phy)	PV_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W
Lower limit of the output scale (phy or %)	OUT_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Upper limit of the output scale (phy or %)	OUT_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W

Proportional gain ( $\geq 0$ )	KP	real	0 / 3.4 E38	1.0	R/W
Integral time ( $\geq 0$ ) (s)	TI	real	0 / 3.4 E38	0.0	R/W
Derivative time ( $\geq 0$ ) (s)	TD	real	0 / 3.4 E38	0.0	R/W
Derivative gain ( $\geq 1$ )	KD	real	1 / 3.4 E38	10.0	R/W
Dead band on the deviation (phy)	DBAND	real	0 / 3.4 E38	0.0	R/W
Manual compensation of static deviation (phy or %)	OUTBIAS	real	-3.4 E38 / 3.4 E38	0.0	R/W
Output variation limit, in units per second ( $\geq 0$ ) (phy/s)	OUTRATE	real	0 / 3.4 E38	0.0	R/W
Integral band ( $\geq 0$ ) (phy)	INT_BAND	real	0 / 3.4 E38	0.0	R/W

Dead band on the deviation (DBAND)

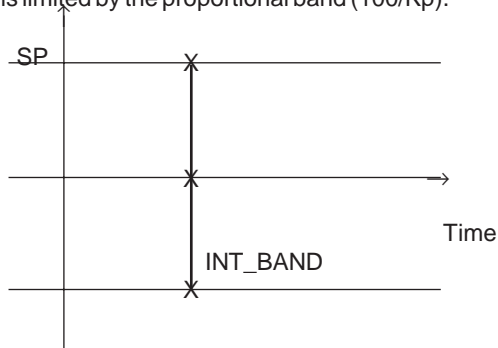
The dead band is used, once it is at the point of operation, to limit small correction jerks at the actuator : as long as the deviation remains below DBAND (in absolute value), the block considers it to be zero for its calculations.



Integral band (INT\_BAND)

The integral band defines a zone around the setpoint in which the integral action is calculated. When the process value-setpoint deviation is greater than this band, the integral action is frozen.

The integral band is limited by the proportional band ( $100/K_p$ ).



P, D or PD controller, BIAS on the command

If integral action is not used ( $T_i = 0$ ), using BIAS on the PID command (OUTBIAS) provides precision at the operating point.

However if  $T_i > 0$ , OUTBIAS is 0 and cannot be modified.

Operating modes and associated commands

For further details, refer to section 5 which describes the operating modes of the loop controller.

---

## Detailed equations

C

The following equations use different variables and functions. The variables corresponding to the block parameters are not described again here.

However, the main intermediate variables and the functions used are described below :

- TermP = value of the proportional action
- TermI = value of the integral action
- TermD = value of the derivative action
- OUTFF = value of the Feedforward action (compensation for disturbance)
- (new) indicates a value calculated during the current execution of the block
- (old) indicates a value calculated during the previous execution of the block
- K : gain of the integral and derivative actions. This gain varies according to the structure of the block (mixed or parallel), and the presence of proportional action :
  - If the structure is mixed and  $k_p \neq 0$ , then  
$$K = k_p$$
  - If the structure is parallel or  $k_p = 0$ , then

$$K = \alpha = \text{scale factor} = \frac{\text{out\_sup} - \text{out\_inf}}{\text{pv\_sup} - \text{pv\_inf}}$$

- VAR : variable used in the derivative action formula. Its value depends on the "derivative action" parameter:
  - VAR = PV If the derivative action is on the process value
  - VAR = DEV If the derivative action is on the deviation.
- direction :
  - is +1 if the action is direct. A positive deviation (PV-SP) increases the output.
  - is -1 if the action is inverse. A positive deviation (PV-SP) decreases the output.
- T\_ECH : sampling period
- $\Delta$  function :  $\Delta(x(t)) = x(t) - x(t-1)$
- limiter function : function limiting the block output.
- **If  $T_i = 0$** , the absolute form of the algorithm is used, and the loop controllers are P or PD type :
  - OUT = TermP + TermD + TermFF + outbias
  - OUTD = OUTP(new) - OUTP(old)
  - OUT = limiter(OUT)

$$\text{TermP} = \text{direction} * k_p * \text{dev}$$



$$\text{TermD} = \frac{\text{td} \cdot \text{TermD}(\text{old}) + \text{direction} \cdot K \cdot \text{td} \cdot \text{kd} \cdot (\text{VAR}(\text{new}) - \text{VAR}(\text{old}))}{\text{kd} \cdot \text{dt} + \text{td}}$$

- If  $T_i \neq 0$ , the incremental form of the algorithm is used, and the loop controllers are PID type :

$$\text{OUTD} = \Delta \text{TermP} + \text{TermI} + \Delta \text{TermD} + \Delta \text{OUT\_FF}$$

$$\text{OUT} = \text{OUT}(\text{old}) + \text{OUTD}(\text{new}) \quad \text{default mode}$$

$$\text{OUT} = \text{RCPY} + \text{OUTD}(\text{new}) \quad \text{actuator in position feedback mode (RCPY).}$$

This mode is used in certain special cases where the position of the actuator may be different from the calculated PID output (see SERVO output, cascaded loop and autoselective loop).

$$\text{OUT} = \text{limiter}(\text{OUT})$$

$$\Delta \text{TermP} = \text{direction} * k_p * [\Delta (\text{DEV})]$$

$$\Delta \text{TermD} = \Delta \cdot \left[ \frac{\text{td} \cdot \text{TermD}(\text{old}) + \text{direction} \cdot K \cdot \text{td} \cdot \text{kd} \cdot (\text{VAR}(\text{new}) - \text{VAR}(\text{old}))}{\text{kd} \cdot \text{dt} + \text{td}} \right]$$

The integral anti-saturation mechanism is implicit in the algorithm.

The controller can operate in pure integral mode ( $k_p = 0$ ). In this case the equations are the following :

$$\text{OUTD} = \text{TermI} + \text{OUTFF}$$

$$\text{OUT} = \text{OUT}(\text{old}) + \text{OUTD}(\text{new}) \quad \text{default mode}$$

$$\text{OUT} = \text{RCPY} + \text{OUTD}(\text{new}) \quad \text{actuator in position feedback mode (RCPY)}$$

$$\text{OUT} = \text{limiter}(\text{OUT})$$

$$\text{TermI} = \text{direction} * \alpha * T\_ECH/TI * \text{DEV}$$

### Execution check

An error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters
- A problem occurs in a floating point calculation
- The output scale is inconsistent on a PLC cold start ( $\text{OUT\_INF} \geq \text{OUT\_SUP}$ )

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

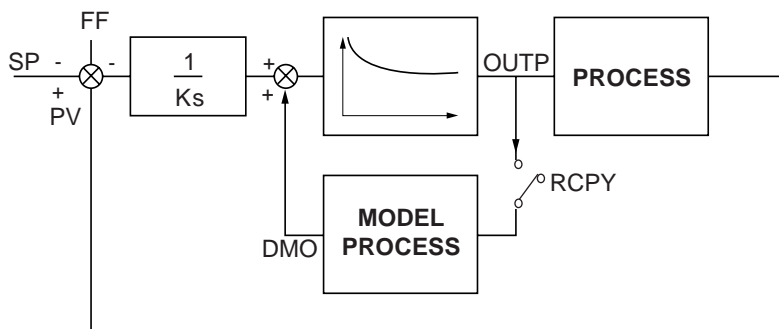
## The "Model-based controller" function

The model-based controller is used for processing in situations where there are significant delay times in relation to the main time constant of the process, where standard PID control is not suitable. It is also useful for controlling non-linear processes.

The model is a first order + delay. However, being intrinsically robust, the loop controller can process any stable aperiodic process of any order. The parameters to be provided are :

- the static gain (ratio of process value delta / command delta in open loop),
- the equivalent time constant (response time / 3),
- the value of the apparent pure delay of the process (estimated),
- Ratio of the open loop time constant / closed loop time constant.

The simplified schematic of the algorithm for the model-based controller is as follows :



The use of this loop controller is similar to that of a PID controller. If only the KP, TI and TD parameters are to be adjusted, the PID controller is replaced by open or closed loop adjustment of the gain, the time constant, the pure delay of the process model and the ratio of the time constants in open and closed loop.

The model-based controller has the same I/O as a PID controller (PV, RSP, FF, OUTP). In addition it has the optional input RCPY (input external to the model) which, when it can be accessed, is used to make the actual process input the model input (for example, flow rate measured at a valve output).

### Functions

Apart from the command calculation, the functions are the same as for PIDFF.

- Direct or inverse action,
- Feedforward action to compensate for disturbance,
- Dead band on the deviation,
- Input external to the model,

- 
- High and low limits of the output signal,
  - Output gradient limit,
  - Selection of Manual / Automatic operating mode,
  - Tracking mode,
  - Autotuning of the main coefficients.

### Managing the delay

In processes to which this loop controller applies, the delay is either :

- variable (transfer of material according to the flow rate in a circuit, speed of the conveyor belt)
- very long.

Both cases will be processed using a buffer whose size can be configured. Depending on its size, it will be possible to sample either all the sampling periods, or one in two, or one in three, etc.

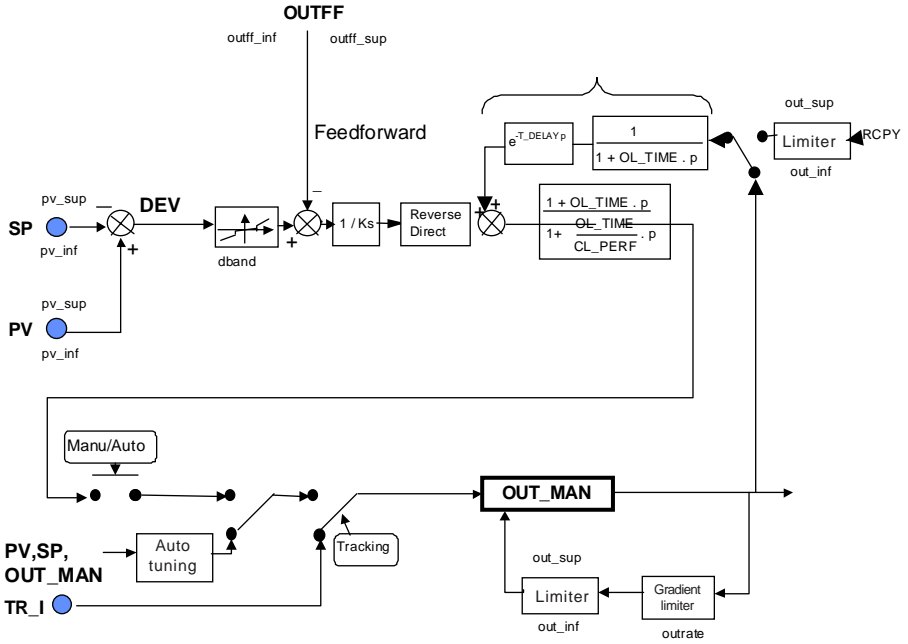
It is possible to increase or decrease the delay  $T\_DELAY$  while the program is running. The new delay is applied instantly, as long as it is compatible with the size of the buffer. The delay sampling period is unchanged. If the value of  $T\_DELAY$  becomes too high in relation to the size of the buffer, it is no longer possible to store sufficient input values to reach the required delay if sampling continues with the same period. The delay sampling period is therefore recalculated and the output is only valid after a period equal to the new delay. To avoid this problem, it is advisable to take account of possible increases in the delay  $T\_DELAY$  when sizing the buffer.

If the delay is decreased, by default the sampling does not change. However, there is a command for recalculating the sampling if necessary.

In the event of dynamic modification of the task time or the sampling period, the output is only valid after a period equal to the delay.

Block diagram

C



Description of the Inverse action / Direct action configuration parameter

The direction of the model-based controller is adapted to that of the actuator / process pair. The action is defined as inverse direction (default direction) and can be redefined as direct.

Description of the tuning parameters

- Internal parameters for the function

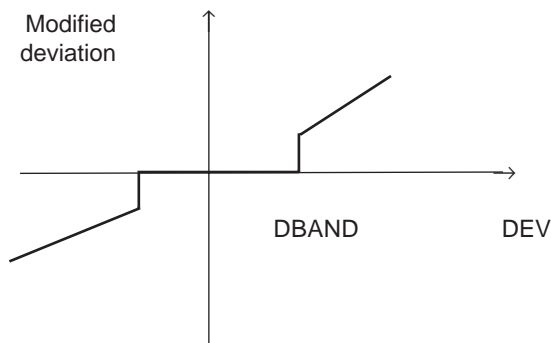
Meaning	Symbolization	Data type	Variation range	Default value	R/W
Lower limit of the output scale	OUT_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Upper limit of the output scale	OUT_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W
Open loop static gain of the process	KS	real	0 / 3.4 E38	1.0 (*)	R/W
Open loop time constant of the process	OL_TIME	real	0 / 3.4 E38	1.0 (*)	R/W
Ratio of the natural (open loop) / required (closed loop) time constants	CL_PERF	real	0.1 / 3.4 E38	1.0	R/W
Current pure time delay of the process(es)	T_DELAY	real	0 / 3.4 E38	0.0	R/W
Dead band on the deviation (phy)	DBAND	real	0 / 3.4 E38	0.0	R/W

(\*) KS and OL\_TIME cannot be 0 (inconsistent value). They will be forced to 1.0.

---

### Dead band on the deviation (DBAND)

This uses the same principle as the PID controller. The dead band is used, once it is at the point of operation, to limit small correction jerks at the actuator : as long as the deviation remains below DBAND (in absolute value), the block considers it to be zero for its calculations.



### Output parameters

The value of command OUT\_MAN is displayed at the output. The value of the delayed output of the DMO model can also be displayed.

### Operating modes and associated commands

For further details, refer to section 5 which describes the operating modes of the loop controller.

### Limitations

Integrating processes are not handled by the model-based controller.

A servomotor without output feedback cannot be used, as the model-based controller does not use an incremental algorithm (the value of the command is calculated then the command variation).

### Execution check

An error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters,
- A problem occurs in a floating point calculation,
- The output scale is inconsistent on a PLC cold start ( $OUT\_INF \geq OUT\_SUP$ ).

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

## The "Autotuning" function

Autotuning saves time when starting an installation by ensuring stable tuning.

The algorithm is based on a Ziegler-Nichols type method. There is initially an analysis lasting 2.5 times the response time of the open loops, making it possible to identify the process as first order with delay.

Then, using this model, a set of tuning parameters is calculated using heuristic, tried and tested rules.

The range of parameters determined is modulated by a performance criterion, so that the response times to disturbance or stability are given priority.

The algorithm handles the following types of process :

- single input / single output processes
- naturally stable or integrating processes
- asymmetrical processes within the limit tolerated by the PID algorithm.

### Main functions

Estimation of tuning of the PID controller (KP, TI, TD) or the model-based controller (KS, T1, T\_DELAY)

Provision of diagnostics

Setting the dynamic range parameters for the proposed tuning

Restoration of the previous settings.

### Internal parameters for the function

Meaning	Symbolization	Datatype	Variation range	Default value	R/W
Amplitude of the control step function (%)	AT_STEP	real	-100 / 100	10.0	R/W
Duration of the step function (s)	AT_TMAX	real	4 / 3.4 E38	100.0	R/W
Autotuning performance criterion	AT_PERF	real	0 / 1	0.5	R/W
Proportional gain	KP	real	0 / 3.4 E38	1.0	R/W
Integral time (s) (*)	TI	real	0 / 3.4 E38	0.0	R/W
Derivative time (s) (**)	TD	real	0 / 3.4 E38	0.0	R/W
Model gain (***)	KS	real	0 / 3.4 E38	1.0	R/W

Model time constant (s) (***)T1	real	0 / 3.4 E38	0.0	R/W
Model delay (s) (***) T_DELAY	real	0 / 3.4 E38	0.0	R/W

(\*): Depending on the type of controller used (PID or model-based controller)

(\*\*): See PID function for further details

(\*\*\*): See model-based controller function for more details

The "Variation range" and "Default value" columns are not applicable for the following types of output parameter. For ease of reading, these columns have been left out of the table.

- Output parameters for the function

Meaning	Symbolization	Data type	R/W
Value before autotuning of the model proportional / gain coefficient	KP_PREV	real	R
Value before autotuning of the model integral / time constant coefficient	TI_PREV	real	R
Value before autotuning of the model derivative / delay coefficient	TD_PREV	real	R
Autotuning in progress	STS_AT_RUNNING	bit	R
Autotuning failed	AT_FAILED	bit	R
Autotuning diagnostics aborted	AT_ABORTED	bit	R
Autotuning diagnostics parameter error	AT_ERR_PARAM	bit	R
Autotuning diagnostics system error or power outage	AT_ERR_PWF_OR_SYS_FAILURE	bit	R
Autotuning diagnostics saturation of the process value or the actuator	AT_ERR_SATUR	bit	R
Autotuning diagnostics process value deviation too small	AT_ERR_DV_TOO_SMALL	bit	R



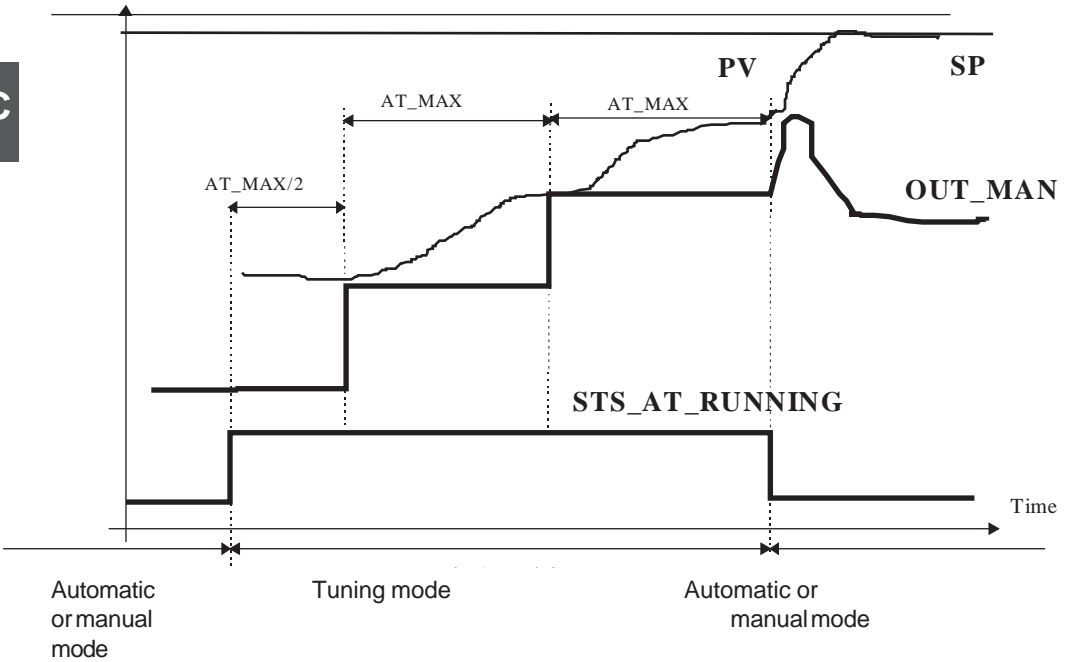
Autotuning diagnostics sampling period too long	AT_ERR_TSAMP_HIGH	bit	R
Autotuning diagnostics inconsistent response	AT_ERR_INCONSISTENT_RESPONSE	bit	R
Autotuning diagnostics process value not initially stable	AT_ERR_NOT_STAB_INIT	bit	R
Autotuning diagnostics duration of the step function too short	AT_ERR_TMAX_TOO_SMALL	bit	R
Autotuning diagnostics process value noise too high	AT_ERR_NOISE_TOO_HIGH	bit	R
Autotuning diagnostics duration of the step function too long	AT_ERR_TMAX_TOO_HIGH	bit	R
Autotuning diagnostics overshoot > 10%	AT_WARN_OVERSHOOT	bit	R
Autotuning diagnostics undershoot too great	AT_WARN_UNDERSHOOT	bit	R
Autotuning diagnostics process too unsymmetrical	AT_WARN_UNSYMETRICAL_PLANT	bit	R
Autotuning diagnostics integrating process	AT_WARN_INTEGRATING_PLANT	bit	R

There are two possible types of autotuning : warm autotuning, cold autotuning.

The first stage of the autotuning sequence is the same for both types : it is a test of the noise and stability of the process lasting  $0.5 * t_{max}$ , during which the outputs remain constant. The next stages depend on the type of autotuning. The choice is made automatically by the algorithm.

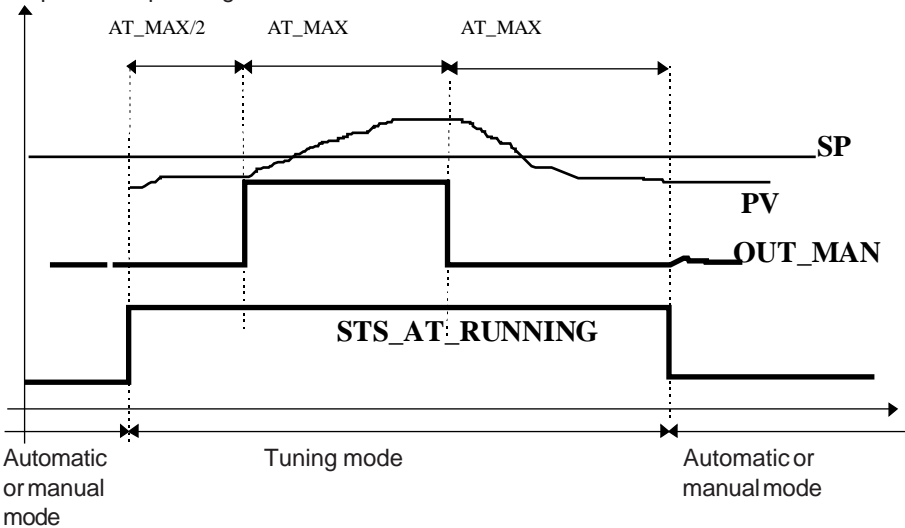
### Cold autotuning

Cold autotuning is performed if the process value/setpoint deviation exceeds 40% and if the process value is less than 30%. Two step functions in the same direction are then applied to the loop controller output (OUT\_MAN). Each step function has a duration of AT\_TMAX. When autotuning is completed, the loop returns to the previous operating mode :



Warm autotuning

If the conditions for cold autotuning are not met, warm autotuning is performed : a step function is applied at the controller output (OUT\_MAN), then an inverse step function. Each step has a duration of AT\_MAX. When autotuning is completed, the loop returns to the previous operating mode :



In both modes, if autotuning fails, the loop controller output is reset to its value before the autotuning operation was launched.

The identification process is divided into 3 steps :

- an analysis of the noise and stability of the process
- a first analysis of the response to a step function providing a first identification model : from this first estimation, a filter is calculated and used for the last step
- a second analysis of the response to a second step function is refined using the data filter.

Finally, a model of the whole process is obtained. If there is too great a difference between the results of the two steps, the estimation is rejected and autotuning fails.

After each of the two steps, a set of parameters is calculated for the loop controller to be autotuned. The equations giving the loop controller parameters are based on the gain and the ratio between the response time and the process delay.

From the point of view of robustness, the algorithm must be capable of withstanding changes of gain and time constant in a ratio of 2:1, without losing its stability. Asymmetrical processes are allowed as long as they keep within this restriction. If they do not, an error is indicated via the diagnostics.

Parameter setting

Setting the parameters of the step function

During autotuning two step functions are applied to the output. A step function is characterized by two parameters : the duration of this step function (AT\_TMAX) and its amplitude (AT\_STEP).

The variation ranges of these parameters are : AT\_TMAX must be higher than 4 seconds and AT\_STEP must be greater than 1 % of the output scale (OUT\_INF, OUT\_SUP).

The function also checks that the output does not exceed the limits of the output scale. The check is performed when autotuning is started.

For information purposes, the following table gives the value of the parameters for some standard types of process control :

Parameters

TYPE OF SCHEME	AT_MAX (s)	AT_STEP (%)
Liquid flow rate or pressure	5-30	10-20
Gas pressure	60-300	10-20
Level	120-600	20
Steam temperature or pressure	600-3600	30-50
Composition	600-3600	30-50

---

Performance criterion : AT\_PERF

Loop controller tuning can be modulated according to the value of the performance criterion.

The parameter AT\_PERF varies between 0 and 1 which enables priority to be given to stability for AT\_PERF close to 0 or to obtain more dynamic tuning (and thus to optimize the response time to disturbance) by making AT\_PERF tend towards 1.

### Operating modes

Various commands are used to control the autotuning function :

- Start autotuning (%MWxy.i.11 = 16#000E)  
Sending this command starts the autotuning process. This command can be sent directly from the autotuning function grid.
- Stop autotuning (%MWxy.i.11 = 16#000F)  
This command enables the user to stop the autotuning process. In this case, the PID parameters are not modified. A diagnosis is given.
- Return to previous setting (%MWxy.i.11 = 16#0010)  
Sending this command performs a swap between the current parameters of the loop controller and the previous parameters (KP\_PREV, TI\_PREV, TD\_PREV). This command is refused if autotuning is in progress.

During autotuning, PID can be in automatic or manual mode.

When autotuning starts, PID changes to autotuning mode and the output retains the last set or calculated value.

At the end of autotuning :

- If it has been successful, the loop is returned to its previous mode (auto or manu)
- If it has not been successful, the output is reinitialized to its initial value (before the start of autotuning), the settings are unchanged and the loop restarts in its previous mode.

The direction of action of the loop controller is checked and compared with the sign of the model gain.

If there is any incompatibility, an error is indicated.

For more detailed information on priority with the other loop controller operating modes, please refer to section 5.

### Diagnostics

Autotuning may not start for various reasons, or it may be aborted during execution or may fail, optionally providing a set of parameters, depending on the cause of the failure. The various situations are listed below.

**Acknowledgment:** There is a function for acknowledging the diagnostics message. It can be accessed from the PL7 screen or via the acknowledgment commands.

### Causes of non-starting

- Parameter error (Bit 2 : AT\_ERR\_PARAM).

Possible causes are as follows :

- step function too short ( $AT\_MAX < 4$  s)
- amplitude too low ( $AT\_STEP < 1\%$  of the output scale)
- protocol not possible : if the current output +  $n \times$  amplitude of the step function ( $n = 1$  for warm autotuning, and  $n = 2$  for cold autotuning) is outside the output scale (OUT\_INF, OUT\_SUP), it will not be possible to apply the test protocol. STEP\_AMPL must be set to a value which is compatible with the current operating point.

- Incorrect sampling period (Bit 6 : AT\_ERR\_TSAMP\_HIGH).

If the sampling period is too long in relation to the duration of the step function (greater than  $AT\_MAX / 25$ ), reading the response will not be sufficiently precise and autotuning is inhibited. This situation is specific to very fast control operations ( $AT\_MAX$  increasing the process stabilization time by several seconds),  $t_{max}$  can therefore be increased as the algorithm is not very sensitive to this parameter (in a ratio of 1 to 3) or the sampling time can be altered.

### Causes of autotuning being aborted

- Stop following a system fault (Bit 3 : AT\_ERR\_PWF\_OR\_SYS\_FAILURE).

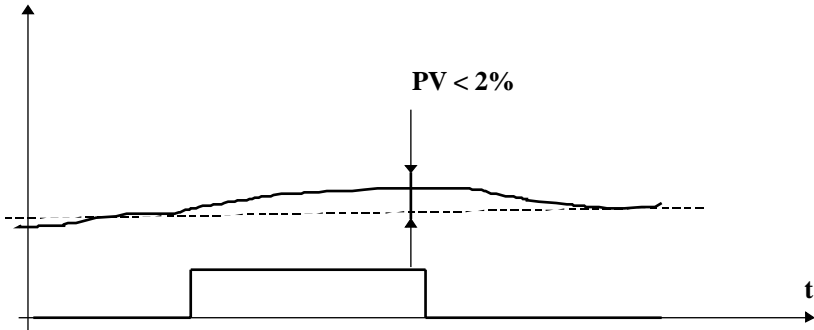
Autotuning is aborted if a PLC system event occurs which prevents the sequence being run completely. For example powering down will automatically stop the function when the power returns.

- Saturation of the process value (Bit 4 : AT\_ERR\_SATUR).

If the process value undershoots or overshoots the full scale range (PV\_INF, PV\_SUP), autotuning is aborted and the loop controller returns to the previous mode. A prediction of the future process value can even be used to stop autotuning before the over/undershoot occurs (when a first model has been identified).

- Insufficient variation (Bit 5 : AT\_ERR\_DV\_TOO\_SMALL).

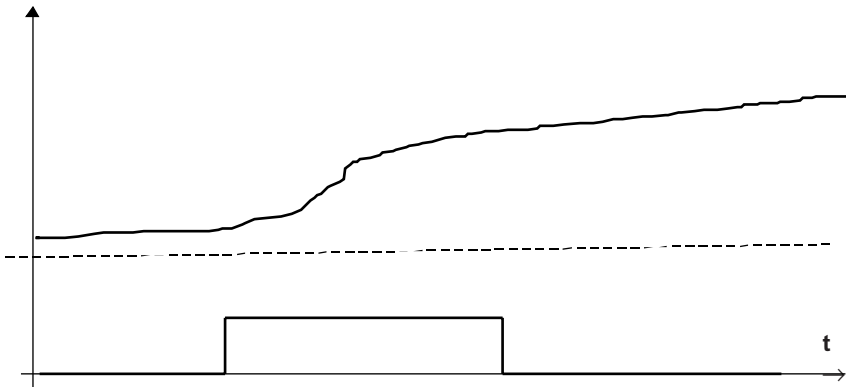
PV



The amplitude of the step function is too low to cause a significant reaction in the process. AT\_STEP can then be increased.

- Inconsistent response (Bit 7 : AT\_ERR\_INCONSISTENT\_RESPONSE).

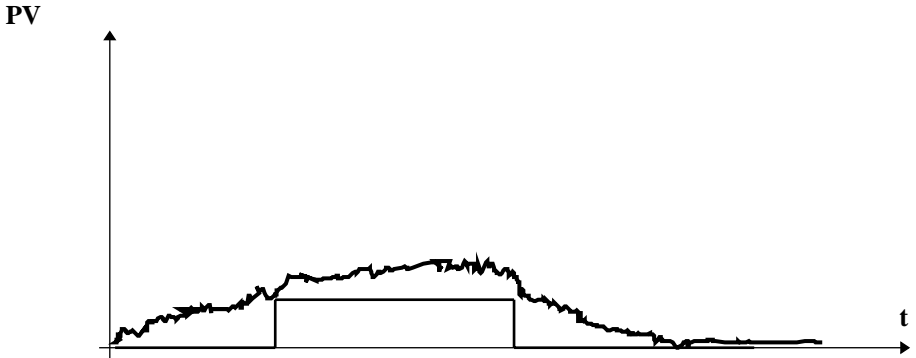
PV



The process response is not consistent (gains with different signs). This may be the result of a high level of disturbance, or coupling with other loops, etc.

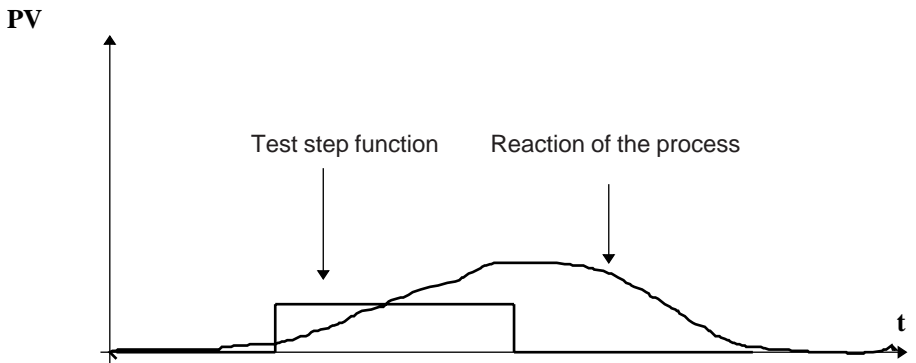
Autotuning has finished and a diagnostic report is generated.

- Noise too high (Bit 10 : AT\_ERR\_NOISE\_TOO\_HIGH).



The reaction of the process to the step function is not enough in relation to the noise. Filter the process value or increase AT\_STEP.

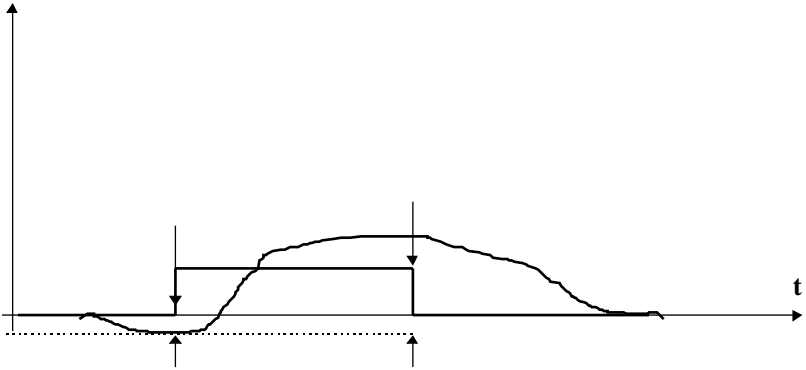
- Duration of the step function (AT\_MAX) too short (Bit 9 : AT\_ERR\_TMAX\_TOO\_SMALL).



The response is not stabilized before the return to the initial command. The calculated parameters are therefore invalidated.

- Process value not initially stabilized (Bit 8 : AT\_ERR\_NOT\_STAB\_INIT).

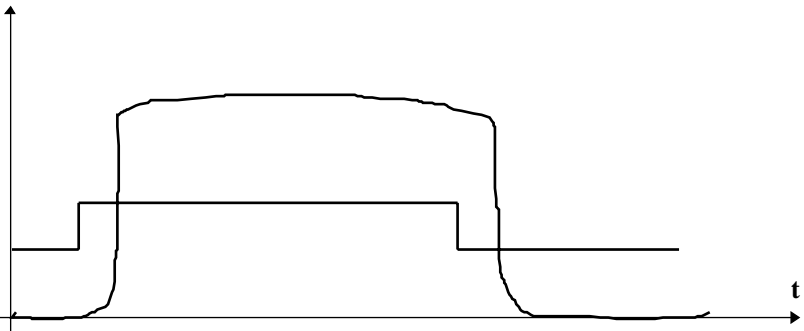
PV



Autotuning is started before the process value has been stabilized. If the process value variation is high in relation to the reaction to the step function, the results of the test will be invalidated.

- Duration of the step function (AT\_TMAX) too long (Bit 11 : AT\_ERR\_TMAX\_TOO\_HIGH).

PV

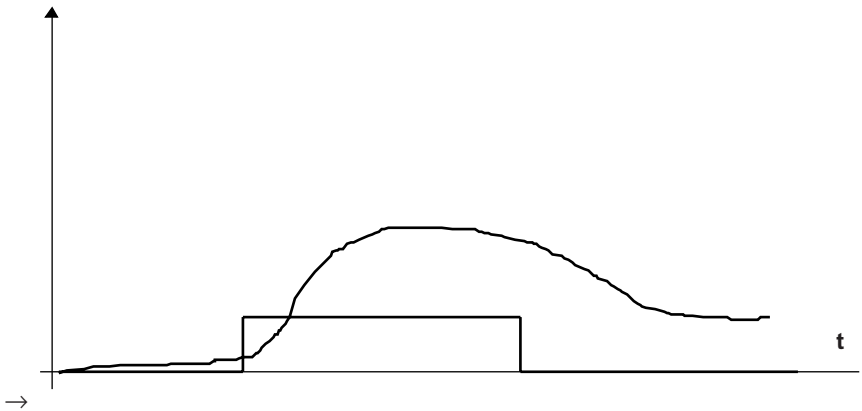


AT\_MAX determines the frequency with which the process values which will be used for calculating the coefficients are taken into account. AT\_TMAX must be between 1 and 5 times the process rise time.



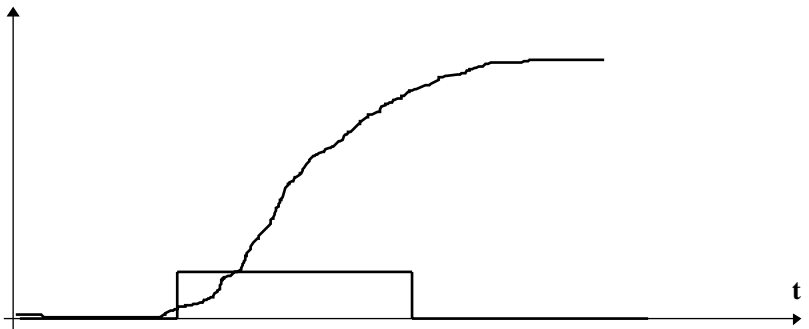
- Process with very high overshoot (Bit 12 : AT\_WARN\_OVERSHOOT).  
This bit is set if the reaction to a command step function triggers too great a process value overshoot (more than 10%). The process does not correspond to the models handled by the algorithm.
- Process with too great an undershoot (Bit 13 : AT\_WARN\_UNDERSHOOT).  
This bit is set if the reaction to a command step function triggers a reversal of the response in its initial phase (undershoot greater than 10%). The process does not correspond to the models handled by the algorithm.
- Non-symmetrical process (Bit 14 : AT\_WARN\_UNSYMMETRICAL\_PLANT).

**PV**



- Integrating process (Bit 15 : AT\_WARN\_INTEGRATING\_PLANT).  
Either the process is integrating, or AT\_MAX is too low and the process is non-symmetrical. The calculated coefficients correspond to the integrating process. If this is not the case, increase AT\_MAX and restart autotuning.

**PV**



## The "Split range" function

This function is useful where two actuators are used to cover the whole of the control range. It is placed downstream of the loop controller.

The split range function also has the following functions :

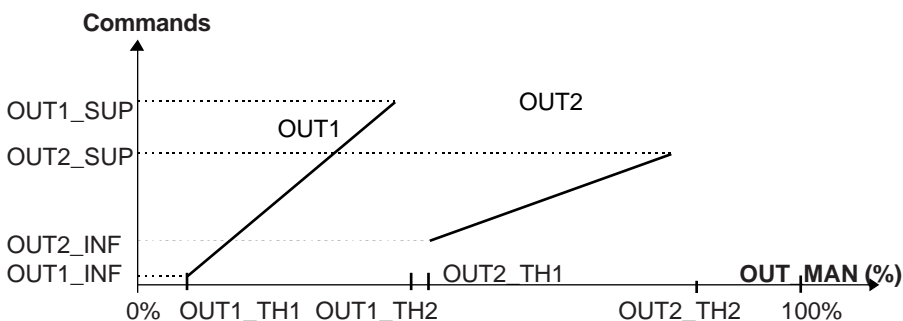
- The function handles overlaps and dead bands between the two actuators,
- It provides one manual command and one manual instruction (similar to a single PID).

It is used to control analog outputs and servomotors with feedback.

It cannot be used to control servomotors without feedback.

When the split range function is used, the output scale of the loop controller must be (0, 100).

Setting the parameters of the function consists of defining the characteristics of each actuator, ie. the way in which the two outputs must vary between the two thresholds. The value of the output varies in a linear manner. Outside these two thresholds, the output is limited to defined thresholds.



where  $OUT_i\_TH_j$ : threshold  $j$  of output  $i$

### Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	0 / 100	-	R/W

Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of OUT1 for OUT_MAN = OUT1_TH1	OUT1_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Value of OUT1 for OUT_MAN = OUT1_TH2	OUT1_SUP	real	-3.4 E38 / 3.4 E38	1000.0	R/W
Value of OUT2 for OUT_MAN = OUT2_TH1	OUT2_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Value of OUT2 for OUT_MAN = OUT2_TH2	OUT2_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W
value of the input for which OUT1 = OUT1_INF	OUT1_TH1	real	0 / 100	0	R/W
value of the input for which OUT1 = OUT1_SUP	OUT1_TH2	real	0 / 100	50	R/W
value of the input for which OUT1 = OUT2_INF	OUT2_TH1	real	0 / 100	50	R/W
value of the input for which OUT1 = OUT2_SUP	OUT2_TH2	real	0 / 100	100	R/W

Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Limitation of the variation of output 1 in units / s	OUTRATE	real	0 / 3.4 E38	0.0	R/W
Limitation of the variation of output 1 in units / s	OUTRATE2	real	0 / 3.4 E38	0.0	R/W

---

### Execution check

An execution error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters,
- A problem occurs in a floating point calculation.

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

A warning is given if thresholds OUT1\_TH1, OUT1\_TH2, OUT2\_TH1 and OUT2\_TH2 are not between 0 and 100%.

**The "Hot/cool" function**

This function is useful where two opposing actuators are used to cover the whole of the control range. It is placed downstream of the loop controller.

The "Hot/cool" function also has the following functions :

- The function handles overlaps and dead bands between the two actuators.
- It provides one manual command and one manual instruction (similar to a single PID).

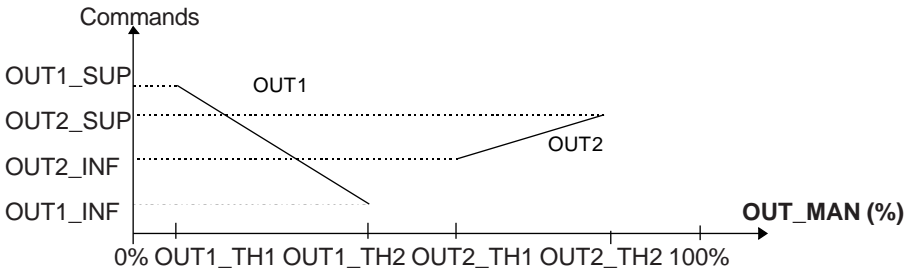
It is used to control analog outputs and servomotors with feedback.

It cannot be used to control servomotors without feedback.

When the hot/cool function is used, the output scale of the loop controller must be (0, 100).

Setting the parameters of the function consists of defining the characteristics of each actuator, ie. the way in which the two outputs must vary between the two thresholds. The value of the output varies in a linear manner. Outside these two thresholds, the output is limited to defined thresholds.

Output 1 handles the "cool", output 2 handles the "hot"



where  $OUT_i\_TH_j$  : threshold j of output i

Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	0 / 100	-	R/W

Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of OUT1 for OUT_MAN = OUT1_TH1	OUT1_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W

Value of OUT1 for OUT_MAN = OUT1_TH2	OUT1_SUP	real	-3.4 E38 / 3.4 E38	1000.0	R/W
Value of OUT2 for OUT_MAN = OUT2_TH1	OUT2_INF	real	-3.4 E38 / 3.4 E38	0.0	R/W
Value of OUT2 for OUT_MAN = OUT2_TH2	OUT2_SUP	real	-3.4 E38 / 3.4 E38	100.0	R/W
value of the input for which OUT1 = OUT1_INF	OUT1_TH1	real	0 / 100	50	R/W
value of the input for which OUT1 = OUT1_SUP	OUT1_TH2	real	0 / 100	0	R/W
value of the input for which OUT1 = OUT2_INF	OUT2_TH1	real	0 / 100	50	R/W
value of the input for which OUT1 = OUT2_SUP	OUT2_TH2	real	0 / 100	100	R/W

### Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Limitation of the variation of output 1 in units / s	OUTRATE	real	0 / 3.4 E38	0.0	R/W
Limitation of the variation of output 1 in units / s	OUTRATE2	real	0 / 3.4 E38	0.0	R/W

### Execution check

An execution error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters
- A problem occurs in a floating point calculation

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

A warning is given if thresholds OUT1\_TH1, OUT1\_TH2, OUT2\_TH1 and OUT2\_TH2 are not between 0 and 100%.

## 2.6-5 Output branch functions

### The "Servo" function

This function performs process control using electric servomotors with or without position feedback. It converts the digital output of the loop controller to generate 2 logic outputs, RAISE and LOWER.

When position feedback is used, the block controls the position of the actuator.

When position feedback is not used, the loop controller and the associated servo function perform "floating point" process control.

When the servo function is used, the output scale of the loop controller must be (0, 100).

#### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	0 / 100	-	R
Value of the command (*)	OUTi	real	0 / 100	-	R
Upper stop	-	bit	-	-	R
Lower stop	-	bit	-	-	R
Position feedback	-	word	0 / 3.4 E38	-	R
Value of the command variation	OUTD	real	-100 / 100	-	R

(\*) For hot/cool or split range

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Opening time (s)	T_MOTORi	real	0 / 3.4 E38	10.0	R/W
Min. time (s)	T_MINIi	real	0 / 3.4 E38	0.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Status of the opening command	STS_RAISEi	bit	-	-	R
Status of the closing command	STS_LOWERi	bit	-	-	R

The servo operates differently depending on whether or not position feedback is used :

- With position feedback (RCPY)

When position feedback is used, for each new value of output OUT\_MAN produced by the loop controller, the SERVO function generates a RAISE or LOWER binary command whose duration is proportional to the deviation between the loop controller command and the position feedback value. This then produces servo control proportional to the position of the actuator.

Note : when the calculated duration exceeds the loop sampling period (in automatic mode) or the task scan time (in the other operating modes), it is not stored for the following scans.

- Without position feedback

For each new command variation value produced by the loop controller, the SERVO generates a RAISE or LOWER binary command whose duration is proportional to the variation of the loop controller output OUTD.

Note : When the calculated duration exceeds the loop sampling period (in automatic mode) or the task scan time (in the other operating modes), the duration remaining to be applied is added to the new duration calculation, which enables it to be processed over several scans.

In this case, the SERVO associated with the loop controller is used to perform floating point process control. The algorithm does not use the absolute output of the loop controller but the output variation. The RAISE output (or LOWER output, depending on the variation sign) is set to 1 for a time which is proportional to the valve opening time (T\_MOTOR), and to the variation value OUTD.

The pulse duration (T\_IMP) to be applied to the output is calculated using the following principle :

An initial theoretical value is given by the formula :

$$T\_IMP = (OUT\_MAN - RCPY) (\%) \times T\_MOTOR \text{ (with feedback)}$$

$$T\_IMP = (T\_IMP + OUTD) (\%) \times T\_MOTOR \text{ (without feedback)}$$

To avoid generating pulses which are too short, the pulses are limited to a minimum duration T\_MINI.



---

When the calculation of the pulse duration gives a value which is less than  $T\_MINI$ , the servo does not generate any pulses, but stores the value for the next calculation. This enables situations where the variations of a loop controller output are small but of long duration to be processed correctly.

#### Actuator opening time ( $T\_MOTOR$ )

This parameter enables the function block to adapt to different servomotors.

The pulse duration to be applied to RAISE or LOWER is proportional to the opening time of the actuator at full scale.

#### Minimum pulse duration ( $T\_MINI$ )

This parameter is used to avoid generating pulses which are too short, which are generally harmful for the actuators.

When the calculated pulse duration to be applied to RAISE or LOWER is less than  $T\_MINI$ , the block does not generate any pulses.

In all cases, any pulse which is started lasts at least  $T\_MINI$ .

#### Execution check

An execution error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters,
- A problem occurs in a floating point calculation.

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

If the time parameters  $T\_MOTOR$  and  $T\_MINI$  are negative, their value is forced to 0.0.

When the loop controller is in manual mode, the controller output  $OUT\_MAN$  also controls the outputs of the servo.

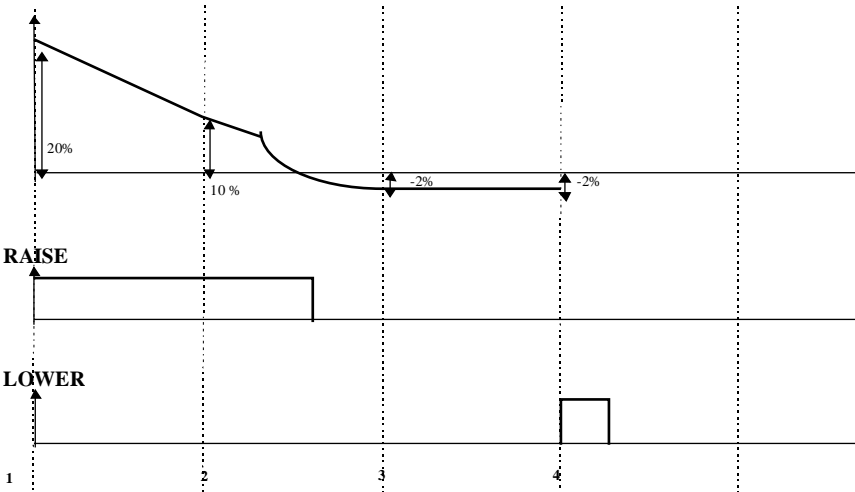
---

## Examples

**C**

Operation in **automatic mode with position feedback** ( $T_{\text{MOTOR}} = 25\text{s}$ ,  $T_{\text{MINI}} = 1\text{s}$  and sampling period = 4s)

### OUT\_MAN-RCPY deviation



1. The deviation OUT\_MAN-RCPY is 20% : a pulse of 5 s (=20% of 25 s) is given at the RAISE output.

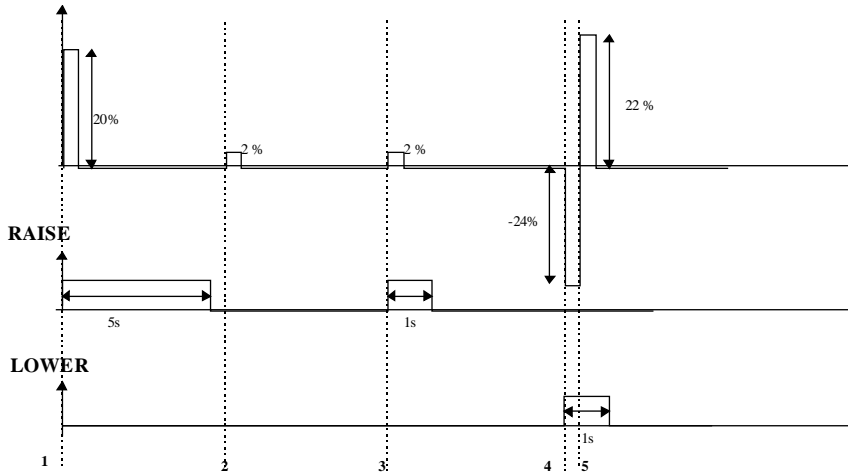
2. The deviation is now only 10%, a pulse of 2.5s (= 10% of 25 s) is given at the RAISE output, without taking account of the second which remained from the previous pulse.

3. The deviation is now -2%, which corresponds to a pulse of 0.5 s on LOWER. As  $T_{\text{MINI}}$  is 1 s, no pulse is generated (however the duration of 0.5 s is stored).

4. The deviation remains at -2%, but the corresponding pulse (0.5 s) is added to the pulse which was stored previously, to reach 1s. This duration is equal to  $T_{\text{MINI}}$ , and the pulse is therefore applied to the LOWER output.

Operation in **automatic mode without position feedback** ( $T_{MOTOR} = 25s$ ,  $T_{MINI} = 1s$ ).

**OUTD**



In this case the command variation value is taken into account each time the SERVO block is executed :

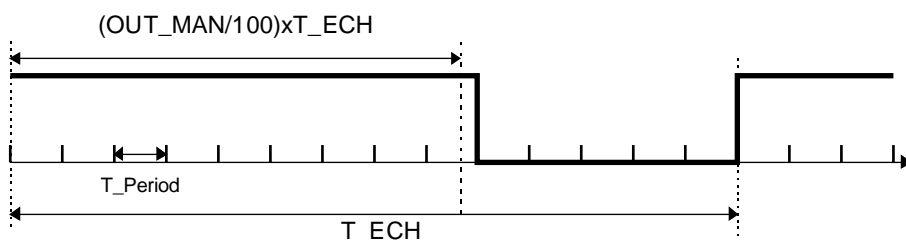
1. The variation of the PID output is +20%, and in this case the pulse affects the RAISE output for a duration of 5 s (= 20% of 25 s).
2. The variation of the PID is +2 %, which would correspond to a pulse of 0.5 s. This pulse is less than  $T_{MINI}$  (=1 s.) and does not affect the outputs.
3. A second variation of +2 % appears, the function adds this variation to the preceding variation (which corresponded to a variation less than the min. value) for its calculation. This then corresponds to a positive global variation of +4 %, and therefore to a pulse of 1 s on the RAISE output.
4. A variation of -24 % appears, and a pulse of 6s is therefore given on the LOWER output
5. Before the next second has elapsed, another variation of + 22 % takes the system to a global variation of 2 % less than the variation of  $T_{MINI}$  (4%). The function stops applying the minimum pulse of 1 s.



## The "PWM" function

This function is used for pulse width modulation control of a discrete actuator. The logic output is set to 1 after a time proportional to the command calculated by the PID and to the given modulation period. The cyclical ratio of such an output is defined as being the rate of activity of the output, ie. the ratio of the time when the output is active to the total time. The cyclical ratio (expressed as a %) of a PWM output is therefore equal to the command calculated by the loop controller (expressed as a %).

When the PWM function is used, the output scale of the loop controller must be (0, 100).



### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	0 / 100	-	R
Value of the command (*)	OUT <sub>i</sub>	real	0 / 100	-	R

(\*) For hot/cool or split range

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Min. time(s)	T_MIN <sub>i</sub>	real	0.0 / 3.4 E38	0.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Status of command	STS_RAISE1	bit	-	-	R

---

The period for the function must be chosen according to the characteristics of the actuator. It is thus logical that it is equal to the loop controller sampling period : the actuator would not be able to accept a command sampled more quickly.

The time base used for the modulation is the MAST or FAST task period. In other words, the smallest possible pulse lasts one task period. The user can however define a longer minimum pulse using the parameter T\_MINI, to observe the requirements of the actuator.

The higher the resolution of the PWM function, the more precise the control achieved. The resolution is defined by the ratio : sampling period / task period. A minimum of 10 : 1 is recommended.

Example :

Sampling period = 2s (chosen according to the characteristics of the actuator).  
The task period must not therefore exceed 200 ms.

#### Execution check

An execution error is indicated in the following cases :

- A non floating point input data item is detected on one of the parameters,
- A problem occurs in a floating point calculation.

In all these cases, the error is considered to be serious, the loop output is frozen, and the status words indicate these faults.

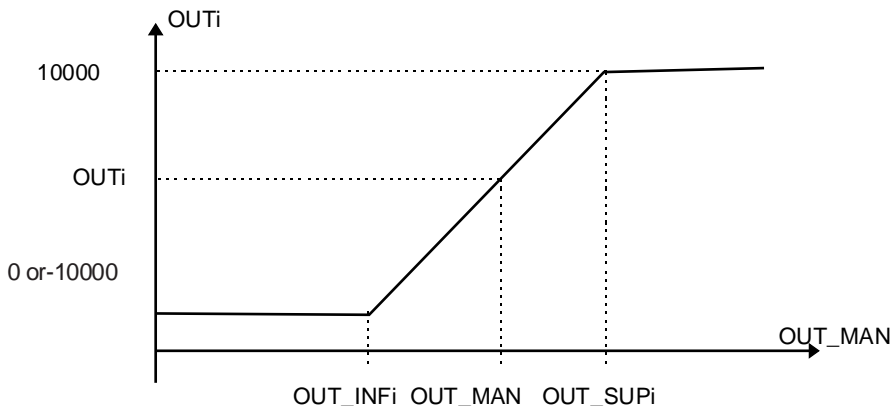
## The output scaling function

This function is used to place the calculated command within the output scale. This function is optional, and is used to choose the scale according to specific outputs. If this function is used, it introduces a scale factor. It performs the following calculation :

$$OUT = (OUT\_MAN - OUT\_INF) \cdot \frac{(OUT\_MAX - OUT\_MIN)}{(OUT\_SUP - OUT\_INF)} + OUT\_MIN$$

where :  $OUT\_MIN = 0$  or  $-10000$

and :  $OUT\_MAX = 10000$



### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	-3.4 E38 / 3.4 E38	-	R
Value of the command (*)	OUTi	real	-3.4 E38 / 3.4 E38	-	R

(\*) For hot/cool or split range

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Low scale (phy)	OUT_INF <sub>i</sub>	real	-3.4 E38 / 3.4 E38	0.0	R/W
High scale (phy)	OUT_SUP <sub>i</sub>	real	-3.4 E38 / 3.4 E38	100.0	R/W

- Output parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	-3.4 E38 / 3.4 E38	-	R

### Execution check

The parameter check for this function is integrated in the output branch error handling.

## The output format function

This function is used to set the value of an analog output.

There are 2 possible formats (ranges) :

- Unipolar : 0 / 10000 (default value)
- Bipolar : -10000 / 10000

### Assigning the output address :

The output address is entered on the graphic part of the PL7 process control configuration screen. It must be a word type variable, ie :

- A %QW of an analog output
- A %MW

### Parameters

- Input parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Value of the command	OUT_MAN	real	-3.4 E38 / 3.4 E38	-	R
Value of the command (*)	OUTi	real	-3.4 E38 / 3.4 E38	-	R

(\*) For hot/cool or split range

- Internal parameters for the function

Meaning	Symbolization	Data type	Variation range	Default value	R/W
Range	-	bit of %KW	-	-	R/W



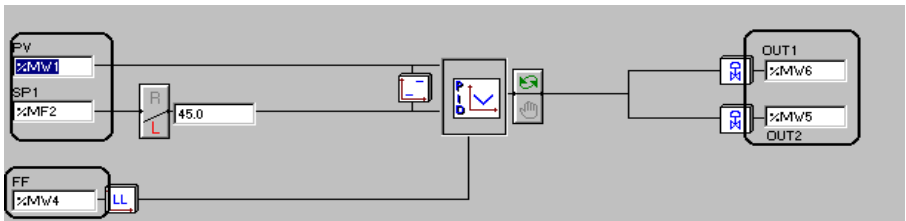
## 2.7 Configuring the I/O

### 2.7-1 Assignment

The configuration parameters for the I/O channels are entered in the application-specific screens associated with these I/O modules.

However, a channel is assigned to the input or the output of a loop schematic in the process control screen using the associated language object (example : %Q4.0).

These parameters are entered in the graphic schematic in the configuration screen :



### 2.7-2 Associated checks and functions

- The I/O module can be configured after configuring the control channels. However, if the interface module has not been configured, the confirmation will fail, generating an error message.
- In the configuration the “ Replace ” function checks that the object is used in the process control parameter settings. A warning message of the same type as the confirmation message is then displayed.
- There is no consistency check between the assignment of the I/O and the control channels. It is advisable to assign all these channels to the same task.

If an I/O module is moved, the address is not automatically modified in the process control parameter screens.

If the language object assigned to an input or output no longer exists, an error message will appear during global validation.

---

### 2.7-3 Types of interface

	Authorized language object	Type
Standard process value input	%IW, %MW	WORD
External process value input	%MF	REAL
Remote1 setpoint input	%MF	REAL
Remote2 setpoint input	%MF	REAL
Feedforward input	%IW, %MW	WORD
Analog output	%QW, %MW	WORD
Servo, PWM output	%Q, %M	EBOOL

---

## 2.8 Symbolization of language objects

Mnemonics already filled in by the language objects of the control loops can be imported into the symbols editor.

This importing is controlled from the data editor. The transfer can be accessed from the %CH of each channel. The "symbol" and "comment" fields are then filled in automatically, but can be modified.

For further details please refer to the general documentation on PL7 application-specific functions.

The suffixes and comments for each language object are given in section 6.

### 3.1 Control loop debug screens

In online mode the process control debug screens are used to :

- Display and animate the loop diagram,
- Display process alarms and channel faults (Diagnostics and Warning),
- Modify the tuning parameters of each function,
- Simulate the input interface values,
- Add / remove / replace calculation functions,
- Modify the configuration parameters of each PID inverse/direct mode function,
- Modify the loop controller operating modes.

The screenshot displays the control loop debug interface for a PMX 57102 processor. The window title is "PMX 57102 [RACK 0 POSITION 0]". The interface is divided into several sections:

- Header:** Includes a "Debug" dropdown menu, a "Configure (MMI)" button, and the designation "PMX P 57102 PROCESSOR".
- Alarms:** A section with a "DL" alarm indicator (red) and buttons for "DIAG..." and "WARNING...".
- LOOP0 Section:**
  - Loop parameters table:**

Loop parameters	
Measurement	Standard
Setpoint	Simple
Loop Controller	Hot/Cold
Feed Forward	Yes
Output 1	Analog
  - Functions list:** A list of functions including Format, Filtering, Fct Generator, Alarms, and Simulation.
  - Parameters table:**

Parameters	
Time constant (s)	10.0
Gain	1.0
Output	4036.995
- Process Flow Diagram:** Shows a control loop with "Loop Scale" (Low: 0.0, High: 100.0), "PV" (5000), "SP1" (63.0), "FF" (3745), and two outputs "OUT1" (1561) and "OUT2" (0).

These screens have the same user interface in online and offline mode. They are directly animated :

- In the module zone, it is possible to toggle between debug mode and configuration mode. In this mode, it is possible to reconfigure the control channel online (see section 3.3 for further details).
- In the channel zone, all the alarms associated with the loops are displayed, if the alarm functions have previously been configured.

---

There are two indicator lamps, "DIAG" and "WARN". DIAG covers all serious faults (%I.ERR=1). "WARN" indicates warning messages concerning the operation of the control loop (example : step function duration too short for autotuning).

When these indicator lamps are on, a window displays the messages associated with error diagnosis. For a detailed description of the diagnostics, please refer to the "branch execution check" sections in Section 2.

**Note** : Each diagnostic has an associated language object bit %MWxy.i:Xj (see section 6). They can be used for global control of the diagnostics.

- Only configured functions are shown in the grid. The values of their associated parameters are animated. They can be modified online from this grid.
- Intermediate calculation values are shown in the diagram (example : the process value at the loop controller input). Auto/Manu and Remote/Local changes can be made by clicking the mouse button. The associated command or setpoint values can be entered directly in the entry fields. Entry fields which are grayed out are inactive.

It is also possible to disconnect the input from the process value and the feedforward to impose a simulation value at the input to these branches. This value is an integer or floating point value (for the external type process value).

To use this function simply activate the simulation function from the process value or feedforward branch functions in the grid. To show that it is simulated, the input value changes to red in the diagram.

If the user exits the process control screens, a message warns that the inputs are in simulated mode.

The simulation value tracks the current value to ensure a smooth change of operation.

---

### 3.2 Modifying the parameters of each loop

---

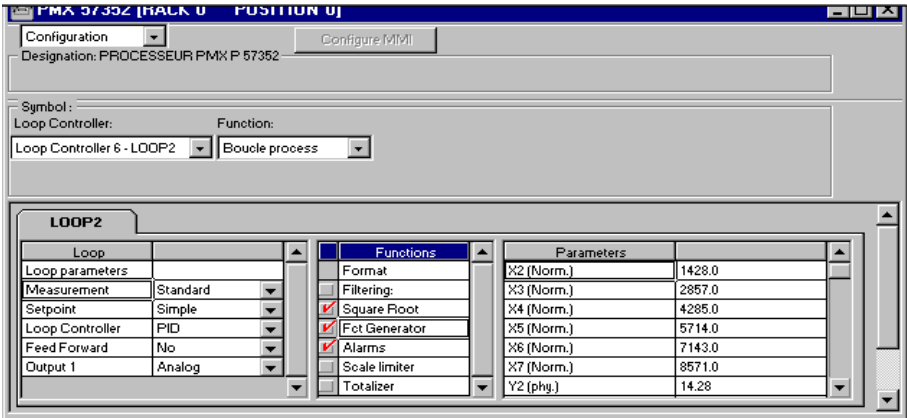
The tuning parameters (example : Kp, Ti) can be modified offline or online by all the PL7 tools, in addition to the process control application-specific screens (example : PL7 program, data editor, dynamic display table, UNITE server, etc).

They do not require global reconfiguration, and they use the save functions (for further details see section 3.6).

### 3.3 Modifying the functions of each loop

Functions can be added to or removed from the control loops in online mode.

Example : Removal of lead-lag, addition of the filtering function in the process value.



These modifications can be performed offline and online, with the PLC in RUN. For safety reasons, modification of these functions (addition, removal, replacement) and the configuration parameters of certain functions such as extrapolation of the Function generator function, peak limiting the limiters, etc, require reconfiguration of the channel. A confirmation message warns of this reinitialization.

The loop restarts in a deterministic state :

- Modification of the process value, setpoint and feedforward branches : the loop performs a warm restart.
- Modification of the loop controller or the output : the loop restarts with the initial operating modes defined at configuration.

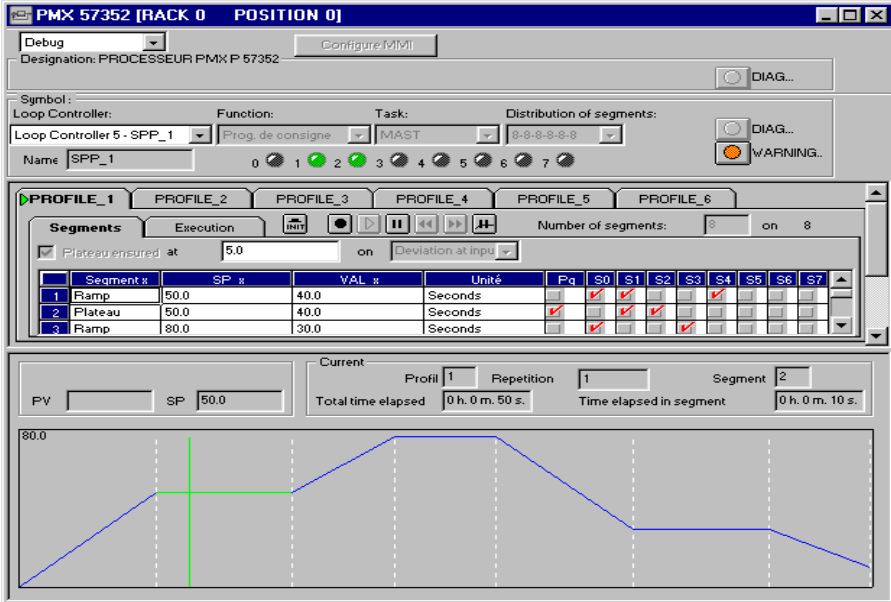
However, addition or modification of I/O addresses or memory word addresses is prohibited in online mode. For this reason, modification of a branch type is not permitted (example : replacement of a simple setpoint with a ratio setpoint).

Note : The totalizing function cannot be added in online mode. Its output is a %MF address.

### 3.4 Debugging the setpoint programmer

C

The “setpoint programmer” channels have their own debug screen. They provide the same functions as the control loop screens.



These screens have all the online reconfiguration and data saving functions. Any reconfiguration in online mode causes operation of the channel concerned to stop and thus stops the setpoint programmer.

Execution of the profile is displayed in realtime on the PL7 screen graphics.

In the same way, the following can be displayed directly in the debug screen :

- the number of the current segment (SEG\_OUT),
- the number of the current iteration (CUR\_ITER),
- the execution times of the current segment (TIME\_SEG),
- the total time (TIME\_TOTAL).

The state of the control outputs is displayed directly in the channel zone. It is possible, using the control button in the tab, to control each profile directly.

The screens operate on the same principle as the process control screens.

---

### 3.5 Optimization of the loop

---

The autotuning function downloaded with the various loop controllers is used to optimize tuning of the control loops.

---

### 3.6 Saving data

---

#### 3.6-1 Saving tuning parameters

Any modification of a tuning parameter from the PL7 process control application-specific screens updates the current value and the initial value of these parameters, in the PLC and in the PC.

Modifying a tuning parameter from the application program or from an animation table (PL7 Junior, Pro, ProDyn) affects the current value but does not change the initial value.

The explicit instruction SAVE-PARAM must be used to save this new value.

A modification from the XBT-F runtime screens affects the current value but does not change the initial value. An explicit save command in these screens can be used to perform this update.

Any operation saving current values to the initial values which were not made by PL7 results in an inconsistency between the PL7 application and the application in the PLC. For further details please refer to the common application-specific documentation.

On a cold restart (%S0) and when loading applications, the current parameters are lost if they have not been saved previously. They are replaced by the initial values.

---

#### 3.6-2 Application backup

Premium PLCs offer the possibility of saving the application (program and constants) to a Backup card. The RAM memory can also be reloaded by the contents of this card. This function is not available if the application is executed on a PCMCIA memory card. The "Load a Backup" and "Restore a Backup" functions are available with PMX CPUs.

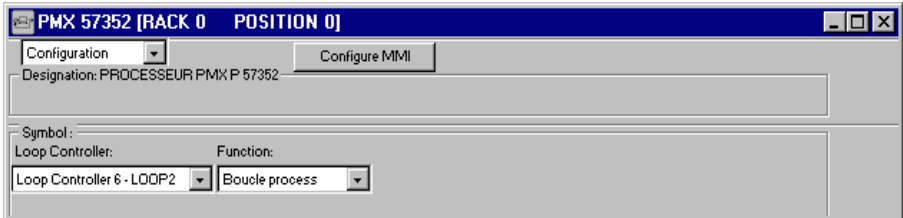




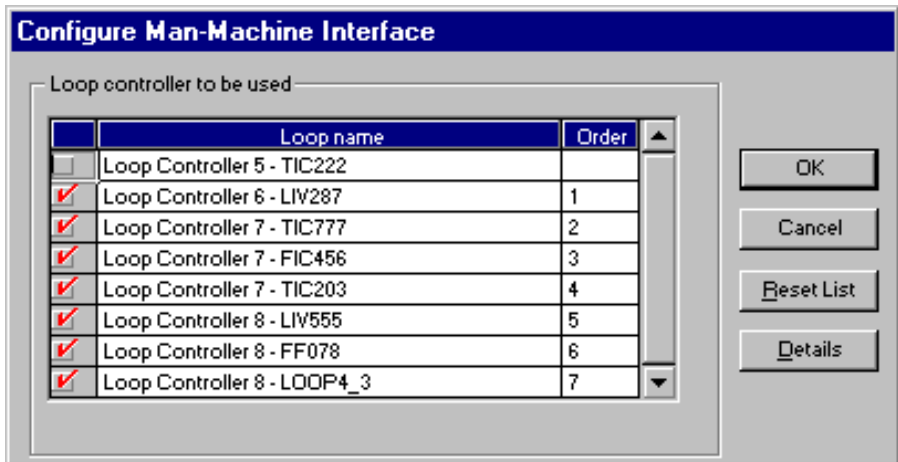
## 4.1 Configuration under PL7

### 4.1-1 Selecting the loops to use

The "Man-Machine Interface" button in the module zone is used to determine, from the configured channels, which control loops will be used by an XBT man-machine interface.



By selecting the various lines, the order of the loops is also determined. The purpose of this is to define the order in which the loops appear on the runtime screens.



The maximum number of loops used by the XBT Magelis is limited to 16.

### 4.1-2 Exchange zones (%MW)

The variables associated with a control loop do not all have the same requirements for exchange with the man-machine interface :

- Exchanges may be contextual depending on the screen displayed (example : parameters Kp, Ti, Td, etc, for the tuning screen).

- Other variables must be continuously exchanged whatever screen is displayed (example : logging, plotting, alarm supervision, etc).
- Other process control variables do not necessarily need to be known by the man-machine interface (example : filtering function time constant, etc).

In order to structure the communication, avoiding any programming, exchange tables associated with the selected control channels are implicitly reserved and structured in the data memory (%MW).

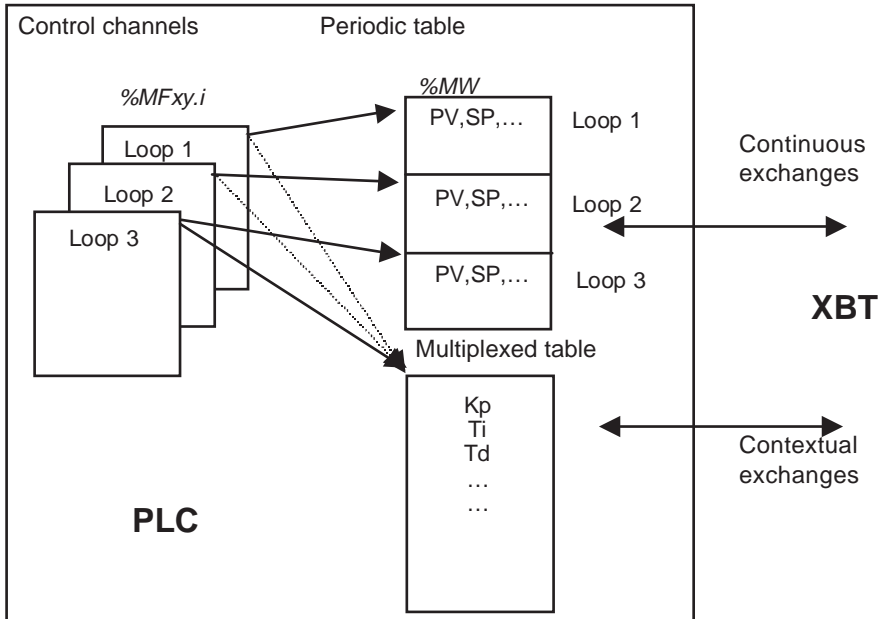
These tables are designed to optimize exchanges between the PLC and the man-machine interface. They are divided up in the following way :

- **Periodic loop table**

This is a contiguous table of 6 %MF for each loop used, containing the process value (PV), the setpoint (SP), the commands (OUTi), the alarms and the status of all the control loops. This zone, which is read continuously, is used to set up the logs, the plotting, the alarm management or the status of all the loops.

- **Multiplexed loop table**

This is a contiguous table of 55 %MF (ie. 110 %MW), containing all the tuning parameters (Kp, Ti, Td, scales, etc) associated with the control loop currently being displayed. There is a single table for all the loops. As it is managed by multiplexing, the number of memory words used can be limited and it is also totally transparent to the user.



---

- **The setpoint programmer multiplexed table**

This is a table of 50 %MF. There is only one and it contains the parameters of all the setpoint programmers, if they exist.

For non-multiplexed use (example : multi-station supervision), these 2 tables can be duplicated and dedicated to each loop (one table per loop or per setpoint programmer).

The following tables should also be added to these 2 (or 3) main zones. These tables are dedicated to operation of the XBTs :

- **The XBT table**

This is a table containing the title and the unit, the status and scales of all the control loops. This table can be read once only or cyclically.

- **The alarms table**

This is a table of 1 %MB per loop, containing all the alarms of all the loops. With XBTs, the alarms are managed using a dialog table. This zone must therefore be consistent with the dialog table address of the XBTs (for further details, please refer to the documentation for these products).

These tables are easy to use, as all these zones are predefined and have a default address. Their contents and detailed descriptions are given in section 6.

---

#### 4.1-3 Method for configuring the man-machine interface

- **Standard use**

For the XBT-F runtime applications provided on the floppy disk, the addresses of the exchange tables have already been filled in. All the user needs to do is select the control loops to be used. When the XBT-F terminal is connected to the process control PLC, the exchanges are set up automatically by the tables which have already been created.

- **Customized use**

If the default memory partition needs to be modified, the runtime applications of the XBT-F graphic man-machine interfaces must also be completely modified.

---

For information purposes, the default table addresses are :

Table	Start address	End address	Maximum size
Alarms table (the dialog table starts at %MW3227	%MW3228	%MW3243	1+16
SPP multiplexed table	%MW3350	%MW3449	100
SPP titles table	%MW3460	%MW3499	40
Periodic loop table	%MW3500	%MW3723	224
Multiplexed loop table	%MW3740	%MW3849	110
Table of loop titles	%MW3850	%MW4073	224

**Note** : For correct operation, initialize (in the PLC) the word authorizing writing of the dialog table (for further details, please refer to the XBT documentation). In a standard application, **word %MW3227 must be set to the value 16#A511**.

---

## 4.2 Process control runtime applications on XBT-F terminals

### 4.2-1 Applications provided

On the floppy disk provided as standard with the product, there are 2 runtime applications programmed using the XBTL-1000 V3 tool, designed for XBT-F graphic terminals :

- The "RFX01MFR.DOP" file, application for 5 inch XBT-F terminals (XBT-FO1 family).
- The "RFX02MFR.DOP" file, application designed for 10 inch XBT and TXBT terminals (XBT-FO2, TXBT-F02 family).

It is easy to enhance these applications by adding personal runtime pages. The programmed process control pages can also be modified or customized (for further details, please refer to the product documentation).

**Note** : The manufacturer guarantees correct operation of the process control pages integrated in the XBTL-1000 applications provided on disk, as long as they are not modified by the user.

In these runtime applications, the following is provided for the process control part :

**For the XBT-F01 :**

- 1 monitoring screen
- 1 multiplexed front panel screen
  
- 1 supervisory control screen for each loop (dynamic trending)
- 1 multiplexed tuning screen
  
- 1 multiplexed autotuning screen
- 1 multiplexed setpoint programmer screen
- Associated alarm pages

The number of loops used is limited to 8.

**For the XBT-F02 and TXBT-F02**

- 1 monitoring screen
- 1 tuning screen with front panel and multiplexed autotuning (bargraphs)
- 1 supervisory control screen for each loop (dynamic trending)
- 1 multiplexed setpoint programmer screen
- Associated alarm pages

The number of loops used is limited to 16.

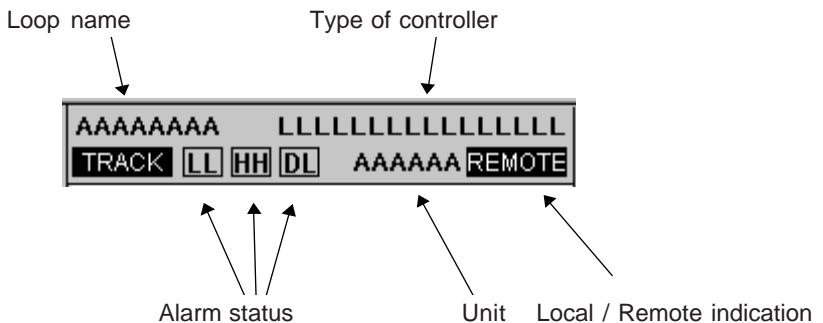
**4.2-2 Runtime page formats**

The design of all the runtime pages is based on the same presentation format :

- There is an alarms zone at the top of the screen. It indicates the active alarm.
- The dynamic function keys each execute a single function (access to the tuning page, starting autotuning, browsing, selecting a loop, etc).

These formats can be changed or imported to enhance personal runtime pages.

**Page formats for the XBT-F01**

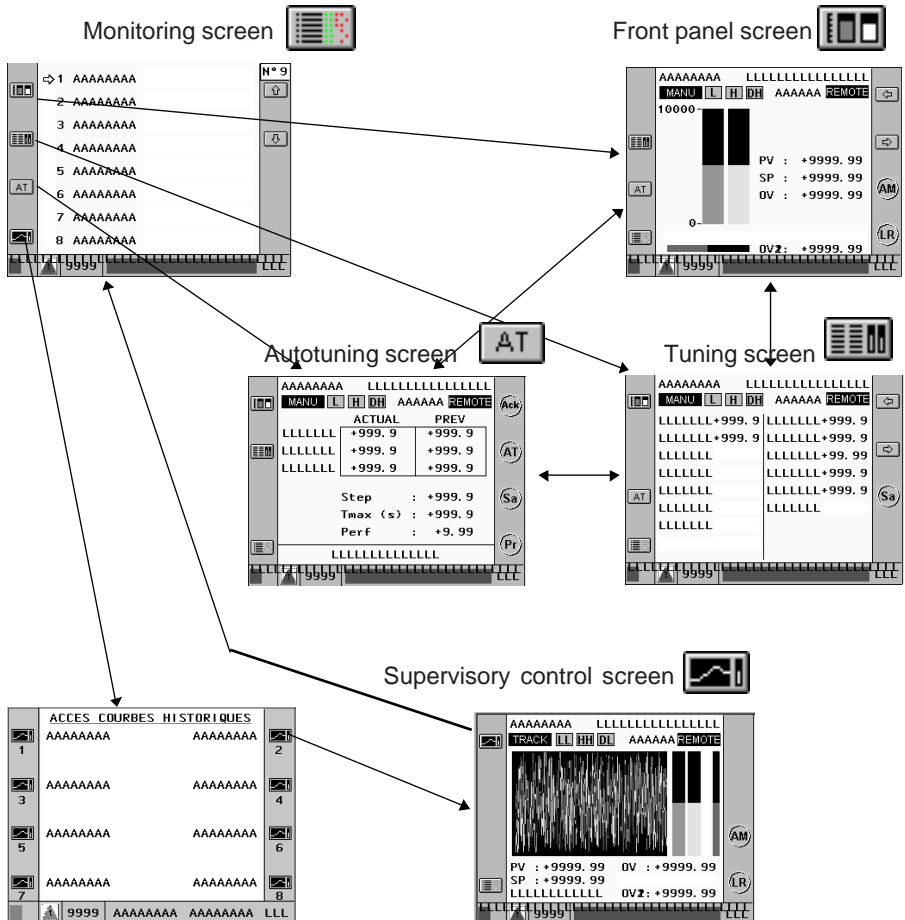




### 4.2-3 Moving around the various views

The dynamic function keys are used for moving between the various views. The suggested browsing can be modified.

#### XBT-F01 application





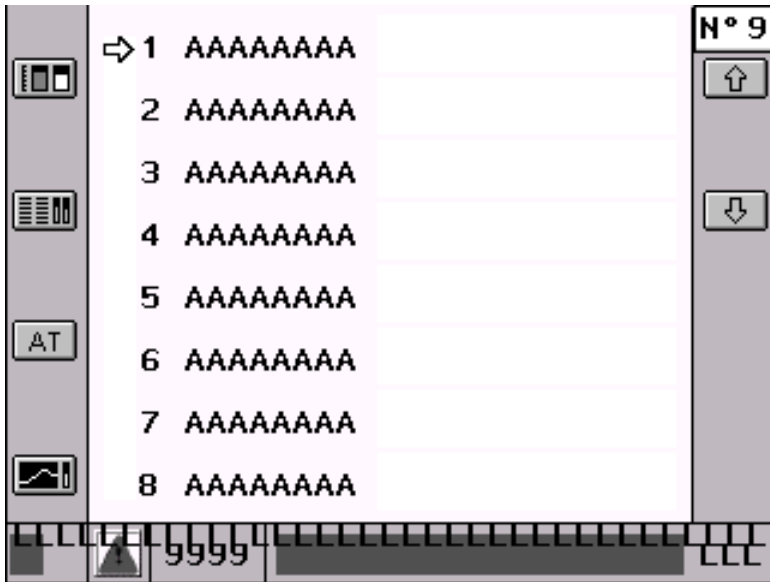


## 4.3 XBT-F01 process control runtime screens

### 4.3-1 Monitoring screen

This is the point of entry to the process control application. It provides an overview of all the loops used.

No data entry is possible in this view.



The following information is displayed for each loop :

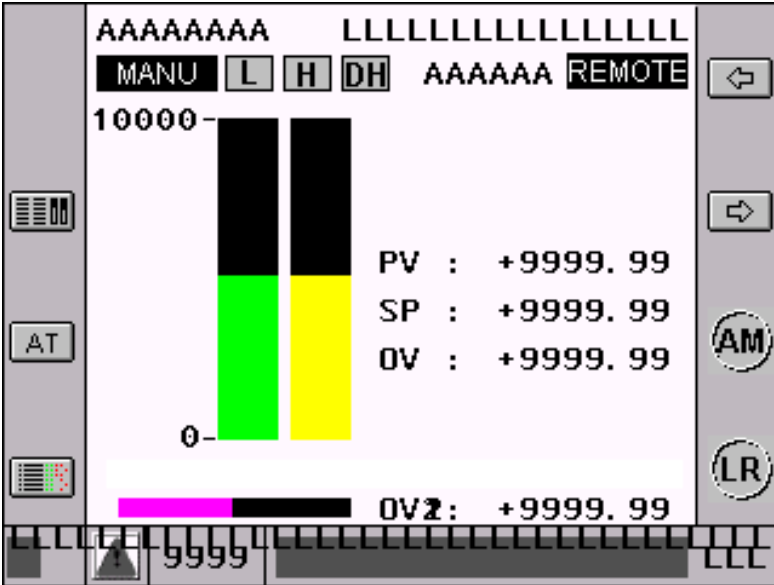
- Loop name,
- Deviation between the process value and the setpoint (bargraph),
- Manu/Auto operating mode,
- Autotuning in progress or not,
- Sum of the alarms.







: These dynamic soft keys are used to select the required loop for the tuning screens.

### 4.3-2 Front panel screen

This is a panel controller view. It gives a global view of a control loop. From this screen it is possible to control the loop in manual mode and the setpoint in Local mode.



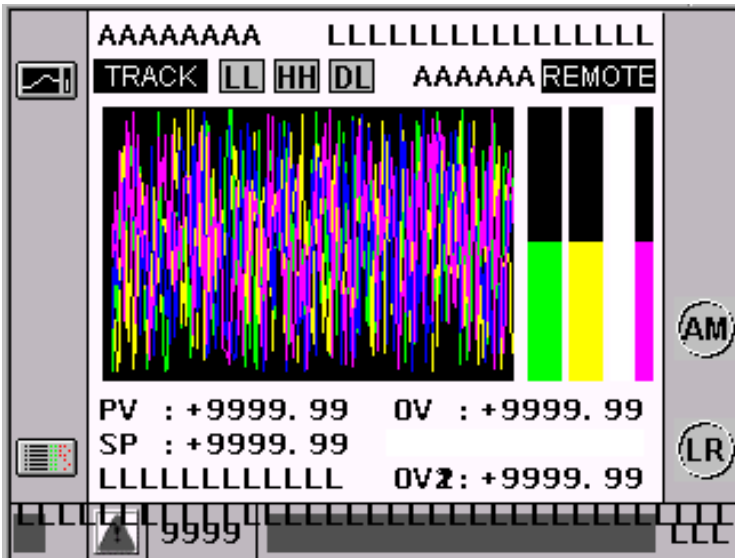
-  : is used to access the previous loop
-  : is used to access the next loop
-  : is used to change the loop to auto or manual. The OV command can be modified
-  : is used to change the setpoint to remote or local. The SP command can be modified

The refresh period is 5 seconds. The total recording time is 26 minutes.

### 4.3-3 Dynamic trending screen

This view contains the same level of information as the loop view, with the additional display of 4 characteristic trends of the loop. The recent history of the trends is recorded. On the XBT-F, there is no log management. As with the previous screen, it is possible to have control in manual mode and the setpoint in local mode.

C



**AM** : is used to change the loop to Auto or Manu

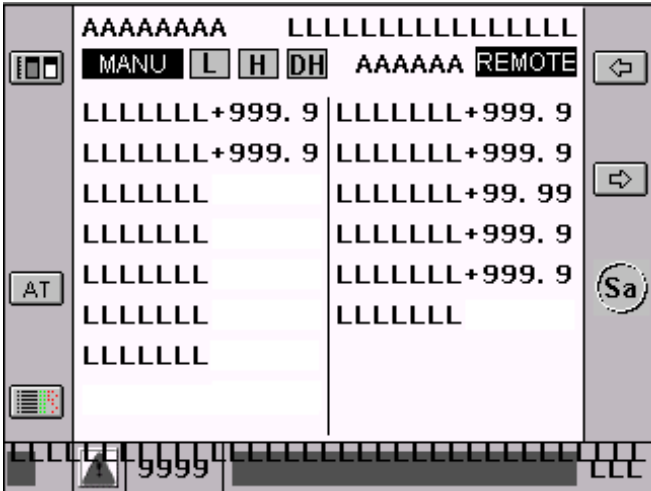
**LR** : is used to change the setpoint to Remote or Local

---

#### 4.3-4 Parameter tuning screen

C

This view is used for tuning the loop controller. This operation should only be performed by a qualified person (tuning engineer). By default, there is no password for these tuning screens. It is always possible to add a password using the XBTL-1000 tool.



: is used to access the previous loop



: is used to access the next loop



: is used to save the current parameter values in the initial values

**Note** : This save operation does not update the PL7 application in any PC which may happen to be connected.

### 4.3-5 Autotuning screen

Specifically for autotuning, this view is used to start a loop autotuning operation. It is also used for returning to the old parameter values before autotuning.



- (AT) : is used to start or stop autotuning
- (Sa) : is used to save the parameters to their initial value
- (Pr) : is used to return to the previous sets of parameters
- (Ack) : is used for acknowledging diagnostics

---

#### 4.3-6 Screen for selecting and tuning the setpoint profile

This screen is used to :

- Display the required configured setpoint profile,
- Modify the segments, ramp and dwell time of the profile,
- Display the current segment and the relative time in the segment,
- Display the process value with monitoring of the deviation in relation to the segment.

To select the required profile, the profile selection view is used to display the names of the various setpoint generators (maximum 10) and select a setpoint generator.

---

#### 4.3-7 Using the alarm pages

The alarm pages and their management are identical to the XBT alarm pages (for further details, see the XBTL1000 document). In the applications which are provided, all process control alarms are in the same group.

There are 6 types of alarm for each control loop :

- Very high process value threshold overshoot,
- High process value threshold overshoot,
- Low process value threshold overshoot,
- Very low process value threshold overshoot,
- High deviation overshoot between process value and setpoint,
- Low deviation overshoot between process value and setpoint.

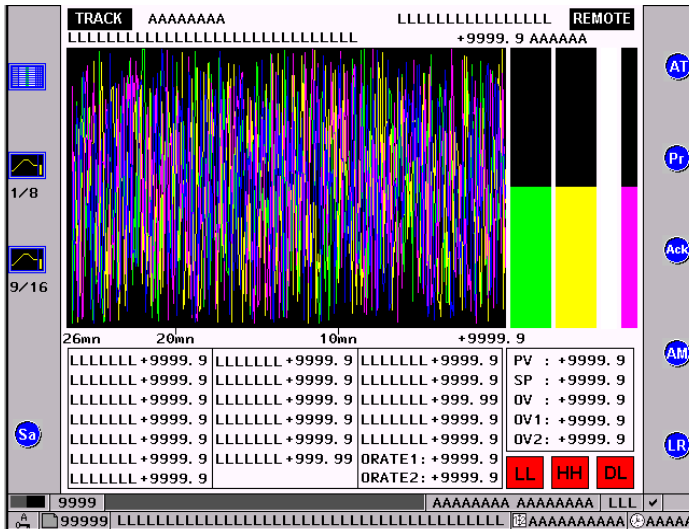











### 4.4-3 Tuning screen

This screen gives an overview of the behavior of a control loop. From this screen it is possible to control the loop in manual mode and the setpoint in Local mode. The recent history of the evolution of the process value, the setpoint and the commands is plotted on the trend chart. This screen is for the user.



-  : is used to change the loop to auto or manual. The OV command can be modified
-  : is used to change the setpoint to remote or local. The SP command can be modified
-  : is used to start or stop autotuning
-  : is used to return to the previous sets of parameters
-  : is used for acknowledging diagnostics

The refresh period is 5 seconds. The total recording time is 26 minutes.

---

#### 4.4-4 Screen for selecting and tuning the setpoint profile

This screen is used to :

- Display the required configured setpoint profile,
- Modify the segments, ramp and dwell time of the profile,
- Display the current segment and the relative time in the segment,
- Display the process value with monitoring of the deviation in relation to the segment.

To select the required profile, the profile selection view can be used to display the names of the various setpoint generators (maximum 10) and select a setpoint generator.

---

#### 4.4-5 Using the alarm pages

The alarm pages and their management are identical to the XBT alarm pages (for further details, see the XBTL1000 document). In the applications which are provided, all process control alarms are in the same group.

There are 6 types of alarm for each control loop :

- Very high process value threshold overshoot,
- High process value threshold overshoot,
- Low process value threshold overshoot,
- Very low process value threshold overshoot,
- High deviation overshoot between process value and setpoint,
- Low deviation overshoot between process value and setpoint.

---

#### 4.4-6 Error messages

There are no messages specific to process control applications. Please refer to the documentation for the XBT terminal.

## 5.1 Executing the control channels

### 5.1-1 Distribution of the process control processing

The processing task period and the control loop sampling periods are different. By default, the MAST task period is 20 ms and the control channel sampling times are 300 ms.

To optimize the processor CPU load, periodic processing of the various control channels is spread over several task scans.

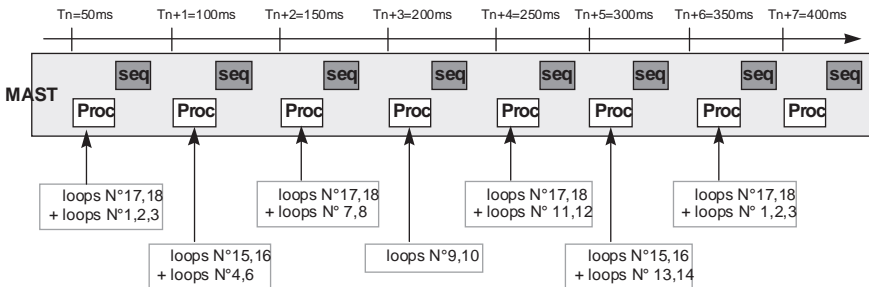
**This distribution of the processing is totally automatic and does not require any programming.**

The order in which the loops are distributed over the task scans is the order in which the loops were created.

Example :

For 18 loops configured with :

- 14 x 300 ms loops : No. 1 to 14
- 2 x 200 ms loops : Nos. 15 and 16
- 2 x 100 ms loops : Nos. 17 and 18



### 5.1-2 Synchronizing the pre- and post-processing

In the rare situations when the user requires close synchronization of the sequential processing and the periodic execution of each control loop, there are 2 bits integrated in the status words for each control loop :

- STS\_TOP\_NEXT\_CYCLE : bit to trigger pre-processing
- STS\_TOP\_CUR\_CYCLE : bit to trigger post-processing

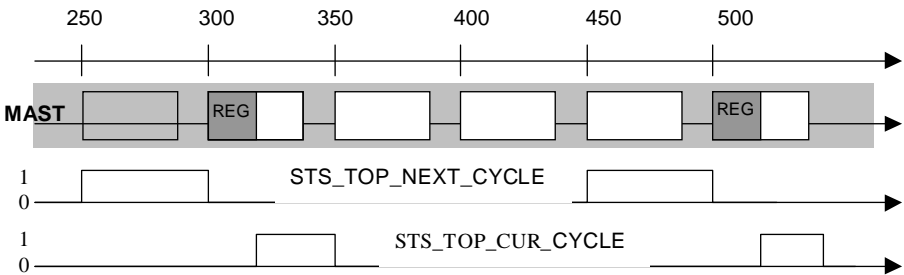
These two bits can be used as an enable condition for a processing operation written in Structured Text or Ladder language.

(eg : start/stop operating modes, calculation of variance, calculation of compensation). The pre-processing status bit changes to 1 during the task scan preceding the process control processing execution scan. The post-processing status bit changes to 1 after processing of the process control for the whole of the task scan.

To synchronize the sequential processing correctly with the process control calculations, processing operations must be integrated in the same task.

In manual mode or tracking mode, the command is generated on every processing scan. The synchronization bits are always set to 1 in these modes.

Example: 200 ms loop on a 50 ms MAST



### 5.1-3 Multitask applications

For optimum, deterministic operation it is advisable, for a given control channel, to assign the following to the same task :

- The associated I/O channels,
- The sequential pre- and post-processing,
- The sequential processing which manages the PLC operating modes.

## 5.2 PLC operating modes

Depending on the modifications made to the behavior of the PLC by the user or by default, the control channels follow a predefined downgraded operating mode. These operating modes are managed as described in the following sections.

---

### 5.2-1 On PLC power-up

When the PLC is powered up, the system looks for a valid application in the user memory space.

If the application is valid : the system changes to a configuration state and each control channel is called. The context of the channel is then set to the initial values which can be used for operation.

If the application is not valid : the system changes to a state of waiting for a reconfiguration request.

---

### 5.2-2 CPU in RUN mode

In RUN mode, the processor executes the following in succession, on each scan :

- Reading of the input channels,
- Execution of the setpoint programs,
- Execution of the control loops,
- Processing of the sequential program,
- Writing of the outputs.

All the control channels are called on each task scan :

- Calculation of the process value (PV) and the feedforward (OUT-FF), the alarm management, the operating modes, the setpoint programmers and the generation of the command in Manu or tracking, are all performed each scan
- Generation of the loop command in Auto mode and the setpoint are performed in the sampling period.

---

### 5.2-3 On a CPU change from RUN to STOP

Stopping of the CPU or the task is not seen directly by the application-specific channels. It corresponds to a stop of all the functions being executed.

The control channels are no longer executed. They refuse all commands (Auto/Manu etc). The results of the calculations remain in the original state. The physical outputs change to the fallback mode defined at configuration.

However, the inputs are refreshed, and the parameters can be modified. The validity will be checked at the next startup.

---

#### 5.2-4 On a cold restart

There may be a number of reasons for the cold restart :

- Change of cartridge (cold start) or reconfiguration (loading a program, transferring a new application, etc).
- An initial configuration.

After configuration or reconfiguration, system bit % S0 is raised.

The control channels check their configuration and initialize their parameters and their state at the first scan. The processing of the algorithm is executed from the second scan onwards.

All the commands generated in the sequential processing during the first scan are taken into account, except for a command for autotuning or tracking on the loop controller. The command is refused.

---

#### 5.2-5 On a warm restart

A warm restart occurs when mains power is returned to the CPU.

At the moment of the power outage, the parameters are saved.

The system and application contexts (application data, operating modes) are retained. Any autotuning in progress is aborted. The control channels are executed from the first scan onwards.

---

### 5.3 Control loop operating modes

#### 5.3-1 Manual control

The operator can use manual mode to apply a value directly to the loop controller output. It can be selected from the PL7 debug screens.

It can also be controlled from the various XBT-F runtime screens.

The change to manual is performed by sending a command. When this command is taken into account, status bit STS\_AUTO\_MANU indicates the status.

The loop controller or the loop can then be controlled.

When the command is a numerical value, it is subject to upper and lower limits and speed limitation.

The output is processed on each task scan.

---

Special case of a SERVO output without position feedback : manual control is always via variable OUT\_MAN. This is limited to between 0 and 100. Since OUT\_MAN has no direct link with the actual position of the actuator, it must be possible to open and close the actuator, even if OUT\_MAN has reached one of its limits. To do this, a value outside its limits can be entered in OUT\_MAN : OUT\_MAN will be peak limited, but the calculated command variation will be taken into account by the SERVO function.

Example : OUT\_MAN = 100.0, the actuator is 50% open. To apply an opening of 70%, the following must be written : OUT\_MAN = 120.0. OUT\_MAN will then return to the peak limited value 100.0.

---

### 5.3-2 Automatic execution

In automatic mode, the command value is calculated by the loop controller from the setpoint value and the PV value.

The user can change to automatic mode from the PL7 or XBT-F screens.

It is also possible to change to automatic mode by sending a command.

When this command is taken into account the Auto/Manu status bit indicates the status.

The output is processed at each sampling period.

---

### 5.3-3 Starting an autotuning operation

To start an autotuning operation, the user must first fill in the step function duration, the performance and the amplitude of the required command.

If the values of these parameters are too small or too large, autotuning will not start.

The loop controller can be in manual or automatic mode before execution of an autotuning operation.

The function is performed by sending a command.

While the autotuning process is being performed (2.5 times the step function duration), the autotuning function controls the loop controller output. The operator cannot modify it.

The function then automatically provides the loop controller coefficients.

The diagnostic word indicates any problems which may be detected.

When autotuning is completed, the loop controller returns to the operating mode it was in before autotuning.

If the loop controller is in automatic mode, it restarts with the new set of parameters.

The operator can return to the previous set of parameters using the "Previous setting" command.

---

### 5.3-4 Execution in tracking mode

This operating mode is used for forcing the numeric outputs of a control loop. It is often used when closing an open loop to ensure a smooth transition on the actuators. It is also used in the special case of redundant architectures consisting of an active PLC and a passive PLC. In fact, the output values of the passive PLC must be identical to those of the active PLC. Tracking mode makes this possible.

Tracking mode uses one parameter (%MF address) and one command (sending a command) :

- The change to tracking mode is performed by sending a command. It is refused if the address containing the tracking value has not been filled in.
- If the command is sent, the control loop output is overwritten by the tracking value, whereas the internal variables are regularly initialized with the output value.
- If the «non tracking» command is given, the function returns to its previous operating mode, with a smooth transition on the output.

Tracking mode has the highest priority in relation to automatic, manual or autotuning modes.

Tracking mode does not exist in all the loop controllers since, in some case, this mode performs no useful function. For example, the ON/OFF controller does not have this mode.

---

### 5.3-5 Auto/manu changeover

The manual command is continuously updated : it is said to follow the command output. At a change from Auto to Manu the first manual value given is the last value calculated by the loop controller. This ensures a smooth transition.

---

### 5.3-6 Manu/Auto changeover (apart from ONOFF loop controller)

The change from Manu to Auto is made smoothly on the command output. For PID there are two possible scenarios :

- **With integral action  $T_i \neq 0$**

The incremental PID algorithm ensures a smooth transition from Manu to Auto.



---

In this case the PID algorithm always tracks the output actually applied. This principle is described in detail for each loop in the following sections.

- **Without integral action  $T_i = 0$**

It is possible to obtain a smooth Manu/Auto changeover by configuring "bumpless" mode, which is a parameter of the PID function (if there is an integral action, this configuration has no effect).

The manual integral parameter OUTBIAS is then recalculated during the changeover to take account of the deviation between the actual output and the output calculated by the PID algorithm in absolute format.

If "bumpless" mode is not selected, OUTBIAS is not recalculated at the changeover.

---

### 5.3-7 Behavior of the loops on an I/O fault

By design, control loops do not take account of any faults which may occur on I/O cards. It is possible to change the operating mode of a loop from the sequential program when there is an I/O fault. Monitoring of the diagnostic words and bits of the associated modules by the application program can be used to generate the appropriate command for this loop.

## 5.4 Process loop operating mode

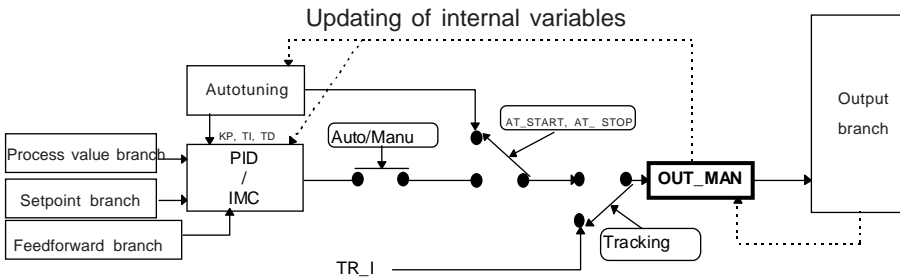
There are 2, 3 or 4 different operating modes (automatic, manual, autotuning, tracking) depending on the type of loop controller.

This means that :

- An autotuning operation can be started if the loop controller is in automatic or in manual mode.
- A change to tracking mode takes priority and will abort any autotuning operation which may be in progress.

It is possible to configure the initial operating mode of the loop on a cold start. The following can be specified :

- Whether the setpoint is to be remote or local, and the initial local setpoint value.
- Whether the loop controller is to start in manual or automatic mode, and the initial manual value if the loop controller is not an ONOFF controller.



The above diagram shows a process loop with a PID controller.

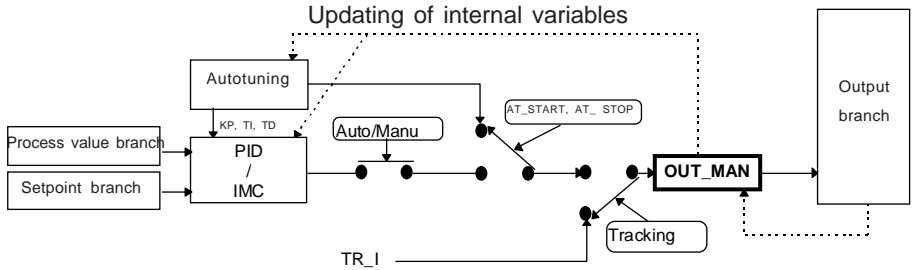
The dotted line from the output branch to OUT\_MAN shows the limits being taken into account.

If a split range or hot-cool controller is used, there are two output branches. In this case, autotuning and SERVO output without position feedback are not available.

If an ONOFF 2 or 3 state loop controller is used, there is no feedforward branch or output branch.

## 5.5 Single loop operating mode (3 single loops)

The 3 single loops are independent and are represented by a diagram equivalent to that of the process loop, except that the feedforward branch does not exist. Moreover, a hot-cool or split range controller cannot be configured. The process value and setpoint branches are simplified (see description of the branches).



One autotuning operation at a time can be started on the 3 loops of the control channel. If another autotuning operation is requested, it is refused.

---

## 5.6 Cascaded loop operating mode

---

All transitions are made smoothly on the loop controller outputs.

Management of the operating modes of the slave loop is identical to that for a process loop : in fact this loop behaves as if it were a single loop.

However there are specific mechanisms for the master loop :

- Manual mode and auto/manu changeover are identical to those for a process loop.
- If the loop controller is in automatic mode (default mode), there are two distinct options :
  - The slave controller is in automatic mode and uses the remote setpoint : the cascade is closed, and the master loop is really in automatic mode.
  - Otherwise (local setpoint, or autotuning of the slave loop controller in progress, or tracking mode), the cascade is open. In this case the master loop controller is in tracking mode.

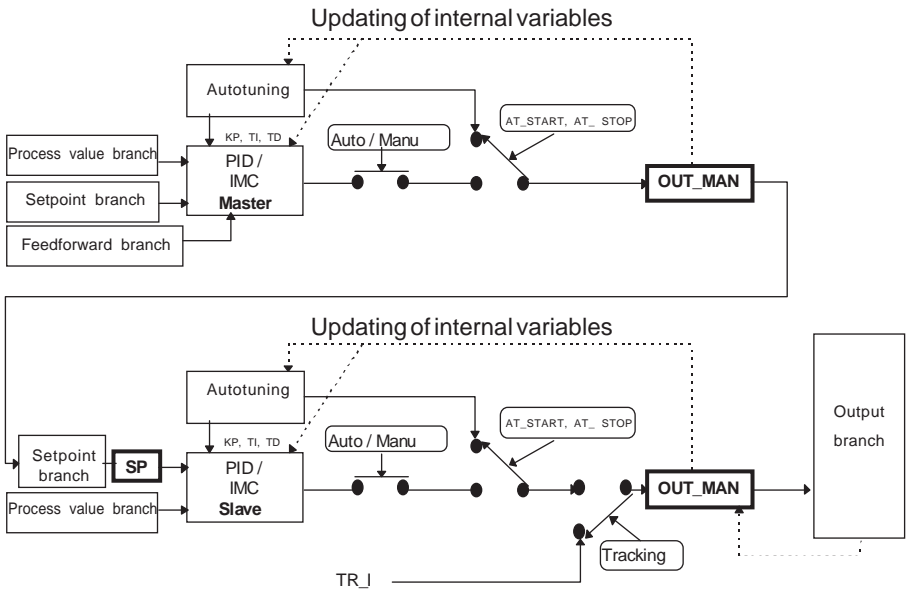
The aim is to prevent bumps when closing the cascade, and there are therefore several situations :

- If the slave loop controller uses the local setpoint, the master loop controller tracks the local setpoint of the slave.
- Otherwise, if the slave loop controller has an integral action, the master loop controller tracks the process value of the slave loop.
- If not, the slave loop controller is a P or a PD, and the output of the master loop is calculated optimally to ensure smooth closing of the cascade (according to the slave loop controller output and its parameters).

On a cold start, the master loop controller always starts in automatic mode. However, the initial operating mode of the slave loop controller can be configured, as well as the type of setpoint (remote/local) for each loop.

The master loop has an additional function, called “ clamping the output ”, which consists, when the master is in automatic mode and the slave output is saturated, of freezing the evolution of the output of the master in the direction which saturates the slave. This has the advantage of limiting saturation of the integral action of the master. This function is thus only active when the master loop controller has an integral action.

Example : the slave loop controller is in automatic mode, it is configured in reverse action mode, and its output is at its upper limit. To release the output from its limit using the setpoint, the setpoint must be lowered. The output of the master is therefore clamped in the direction of an increase.



The cascaded loop consists globally of 2 process loops with a number of restrictions and some additional functions.

The OUT\_MAN output of the master loop is the remote setpoint of the setpoint branch of the slave loop. The OUT\_MAN output is thus expressed in the scale of the slave loop. It is subject to the level limit of the setpoint branch of the slave loop.

The loop controller of the slave loops can also be a hot-cool or split range controller. The auto/manu operating mode and the manual command value of the master loop cannot be controlled from the process control application-specific screen. They can however be accessed from the user program.

The restrictions are :

- No totalizer on the master loop process value branch,
- No feedforward branch on the slave loop,
- No ONOFF controller on any of the loops,
- The setpoint branch of the slave loop is a simple branch, without scaling,
- Only one model-based controller can be configured on the master loop or the slave loop.

## 5.7 Autoselective loop operating modes

This loop consists of making two controllers operate on the same output. Each loop controller produces an action, and a comparator (min or max) selects the action to be applied. There is a main loop consisting of a process loop, and a secondary loop consisting of a single loop, and the two loops share a single output branch.

The autoselective loop is used, for example, to implement restricted process control, with the main loop controlling the main value and the secondary loop being used to prevent an auxiliary value from exceeding a limit (or restriction) specified by the setpoint of that loop.

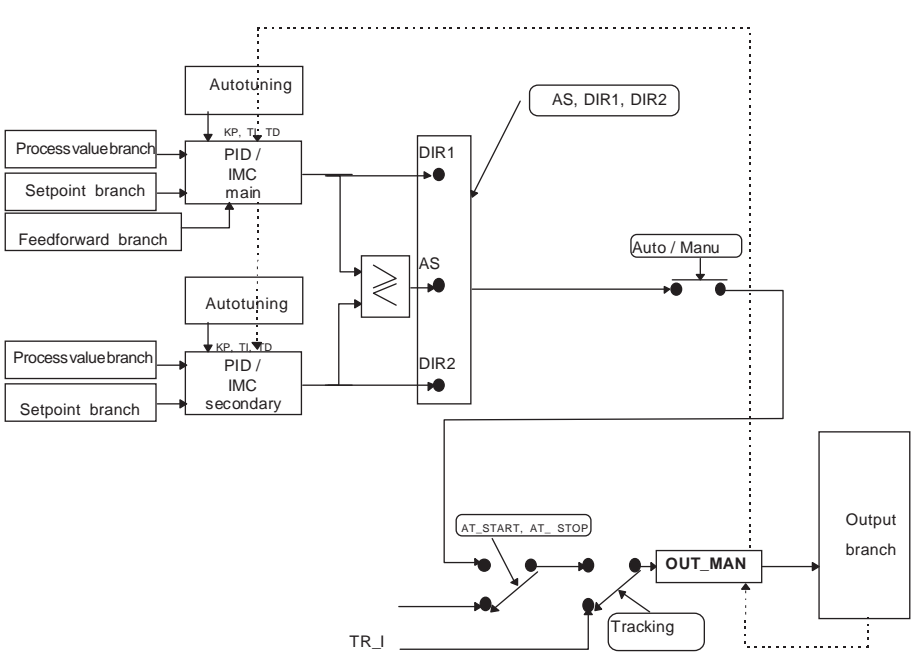
It is also possible to inhibit one of the loops in order to have only a process loop or a single loop, using one of the following commands : direct output 1 or direct output 2. Both loop controllers have the same sampling period.

The autoselective loop can be configured in two different ways.

### Case 1

A single auto/manu on the output branch after the selector : the manual command value OUT\_MAN is therefore applied directly to the loop output.

Updating of internal variables



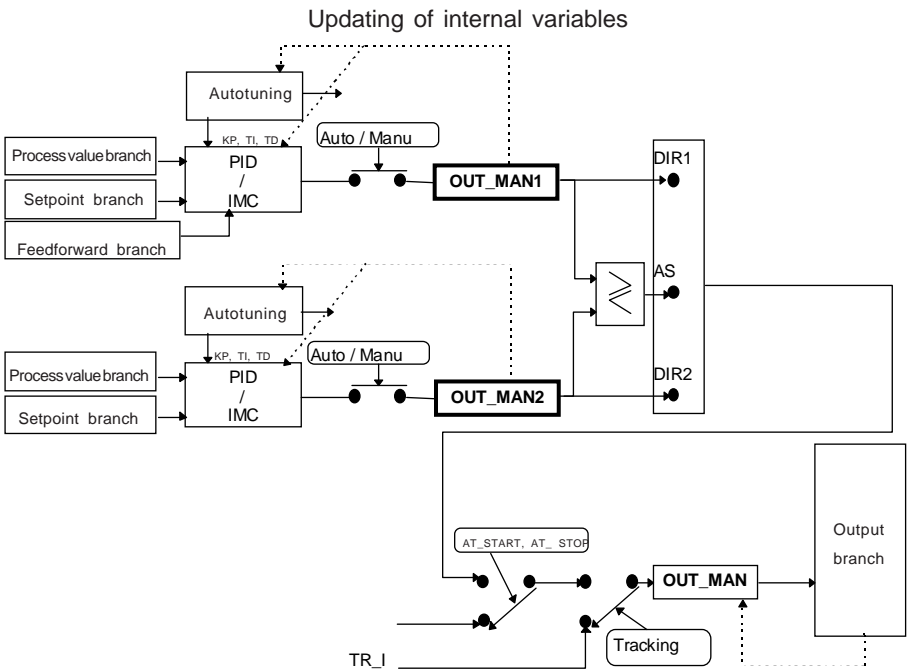


In this case, both loop controllers are always in automatic mode, and track the applied command, OUT\_MAN. When the loop is in automatic mode the output of these loop controllers is taken into account. If the loop is not in automatic mode the output is not taken into account. As it tracks the actual output, the changeover will be smooth, as long as the integral action of the loop controllers is used.

The initial operating mode of the loop can be configured, as can the type of initial setpoint (remote/local) of each loop controller.

**Case 2**

An auto/manu on the output of each loop controller : the operating mode of each loop controller is thus independent, and the output value of each loop controller can be set manually, downstream of the selector.



In this case, in manual mode, the operator does not alter the actuator command, he performs operations on each loop controller (OUT\_MAN1 and OUT\_MAN2). As long as one of the loop controllers is in automatic mode, the output is selected during the sampling period of each loop. If they are both in manual mode, it is performed on each task scan.

---

Both loop controllers continuously track the actual output OUT\_MAN. In automatic mode, if they contain an integral action, they take account of the preceding value of the OUT\_MAN output. This means that at a manu/auto changeover, the loop controller does not restart from its last manual value, but from the last value of the actual output OUT\_MAN.

The initial operating mode of each loop controller can be configured, as well as the type of initial setpoint (remote/local) for each loop controller.

Starting an autotuning operation forces the autoselector to the direct position of the autotuned loop. At the end of autotuning, the autoselector must be returned to the required position, if it is different from the position which has been imposed.



## 6.1 Object language addressing

During application programming, the language objects associated with the I/O and with the parameters of the configured control channels are available to the user. These language objects can be used in the various PL7 tools, specifically language editors and dynamic animation tables.

The syntax of these language objects is as follows :

%	M, K	W,D,F	x	• i	• r
IEC sign	<b>Object type</b> M : internal words K : internal const.	<b>Format</b> W=word D=dble word F=floating point	<b>Position</b> x=0or1 (CPU)	<b>Chann. no.</b> i=4 to 13	<b>Rank</b> r=0to255

Example : %MF1.5.30 : process loop value of control channel no. 5 of the processor located at slot 1.

## 6.2 Language objects associated with process control channels

All the variables associated with a process control channel (example : Kp, T\_FILT, etc) are in read and/or implicit write mode.

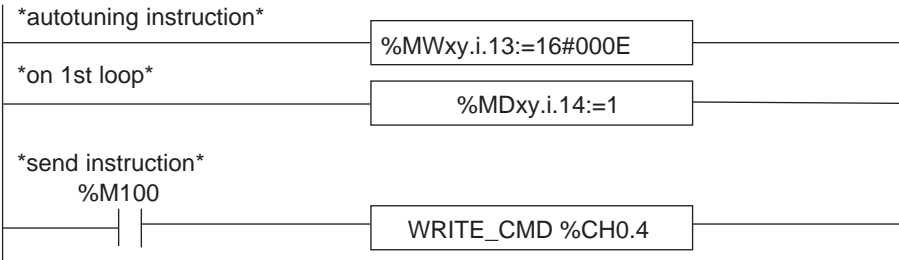
Language object %CH is used to simplify explicit reading and writing. It is used to :

- Read module and channel status words,
- Write parameters,
- Save parameters,
- Send commands.

### 6.2-1 Sending commands

Explicit instructions apply to channel language object %MWxy.i.

Example : Sending an autotuning instruction to the first loop of "3 single loops"  
This word contains the explicit command which acts on the control loop.



Any number of commands can be sent on a PLC scan. The instruction is taken into account and the appropriate control channel status updated in the scan following the task.

**Note :** Instructions associated with the controller operating mode (Auto, Manu, tracking, autotuning) cannot be sent simultaneously in the same cycle (only the last instruction executed in the cycle is taken into account).

However, complementary instructions (such as Remote, freeze totalizing, etc), can be sent in the same cycle.

## 6.2-2 Command parameter values (%MDxy.i.j)

### Setpoint programmer

%MDxy.i.8 = j      profile j    j={1, ..., 3}

### Cascaded loop

%MDxy.i.12 = 1    master loop  
%MDxy.i.12 = 2    slave loop

### Autoselective loop

%MDxy.i.12 = 1    main loop  
%MDxy.i.12 = 2    secondary loop

### 3 single loops

%MDxy.i.14 = j      loop j    j={1, 2, 3}

**6.2-3 Control loop word command values (%MWxy.i.11)**

<b>Value</b>	<b>Meaning</b>
16#0001	Switch to Simulation or non-simulation of process value input (flip/flop)
16#0002	Switch to Remote or Local mode (flip/flop)
16#0003	Switch to Manual or Automatic mode (flip/flop)
16#0004	Freeze totalizing
16#0005	Unfreeze totalizing
16#0006	Reinitialize totalizing
16#0007	Select Remote1 setpoint
16#0008	Select Remote2 setpoint
16#0009	Not used
16#000A	Not used
16#000B	Switch to Simulation or non-simulation of Feed Forward input (flip/flop)
16#000C	Switch to tracking mode
16#000D	Switch to non-tracking mode
16#000E	Start autotuning
16#000F	Stop autotuning
16#0010	Return to previous settings
16#0011	Feedback used
16#0012	Feedback not used
16#0013	Acknowledgment of autotuning diagnostics
16#0014	Activate Raise
16#0015	Deactivate Raise
16#0016	Activate Lower
16#0017	Deactivate Lower
16#0018	Not used
16#0019	Reinitialize Servo1
16#0020	Reinitialize Servo2
16#0021	Select setpoint in Local mode
16#0022	Select setpoint in Remote mode
16#0023	Switch to Manu
16#0024	Switch to Auto

16#0025	Position of selector switch set for autoselection
16#0026	Position of selector switch on main loop
16#0027	Position of selector switch on secondary loop

#### 6.2-4 Setpoint programmer command values (%MWxy.i.7)

Value	Meaning
16#0001	Reinitialize setpoint programmer
16#0002	Trigger execution of selected profile
16#0003	Stop execution of selected profile
16#0004	Freeze profile changes
16#0005	Unfreeze current profile
16#0006	Jump to next segment
16#0007	Jump to previous segment
16#0008	Disable the guaranteed dwell time function
16#0009	Activate the guaranteed dwell time function
16#000A	Freeze/unfreeze profile changes (flip/flop)
16#000B	Freeze/unfreeze the guaranteed dwell time function (flip/flop)

## 6.3 Language objects associated with the process loop channel

### 6.3-1 Configuration language objects

Address	Parameter name	Default value	Comment
%KW@.chann.0	CONFIG_0	Not applicable	Word containing the various Process value configuration bits
%KW@.chann.0:X0	Filtrage	Absent(0)	Process value branch filtering function
%KW@.chann.0:X1	Générateur de fonction	Absent(0)	Process value branch function generator
%KW@.chann.0:X2	Totalisateur	Absent(0)	Process value branch Totalizer function
%KW@.chann.0:X3	Racine Carrée	Absent(0)	Process value branch root function
%KW@.chann.0:X4	Alarmes	Present	Process value branch alarm function
%KW@.chann.0:X8	PV_CLIP	Absent(0)	Peak limiting (or not) of process value
%KW@.chann.0:X9	EXTRAPOL	No (0)	Extrapolation of function generator
%KW@.chann.0:X10	PV_UNI_BIP	Unipolar (0)	Process value type unipolar/bipolar
%KW@.chann.0:X11	PV_EXTERN	Absent (0)	Select Standard (0) / External (1) process value
%KW@.chann.0:X13	Totalisateur: Unité mesure	1	(X13=0, X14 =0): phys/ms (X13=1, X14 =0): phys/s
%KW@.chann.0:X14	Totalisateur: Unité mesure	0	(X13=0, X14 =1): phys/mn (X13=1, X14 =1): phys/h
%KW@.chann.1	CONFIG_1	Not applicable	Word containing the various Setpoint configuration bits
%KW@.chann.1:X0	SP_Simple	Selected(1)	Type of setpoint selected : Simple
%KW@.chann.1:X1	SP_Sélection	Not selected(0)	Type of setpoint selected : Selection
%KW@.chann.1:X2	Speed_Limiteur	Not selected(0)	Setpoint speed limiter
%KW@.chann.1:X3	SP_SPP	Not selected(0)	Type of setpoint selected : Programmer
%KW@.chann.1:X4	RL/L	Remote local(0)	Speed limiter on local setpoint, or in remote/local mode

Address	Parameter name	Default value	Comment
%KW@.chann.1:X8	Sel_min	Absent(0)	Selected function for Selection setpoint type
%KW@.chann.1:X9	Sel_max	Absent(0)	Selected function for Selection setpoint type
%KW@.chann.1:X10	Sel_switch	Present on	Selected function for Selection setpoint type
%KW@.chann.1:X11	R/L_INIT	Local (1)	Initial value of Remote/ Local selected setpoint
%KW@.chann.1:X12	R1/R2_INIT	R1 (0)	Initial value of state of selected setpoint
%KW@.chann.1:X13	SP_Ratio	Not selected(0)	Type of setpoint selected : Ratio
%KW@.chann.1:X14	SP_Limiteur	Not present	Setpoint limiter (eg. Param_SP)
%KW@.chann.1:X15	SP_Folw	Non-tracking setpoint (0)	Tracking setpoint
%KW@.chann.2	CONFIG_2	Not applicable	Word containing the various loop controller and FF configuration bits
%KW@.chann.2:X0	PID	Present	PID function of loop controller branch
%KW@.chann.2:X1	ONOFF 2	Absent(0)	Controller 2-state ONOFF branch
%KW@.chann.2:X2	ONOFF 3	Absent(0)	Controller 3-state ONOFF branch
%KW@.chann.2:X3	SPLRG/ChFroid	Not applicable	OR presence bits Hot/Cool and Split/Range
%KW@.chann.2:X4	Split/Range	Absent(0)	Loop controller branch Split/Range function
%KW@.chann.2:X5	Chaud/Froid	Not selected	Loop controller branch Hot/Cool function
%KW@.chann.2:X6	Alarmes_DEV	Present	Alarm function on loop controller branch deviation
%KW@.chann.2:X7	Feed Forward	Absent(0)	Presence of a Feed Forward input
%KW@.chann.2:X8	BUMP	With bumps (1)	Management of bumps on change of operating mode
%KW@.chann.2:X9	PV_DEV	On PV (0)	Type of derivative action

Address	Parameter name	Default value	Comment
%KW@.chann.2:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.2:X11	REV_DIR	Inverse PID action(0)	Type of loop controller action
%KW@.chann.2:X12	MANU/AUTO_INIT	Manu (0)	Initial value of loop controller operating mode
%KW@.chann.2:X13	Lead Lag	Absent(0)	Feed Forward branch Lead Lag function
%KW@.chann.2:X14	FF_UNI_BIP	Unipolar	Feed Forward value type unipolar/bipolar
%KW@.chann.2:X15	IMC	Absent(0)	Loop controller branch IMC function
%KW@.chann.3	CONFIG_3	Not applicable	Word containing the various output configuration bits
%KW@.chann.3:X0	Servo	Selected	Type of output selected : Servo
%KW@.chann.3:X1	Servo2	Selected	Type of output selected : Servo
%KW@.chann.3:X2	Analogique1	Selected	Type of output selected : Analog
%KW@.chann.3:X3	Analogique2	Selected	Type of output selected : Analog
%KW@.chann.3:X4	PWM1	Selected	Type of output selected : PWM
%KW@.chann.3:X5	PWM2	Selected	Type of output selected : PWM
%KW@.chann.3:X8	POT_REV1	Direct (0)	Servo feedback direction
%KW@.chann.3:X9	POT_REV2	Direct (0)	Servo feedback direction
%KW@.chann.3:X10	POT_VAL1_INIT	No (0)	Existence of Servo feedback
%KW@.chann.3:X11	POT_VAL2_INIT	Yes (1)	Existence of Servo feedback (Reserved)
%KW@.chann.3:X12	ANALOG1_UNI_BIP	Unipolar	Type of analog output unipolar/bipolar
%KW@.chann.3:X13	ANALOG2_UNI_BIP	Unipolar (0)	Type of analog output unipolar/bipolar
%KW@.chann.4	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.8	Unité de la boucle		

Address	Parameter name	Default value	Comment
%KW@.chann.2:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.2:X11	REV_DIR	Inverse PID action (0)	Type of loop controller action
%KW@.chann.2:X12	MANU/AUTO_INIT	Manu (0)	Initial value of loop controller operating mode
%KW@.chann.2:X13	Lead Lag	Absent(0)	Feed Forward branch Lead Lag function
%KW@.chann.2:X14	FF_UNI_BIP	Unipolar	Feed Forward value type unipolar/bipolar
%KW@.chann.2:X15	IMC	Absent(0)	Loop controller branch IMC function
%KW@.chann.3	CONFIG_3	Not applicable	Word containing the various output configuration bits
%KW@.chann.3:X0	Servo	Selected	Type of output selected : Servo
%KW@.chann.3:X1	Servo2	Selected	Type of output selected : Servo
%KW@.chann.3:X2	Analogique1	Selected	Type of output selected : Analog
%KW@.chann.3:X3	Analogique2	Selected	Type of output selected : Analog
%KW@.chann.3:X4	PWM1	Selected	Type of output selected : PWM
%KW@.chann.3:X5	PWM2	Selected	Type of output selected : PWM
%KW@.chann.3:X8	POT_REV1	Direct (0)	Servo feedback direction
%KW@.chann.3:X9	POT_REV2	Direct (0)	Servo feedback direction
%KW@.chann.3:X10	POT_VAL1_INIT	No (0)	Existence of Servo feedback
%KW@.chann.3:X11	POT_VAL2_INIT	Yes (1)	Existence of Servo feedback (Reserved)
%KW@.chann.3:X12	ANALOG1_UNI_BIP	Unipolar	Analog output type unipolar/bipolar
%KW@.chann.3:X13	ANALOG2_UNI_BIP	Unipolar (0)	Analog output type unipolar/bipolar
%KW@.chann.4	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.8	Unité de la boucle		Loop unit



### 6.3-2 Default and diagnostic language objects

Address	Parameter name	Default value	Comment
%MW@.chann.0	Echange en cours		
%MW@.chann.1	Compte Rendu		
%MW@.chann.2	STATUS1_VOIE		
%MW@.chann.2:X0	STS_DEF_EXTERNE		
%MW@.chann.2:X1	STS_DEF_DEPASS_GAMME		
%MW@.chann.2:X2	STS_DEF_BORNIER		
%MW@.chann.2:X3	STS_DEF_EXT_PROT		
%MW@.chann.2:X4	STS_DEFAULT_INTERNE		Serious internal fault
%MW@.chann.2:X5	STS_DEFAULT_CONFIG		
%MW@.chann.2:X6	STS_DEFAULT_COMMUNIC		
%MW@.chann.2:X7	WARN		Warning sigma
%MW@.chann.2:X8	STS_ERR_CALC_CORR		Loop controller branch calculation error
%MW@.chann.2:X9	STS_ERR_FLOT_CORR		Loop controller branch floating point error
%MW@.chann.2:X10	STS_ERR_CALC_PV		PV branch calculation error
%MW@.chann.2:X11	STS_ERR_FLOT_PV		PV branch floating point error
%MW@.chann.2:X12	STS_ERR_CALC_OUT		OUT branch calculation error
%MW@.chann.2:X13	STS_ERR_FLOT_OUT		OUT branch floating point error
%MW@.chann.3	STATUS2_VOIE		
%MW@.chann.3:X0	STS_ERR_SCALE_PV		PV branch incorrect scale
%MW@.chann.3:X1	STS_ERR_TH_SPLRG		SPLRG function thresholds incorrect
%MW@.chann.3:X2	STS_ERR_SCALE_OUT1		OUT1 branch scale incorrect
%MW@.chann.3:X3	STS_ERR_SCALE_OUT2		OUT2 branch scale incorrect
%MW@.chann.3:X4	STS_MISSING_POT_ADR		Servo feedback address missing
%MW@.chann.4	STATUS1		Word containing the various Process Value/ Setpoint status bits

Address	Parameter name	Default value	Comment
%MW@.chann.4:X0	STS_HOLD_TOT		Freezes totalizing function
%MW@.chann.4:X1	STS_PV_Sim		Simulated value
%MW@.chann.4:X2	STS_PV_H_LIM		Upper limit on process value
%MW@.chann.4:X3	STS_PV_L_LIM		Lower limit on process value
%MW@.chann.4:X4	STS_SP_H_LIM		Upper limit on setpoint
%MW@.chann.4:X5	STS_SP_L_LIM		Lower limit on setpoint
%MW@.chann.4:X6	STS_L_R	R/L Init	Remote setpoint (1) Local setpoint (0)
%MW@.chann.4:X7	STS_R1_R2		Remote2 setpoint (1) Remote1 setpoint (0)
%MW@.chann.4:X8	STS_ALARMS		Logic OR of value alarms
%MW@.chann.4:X9	STS_HH		Very high alarm
%MW@.chann.4:X10	STS_H		High alarm
%MW@.chann.4:X11	STS_L		Low alarm
%MW@.chann.4:X12	STS_LL		Very low alarm
%MW@.chann.4:X13	STS_DEVH		High alarm for setpoint process value deviation (>0)
%MW@.chann.4:X14	STS_DEVL		Low alarm for setpoint process value deviation (<0)
%MW@.chann.4:X15	STS_THLD_DONE		Totalizer threshold reached
%MW@.chann.5	STATUS2	Not applicable	Word containing the various loop controller/ Feed Forward status bits
%MW@.chann.5:X0	STS_AT_RUNNING		Autotuning in progress
%MW@.chann.5:X1	STS_TR_S		Tracking in progress
%MW@.chann.5:X2			
%MW@.chann.5:X3	STS_M_A		PID operating mode state
%MW@.chann.5:X4	STS_RAISE1		Opening command
%MW@.chann.5:X5	STS_LOWER1		Closing command
%MW@.chann.5:X6	STS_RAISE2		Output 2 branch opening command
%MW@.chann.5:X7	STS_LOWER2		Output 2 branch closing command
%MW@.chann.5:X8	STS_OUT_L_LIM		

Address	Parameter name	Default value	Comment
%MW@.chann.5:X9	STS_OUT_H_LIM		
%MW@.chann.5:X10	STS_TOP_NEXT_CYCLE		Sampling pulse in next cycle
%MW@.chann.5:X11	STS_TOP_CURRENT_CYCLE		Sampling pulse in current cycle
%MW@.chann.5:X12	STS_FF_Sim		Status of FF process value simulation
%MW@.chann.6	STATUS3	Not applicable	Word containing the various Servo status bits
%MW@.chann.6:X0	POT_VAL1		Servo operation with feedback
%MW@.chann.6:X1	POT_VAL2		Servo operation with feedback (Reserved)
%MW@.chann.6:X2	RAISE STOP1		Opening stop reached on Servomotor (Reserved)
%MW@.chann.6:X3	LOWER STOP1		Closing stop reached on Servomotor (Reserved)
%MW@.chann.6:X4	RAISE STOP2		Opening stop reached on Servomotor (Reserved)
%MW@.chann.6:X5	LOWER STOP2		Closing stop reached on Servomotor (Reserved)
%MW@.chann.7	STATUS4	Not applicable	Word containing the precise diagnostics of the various warnings (FF setpoint process value)
%MW@.chann.7:X0	SP_INF_WARN		Control warning for SP_INF and SUP parameters
%MW@.chann.7:X1	Xi_WARN		Control warning for Xi parameters
%MW@.chann.7:X2	Yi_WARN		Control warning for Yi parameters
%MW@.chann.7:X6	OVER_TOT_WARN		Totalizing overflow warning
%MW@.chann.7:X8	INP_MINR1_WARN		Control warning for INP_INFR1 and INP_SUPR1 parameters
%MW@.chann.7:X9	INP_MINR2_WARN		Control warning for INP_INFR2 and INP_SUPR2 parameters
%MW@.chann.7:X10	RATIO_WARN		Control warning for RATIO_MIN and MAX parameters

Address	Parameter name	Default value	Comment
%MW@.chann.7:X11	SP_CALC_WARN		Setpoint calculation warning
%MW@.chann.7:X12	SP_FLOAT_WARN		Setpoint floating point warning
%MW@.chann.7:X13	FF_CALC_WARN		Feed Forward calculation warning
%MW@.chann.7:X14	FF_FLOAT_WARN		Feed Forward floating point warning
%MW@.chann.8	STATUS5	Not applicable	Word containing the autotuning diagnostics
%MW@.chann.8:X0	AT_FAILED		Autotuning failed
%MW@.chann.8:X1	AT_ABORTED		Autotuning diagnostics aborted
%MW@.chann.8:X2	AT_ERR_PARAM		Autotuning diagnostics parameter error
%MW@.chann.8:X3	AT_ERR_PWF_OR_EFB_FAILURE		Autotuning diagnostics system error or power outage
%MW@.chann.8:X4	AT_ERR_SATUR		Autotuning diagnostics saturation of the process value
%MW@.chann.8:X5	AT_ERR_DV_TOO_SMALL		Autotuning diagnostics process value deviation too small
%MW@.chann.8:X6	AT_ERR_TSAMP_HIGH		Autotuning diagnostics sampling period too long
%MW@.chann.8:X7	AT_ERR_INCONSISTENT_RESPONSE		Autotuning diagnostics inconsistent response
%MW@.chann.8:X8	AT_ERR_NOT_STAB_INIT		Autotuning diagnostics process value not initially stable
%MW@.chann.8:X9	AT_ERR_TMAX_TOO_SMALL		Autotuning diagnostics duration of the step function too short
%MW@.chann.8:X10	AT_ERR_NOISE_TOO_HIGH		Autotuning diagnostics process value noise too high
%MW@.chann.8:X11	AT_ERR_TMAX_TOO_HIGH		Autotuning diagnostics duration of the step function too long
%MW@.chann.8:X12	AT_WARN_OVERSHOOT		Autotuning diagnostics overshoot greater than 10%

---

Address	Parameter name	Default value	Comment
%MW@.chann.8:X13	AT_WARN_UNDERSHOOT		Autotuning diagnostics undershoot too great
%MW@.chann.8:X14	AT_WARN_UNSYMMETRICAL_PLANT		Autotuning diagnostics process too unsymmetrical
%MW@.chann.8:X15	AT_WARN_INTEGRATING_PLANT		Autotuning diagnostics integrating process

---

### 6.3-3 Process control language objects

Address	Parameter name	Default value	Comment
%MW@.chann.14	PV_SIM	Not applicable	Simulated PV value
%MW@.chann.15	FF_SIM	Not applicable	Simulated Feed Forward input
%MF@.chann.16	T_ECH	0.3	Sampling period
%MF@.chann.18	OUT1	Not applicable	Value of output 1 of Hot/Cool or Split/Range
%MF@.chann.20	OUT2	Not applicable	Value of output 2 of Hot/Cool or Split/Range
%MF@.chann.22	OUTD	Not applicable	Command variation value
%MF@.chann.24	OUTFF	Not applicable	Value of Feed Forward action in Physical scale
%MF@.chann.26	OUT_MAN	Not applicable	Command value
%MF@.chann.28	DEV	Not applicable	Process value setpoint deviation
%MF@.chann.30	PV	Not applicable	PV value in Physical scale
%MF@.chann.32	SP	Not applicable	Setpoint value in Physical scale
%MF@.chann.34	PV_INF	0.	Process value lower limit
%MF@.chann.36	PV_SUP	100.	Process value upper limit
%MF@.chann.38	KP	1.0	Proportional coefficient
%MF@.chann.40	TI	0.0	Integral time
%MF@.chann.42	TD	0.0	Derivative time
%MF@.chann.44	OUTBIAS	0.0	Bias on PID controller output
%MF@.chann.46	INT_BAND	0.0	Integral band
%MF@.chann.48	DBAND	0.0	Dead band on the deviation
%MF@.chann.50	KD	10.0	Derivative filtering
%MF@.chann.52	OUTRATE	0.0	Limitation of output 1 variation speed
%MF@.chann.54	OUTRATE2	0.0	Limitation of output 2 variation speed
%MF@.chann.56	OUT1_INF	0.	Output 1 lower limit

Address	Parameter name	Default value	Comment
%MF@.chann.58	OUT1_SUP	100.0	Output 1 upper limit
%MF@.chann.60	SP_INF	0.0	Setpoint lower limit
%MF@.chann.62	SP_SUP	100.	Setpoint upper limit
%MF@.chann.64	OUT2_INF	0.	Output 2 lower limit
%MF@.chann.66	OUT2_SUP	100.	Output 2 upper limit
%MF@.chann.68	OUT1_TH1	0.	Threshold 1 of output 1 of Hot/Cool or Split/Range
%MF@.chann.70	OUT1_TH2	50.0	Threshold 2 of output 1 of Hot/Cool or Split/Range
%MF@.chann.72	OUT2_TH1	50.0	Threshold 1 of output 2 of Hot/Cool or Split/Range
%MF@.chann.74	OUT2_TH2	100.	Threshold 2 of output 2 of Hot/Cool or Split/Range
%MF@.chann.76	PV_LL	5.	PV very low threshold
%MF@.chann.78	PV_L	5.	PV low threshold
%MF@.chann.80	PV_H	95.	PV high threshold
%MF@.chann.82	PV_HH	95.	PV very high threshold
%MF@.chann.84	RATIO	1.0	Ratio value
%MF@.chann.86	RATIO_MIN	0.	Min. ratio value
%MF@.chann.88	RATIO_MAX	100	Max. ratio value
%MF@.chann.90	RATIO_BIAS	0	Bias ratio value
%MF@.chann.92	ONOFF_L	-5	Low threshold of ONOFF controller
%MF@.chann.94	ONOFF_H	5	High threshold of ONOFF controller
%MF@.chann.96	HYST1	0.0	Hysteresis of 3-state ONOFF controller
%MF@.chann.98	DEV_L	-5	Low threshold of deviation
%MF@.chann.100	DEV_H	5	High threshold of deviation

Address	Parameter name	Default value	Comment
%MF@.chann.102	T_FILTER	0.0	Process value filter time
%MF@.chann.104	K_FILTER	1.0	Multiplying coefficient on PV filtering
%MF@.chann.106	FILT_OUT	Not applicable	Filter output value
%MF@.chann.108	SQRT_OUT	Not applicable	Square root output value
%MF@.chann.110	E2_IN	1428	Abscissa of first point of segment S2
%MF@.chann.112	E3_IN	2857	Abscissa of first point of segment S3
%MF@.chann.114	E4_IN	4285	Abscissa of first point of segment S4
%MF@.chann.116	E5_IN	5714	Abscissa of first point of segment S5
%MF@.chann.118	E6_IN	7143	Abscissa of first point of segment S6
%MF@.chann.120	E7_IN	8571	Abscissa of first point of segment S7
%MF@.chann.122	E2_OUT	14.28	Ordinate of first point of segment S2
%MF@.chann.124	E3_OUT	28.57	Ordinate of first point of segment S3
%MF@.chann.126	E4_OUT	42.85	Ordinate of first point of segment S4
%MF@.chann.128	E5_OUT	57.14	Ordinate of first point of segment S5
%MF@.chann.130	E6_OUT	71.43	Ordinate of first point of segment S6
%MF@.chann.132	E7_OUT	8571	Ordinate of first point of segment S7
%MF@.chann.134	THLD	1E+8	Totalizing limit
%MF@.chann.136	R_RATE	0.0	Setpoint increase speed limit
%MF@.chann.138	D_RATE	0.0	Setpoint decrease speed limit
%MF@.chann.140	SPEED_LIM_OUT	Not applicable	Setpoint speed limiter output value
%MF@.chann.142	INP_MINR1	0.0	Low scale of setpoint R1



Address	Parameter name	Default value	Comment
%MF@.chann.144	INP_MAXR1	100.0	High scale of setpoint R1
%MF@.chann.146	INP_MINR2	0.0	Low scale of setpoint R2
%MF@.chann.148	INP_MAXR2	100.0	High scale of setpoint R2
%MF@.chann.150	T1_FF	0.0	Feed Forward process value filter time
%MF@.chann.152	T2_FF	0.0	Feed Forward process value filter time
%MF@.chann.154	OUT_FF_INF	0.	Lower limit of Feed Forward action
%MF@.chann.156	OUT_FF_SUP	100.	Upper limit of Feed Forward action
%MF@.chann.158	T_MOTOR1	10.	Opening time for valve controlled by Servomotor
%MF@.chann.160	T_MINI1	0.	Minimum opening time for valve controlled by Servomotor
%MF@.chann.162	T_MOTOR2	10.	Opening time for valve controlled by Servomotor
%MF@.chann.164	T_MINI2	0.	Minimum opening time for valve controlled by Servomotor
%MF@.chann.166	AT_STEP	10	Amplitude of the autotuning step function
%MF@.chann.168	AT_TMAX	100	Duration of the autotuning step function
%MF@.chann.170	AT_PERF	0.5	Autotuning stability criterion
%MF@.chann.172	KP_PREV	Not applicable	Value of proportional coefficient before autotuning
%MF@.chann.174	TI_PREV	Not applicable	Value of integral coefficient before autotuning
%MF@.chann.176	TD_PREV	Not applicable	Value of derivative coefficient before autotuning
%MF@.chann.178	KS	1.0	IMC static gain
%MF@.chann.180	T1	1.0	Time constant in OL

---

<b>Address</b>	<b>Parameter name</b>	<b>Default value</b>	<b>Comment</b>
%MF@.chann.182	T_DELAY	0.0	Current pure time delay
%MF@.chann.184	CC_PERF	0.1	OL / CL time ratio

---

## 6.4 Language objects associated with a 3 single loop channel

### 6.4-1 Configuration language objects

Address	Parameter name	Default value	Comment
%KW@.chann.0	CONFIG_0_B1	Not applicable	Word containing the various Process value configuration bits
%KW@.chann.0:X0	Filtrage	Non-configurable(0)	Process value branch filtering function
%KW@.chann.0:X1	Générateur de fonction	Non-configurable(0)	Process value branch function generator
%KW@.chann.0:X2	Totalisateur	Absent(0)	Process value branch process value branch
%KW@.chann.0:X3	Racine Carrée	Absent(0)	Process value branch root function
%KW@.chann.0:X4	Alarmes	Present	Process value branch alarm function
%KW@.chann.0:X8	PV_CLIP	Absent(0)	Peak limiting (or not) of process value
%KW@.chann.0:X9	EXTRAPOL	Non-configurable(0)	Extrapolation of function generator
%KW@.chann.0:X10	PV_UNI_BIP	Unipolar (0)	Process value type unipolar/bipolar
%KW@.chann.0:X11	PV_EXTERN	Absent (0)	Select Standard (0) / External (1) Process value
%KW@.chann.0:X12	VALID_C1	Enabled(1)	Loop used(1) / not used(0)
%KW@.chann.0:X13	Totalisateur: Unité mesure	1	(X13=0, X14 =0): phys/ms (X13=1, X14 =0): phys/s
%KW@.chann.0:X14	Totalisateur: Unité mesure	0	(X13=0, X14 =1): phys/mn (X13=1, X14 =1): phys/h
%KW@.chann.1	CONFIG_1	Not applicable	Word containing the various Setpoint configuration bits
%KW@.chann.1:X0	SP_Simple	Selected(1)	Type of setpoint selected : Simple
%KW@.chann.1:X1	SP_Sélection	Non-configurable(0)	Type of setpoint selected : Selection
%KW@.chann.1:X2	Speed_Limiteur	Not selected(0)	Setpoint speed limiter
%KW@.chann.1:X3	SP_SPP	Not selected(0)	Type of setpoint selected : Programmer

Address	Parameter name	Default value	Comment
%KW@.chann.1:X4	RL/L	Remote local (0)	Speed limiter on local setpoint or in remote/local mode
%KW@.chann.1:X8	Sel_min	Non-configurable(0)	Function selected for Selection setpoint type
%KW@.chann.1:X9	Sel_max	Non-configurable(0)	Function selected for Selection setpoint type
%KW@.chann.1:X10	Sel_switch	Non-configurable(0)	Function selected for Selection setpoint type
%KW@.chann.1:X11	R/L_INIT	Local (1)	Initial value of selected setpoint Remote/Local
%KW@.chann.1:X12	R1/R2_INIT	Non-configurable	Initial value of selected setpoint state
%KW@.chann.1:X13	SP_Ratio	Non-configurable(0)	Type of setpoint selected : Ratio
%KW@.chann.1:X14	SP_Limiteur	Not selected (0)	Setpoint limiter (eg. Param_SP)
%KW@.chann.1:X15	SP_Folw	Non-tracking setpoint (0)	Tracking setpoint
%KW@.chann.2	CONFIG_2_B1	Not applicable	Word containing the various loop controller and FF configuration bits
%KW@.chann.2:X0	PID	Present	PID function of loop controller branch
%KW@.chann.2:X1	ONOFF 2	Absent(0)	Controller 2-state ONOFF branch
%KW@.chann.2:X2	ONOFF 3	Absent(0)	Controller 3-state ONOFF branch
%KW@.chann.2:X3	SPLRG/ChFroid	Non-configurable(0)	OR presence bits Hot/Cool and Split/Range
%KW@.chann.2:X4	Split/Range	Non-configurable(0)	Loop controller branch Split/Range function
%KW@.chann.2:X5	Chaud/Froid	Non-configurable(0)	Loop controller branch Hot/Cool function
%KW@.chann.2:X6	Alarmes_DEV	Present	Alarm function on loop controller branch deviation
%KW@.chann.2:X7	Feed Forward	Non-configurable(0)	Presence of a Feed Forward input

Address	Parameter name	Default value	Comment
%KW@.chann.2:X8	BUMP	With bumps (1)	Management of bumps on change of operating mode
%KW@.chann.2:X9	PV_DEV	On PV (0)	Type of derivative action
%KW@.chann.2:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.2:X11	REV_DIR	Inverse	Type of loop controller PID action (0) action
%KW@.chann.2:X12	MANU/AUTO_INIT	Manu (0)	Initial value of loop controller operating mode
%KW@.chann.2:X13	Lead Lag	Non-configurable(0)	Feed Forward branch Lead Lag function
%KW@.chann.2:X14	FF_UNI_BIP	Non-configurable(0)	Type of Feed Forward PV (unipolar/bipolar)
%KW@.chann.2:X15	IMC	Absent(0)	Loop controller branch IMC function
%KW@.chann.3	CONFIG_3_B1	Not applicable	Word containing the various output configuration bits
%KW@.chann.3:X0	Servo	Not selected	Type of output selected : Servo
%KW@.chann.3:X1	Servo2	Non-configurable(0)	Type of output selected : Servo
%KW@.chann.3:X2	Analogique1	Selected	Type of output selected : Analog
%KW@.chann.3:X3	Analogique2	Non-configurable(0)	Type of output selected : Analog
%KW@.chann.3:X4	PWM1	Not selected	Type of output selected : PWM
%KW@.chann.3:X5	PWM2	Non-configurable(0)	Type of output selected : PWM
%KW@.chann.3:X8	POT_REV1	Direct (0)	Servo feedback direction
%KW@.chann.3:X9	POT_REV2	Non-configurable(0)	Servo feedback direction
%KW@.chann.3:X10	POT_VAL1_INIT	No (0)	Existence of Servo feedback
%KW@.chann.3:X11	POT_VAL2_INIT	Non-configurable(0)	Existence of Servo feedback (Reserved)
%KW@.chann.3:X12	ANALOG1_UNI_BIP	Unipolar	Type of analog output unipolar/bipolar

Address	Parameter name	Default value	Comment
%KW@.chann.3:X13	ANALOG2_UNI_BIP	Non-configurable(0)	Type of analog output unipolar/bipolar
%KW@.chann.4	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.8	Unité de la boucle		Loop unit
%KW@.chann.11	IDEM BOUCLE 1 %KW0		PV B2 Functions not used have their bit at 0
%KW@.chann.12	IDEM BOUCLE 1 %KW1		Setpoint B2
%KW@.chann.13	IDEM BOUCLE 1 %KW2		Loop controller and FF B2
%KW@.chann.14	IDEM BOUCLE 1 %KW3		Output B2
%KW@.chann.15	IDEM BOUCLE 1 %KW4		Loop i where i [0;9] Loop name
%KW@.chann.19	IDEM BOUCLE 1 %KW8		Loop unit
%KW@.chann.22	IDEM BOUCLE 1 %KW0		PV B3 Functions not used have their bit at 0
%KW@.chann.23	IDEM BOUCLE 1 %KW1		Setpoint B3
%KW@.chann.24	IDEM BOUCLE 1 %KW2		Loop controller and FF B3
%KW@.chann.25	IDEM BOUCLE 1 %KW3		Output B3
%KW@.chann.26	IDEM BOUCLE 1 %KW4		Loop i where i [0;9] Loop name
%KW@.chann.30	IDEM BOUCLE 1 %KW8		Loop unit

### 6.4-2 Diagnostic and default language objects

Address	Parameter name	Default value	Comment
%MW@.chann.0	Echange en cours		
%MW@.chann.1	Compte Rendu		
%MW@.chann.2	STATUS_VOIE1		Channel status defined by FM standard
%MW@.chann.2:X0	STS_DEF_EXTERNE		
%MW@.chann.2:X1	STS_DEF_DEPASS_GAMME		
%MW@.chann.2:X2	STS_DEF_BORNIER		
%MW@.chann.2:X3	STS_DEF_EXT_PROT		
%MW@.chann.2:X4	STS_DEFAULT_INTERNE		Serious internal fault on loop 1
%MW@.chann.2:X5	STS_DEFAULT_CONFIG		Configuration fault on loop 1
%MW@.chann.2:X6	STS_DEFAULT_COMMUNIC		
%MW@.chann.2:X7	WARN		Warning sigma
%MW@.chann.2:X8	STS_ERR_CALC_CORR_B1		Loop controller branch calculation error on loop 1
%MW@.chann.2:X9	STS_ERR_FLOT_CORR_B1		Loop controller branch floating point error on loop 1
%MW@.chann.2:X10	STS_ERR_CALC_PV_B1		PV branch calculation error on loop 1
%MW@.chann.2:X11	STS_ERR_FLOT_PV_B1		PV branch floating point error on loop 1
%MW@.chann.2:X12	STS_ERR_CALC_OUT_B1		OUT branch calculation error on loop 1
%MW@.chann.2:X13	STS_ERR_FLOT_OUT_B1		OUT branch floating point error on loop 1
%MW@.chann.2:X14	STS_ERR_CALC_OUT_B1		OUT1 branch incorrect scale on loop 1
%MW@.chann.2:X15	STS_ERR_SCALE_PV_B1		PV branch incorrect scale on loop 1
%MW@.chann.3	STATUS_VOIE2	Not applicable	Channel status defined by FM standard
%MW@.chann.3:X4	STS_DEFAULT_INTERNE		Serious internal fault on loop 2
%MW@.chann.3:X5	STS_DEFAULT_CONFIG		Configuration fault on loop 2
%MW@.chann.3:X8	STS_ERR_CALC_CORR_B2		Loop controller branch calculation error on loop 2

Address	Parameter name	Default value	Comment
%MW@.chann.3:X9	STS_ERR_FLOT_CORR_B2		Loop controller branch floating point error on loop 2
%MW@.chann.3:X10	STS_ERR_CALC_PV_B2		PV branch calculation error on loop 2
%MW@.chann.3:X11	STS_ERR_FLOT_PV_B2		PV branch floating point error on loop 2
%MW@.chann.3:X12	STS_ERR_CALC_OUT_B2		OUT branch calculation error on loop 2
%MW@.chann.3:X13	STS_ERR_FLOT_OUT_B2		OUT branch floating point error on loop 2
%MW@.chann.3:X14	STS_ERR_SCALE_OUT1_B2		OUT1 branch incorrect scale on loop 2
%MW@.chann.3:X15	STS_ERR_SCALE_PV_B2		PV branch incorrect scale on loop 2
%MW@.chann.4		Not applicable	
%MW@.chann.4:X4	STS_DEFAULT_INTERNE		Serious internal fault on loop 3
%MW@.chann.4:X5	STS_DEFAULT_CONFIG		Configuration fault on loop 3
%MW@.chann.4:X8	STS_ERR_CALC_CORR_B3		Loop controller branch calculation error on loop 3
%MW@.chann.4:X9	STS_ERR_FLOT_CORR_B3		Loop controller branch floating point error on loop 3
%MW@.chann.4:X10	STS_ERR_CALC_PV_B3		PV branch calculation error on loop 3
%MW@.chann.4:X11	STS_ERR_FLOT_PV_B3		PV branch floating point error on loop 3
%MW@.chann.4:X12	STS_ERR_CALC_OUT_B3		OUT branch calculation error on loop 3
%MW@.chann.4:X13	STS_ERR_FLOT_OUT_B3		OUT branch floating point error on loop 3
%MW@.chann.4:X14	STS_ERR_SCALE_OUT1_B3		OUT1 branch incorrect scale on loop 3
%MW@.chann.4:X15	STS_ERR_SCALE_PV_B3		PV branch incorrect scale
%MW@.chann.5	STATUS0_B1	Not applicable	Word containing the various Process value/ Setpoint status bits
%MW@.chann.5:X0	HOLD_TOT_B1		State of totalizing function



Address	Parameter name	Default value	Comment
%MW@.chann.5:X1	PV Simulée_B1		Status of process value simulation
%MW@.chann.5:X2	PV_H_LIM_B1		Upper limit on process value branch(PV_SUP)
%MW@.chann.5:X3	PV_L_LIM_B1		Lower limit on process value branch (PV_INF)
%MW@.chann.5:X4	SP_H_LIM_B1		Upper limit on setpoint branch
%MW@.chann.5:X5	SP_B_LIM_B1		Lower limit on setpoint branch
%MW@.chann.5:X6	R/L_B1	R/L Init	State of selected setpoint Remote/Local
%MW@.chann.5:X7	TR_S_B1		Tracking state bit
%MW@.chann.5:X8	ALARMES_B1		Process value alarms sigma
%MW@.chann.5:X9	HH_B1		Very high alarm
%MW@.chann.5:X10	H_B1		High alarm
%MW@.chann.5:X11	L_B1		Low alarm
%MW@.chann.5:X12	LL_B1		Very low alarm
%MW@.chann.5:X13	DEV_H_B1		High threshold of Process value/Setpoint deviation (>0)
%MW@.chann.5:X14	DEV_L_B1		Low threshold of Process value/Setpoint deviation (<0)
%MW@.chann.5:X15	STS_THLD_DONE_B1		Totalizing threshold reached
%MW@.chann.6	STATUS1_B1	Not applicable	Word containing the various loop controller output +D119 status bits
%MW@.chann.6:X0	AT_EN_COURS_B1		Autotuning in progress (common to all 3 loops)
%MW@.chann.6:X1	MANU/AUTO_B1		PID operating mode state
%MW@.chann.6:X2	RAISE 1_B1		Opening command
%MW@.chann.6:X3	LOWER 1_B1		Closing command
%MW@.chann.6:X4	LIM_PID_SUP_B1		The calculated PID output is greater than or equal to OUT_SUP
%MW@.chann.6:X5	LIM_PID_INF_B1		The calculated PID output is less than or equal to OUT_INF

Address	Parameter name	Default value	Comment
%MW@.chann.6:X6	POT_VAL1_B1		Servo operation with feedback
%MW@.chann.6:X7	RAISE STOP1_B1		Opening stop reached on Servomotor (Reserved)
%MW@.chann.6:X8	LOWER STOP1_B1		Closing stop reached on Servomotor (Reserved)
%MW@.chann.6:X9	STS_TOP_NEXT_CYCLE_B1		Sampling pulse in next cycle
%MW@.chann.6:X10	STS_TOP_CURRENT_CYCLE_B1		Sampling pulse in current cycle
%MW@.chann.6:X11	OVER_TOT_WARN_B1		Overflow totalizing warning (T_MOTOR1_WARN canceled)
%MW@.chann.6:X12	INP_MINR1_WARN_B1		Control warning for INP_MINR1 and MAXR1 parameters
%MW@.chann.6:X13	SP_INF_WARN_B1		Control warning for SP_INF and SUP parameters
%MW@.chann.6:X14	CALC_SP_WARN_B1		Setpoint calculation error warning
%MW@.chann.6:X15	FLOAT_SP_WARN_B1		Floating point error in the setpoint warning
%MW@.chann.7	STATUS1_B2	Not applicable	Word containing the various Process value/ Setpoint status bits
%MW@.chann.7:X0	HOLD_TOT_B2		State of totalizing function
%MW@.chann.7:X1	PV Simulée_B2		Status of process value simulation
%MW@.chann.7:X2	PV_H_LIM_B2		Upper limit on process value branch (PV_SUP)
%MW@.chann.7:X3	PV_L_LIM_B2		Lower limit on process value branch (PV_INF)
%MW@.chann.7:X4	SP_H_LIM_B2		Upper limit on setpoint branch
%MW@.chann.7:X5	SP_B_LIM_B2		Lower limit on setpoint branch
%MW@.chann.7:X6	R/L_B2	R/L Init	State of selected setpoint Remote/Local

Address	Parameter name	Default value	Comment
%MW@.chann.7:X7	TR_S_B2		Tracking state bit
%MW@.chann.7:X8	ALARMES_B2		Process value alarms sigma
%MW@.chann.7:X9	HH_B2		Very high alarm
%MW@.chann.7:X10	H_B2		High alarm
%MW@.chann.7:X11	L_B2		Low alarm
%MW@.chann.7:X12	LL_B2		Very low alarm
%MW@.chann.7:X13	DEV_H_B2		High threshold of Process value Setpoint deviation (>0)
%MW@.chann.7:X14	DEV_L_B2		Low threshold of Process value Setpoint deviation (<0)
%MW@.chann.7:X15	STS_THLD_DONE_B2		Totalizing threshold reached
%MW@.chann.8	STATUS2_B2	Not applicable	Word containing the various loop controller/output status bits
%MW@.chann.8:X0	AT_EN_COURS_B2		Autotuning in progress (common to all 3 loops)
%MW@.chann.8:X1	MANU/AUTO_B2		PID operating mode state
%MW@.chann.8:X2	RAISE 1_B2		Opening command
%MW@.chann.8:X3	LOWER 1_B2		Closing command
%MW@.chann.8:X4	LIM_PID_SUP_B2		The calculated PID output is greater than or equal to OUT_SUP
%MW@.chann.8:X5	LIM_PID_INF_B2		The calculated PID output is less than or equal to OUT_INF
%MW@.chann.8:X6	POT_VAL1_B2		Servo operation with feedback
%MW@.chann.8:X7	RAISE STOP1_B2		Opening step reached on Servomotor (Reserved)
%MW@.chann.8:X8	LOWER STOP1_B2		Closing step reached on Servomotor (Reserved)
%MW@.chann.8:X9	STS_TOP_NEXT_CYCLE_B2		Sampling pulse in next cycle
%MW@.chann.8:X10	STS_TOP_CURRENT_CYCLE_B2		Sampling pulse in current cycle
%MW@.chann.8:X11	OVER_TOT_WARN_B2		Overflow totalizing warning (T_MOTOR1_WARN canceled)

Address	Parameter name	Default value	Comment
%MW@.chann.8:X12	INP_MINR1_WARN_B2		Control warning for INP_MINR1 and MAXR1 parameters
%MW@.chann.8:X13	SP_INF_WARN_B2		Control warning for SP_INF and SUP parameters
%MW@.chann.8:X14	CALC_SP_WARN_B2		Calculation error in setpoint warning
%MW@.chann.8:X15	FLOAT_SP_WARN_B2		Floating point error in setpoint warning
%MW@.chann.9:X0	HOLD_TOT_B3		State of totalizing function
%MW@.chann.9:X1	PV Simulée_B3		Status of process value simulation
%MW@.chann.9:X2	PV_H_LIM_B3		Upper limit on process value branch (PV_SUP)
%MW@.chann.9:X3	PV_L_LIM_B3		Lower limit on process value branch (PV_INF)
%MW@.chann.9:X4	SP_H_LIM_B3		Upper limit on setpoint branch
%MW@.chann.9:X5	SP_B_LIM_B3		Lower limit on setpoint branch
%MW@.chann.9:X6	R/L_B3	R/L Init	State of selected setpoint Remote/Local
%MW@.chann.9:X7	TR_S_B3		Tracking state bit
%MW@.chann.9:X8	ALARME_B3		Process value alarms sigma
%MW@.chann.9:X9	HH_B3		Very high alarm
%MW@.chann.9:X10	H_B3		High alarm
%MW@.chann.9:X11	L_B3		Low alarm
%MW@.chann.9:X12	LL_B3		Very low alarm
%MW@.chann.9:X13	DEV_H_B3		High threshold of Process value Setpoint deviation (>0)
%MW@.chann.9:X14	DEV_L_B3		Low threshold of Process value Setpoint deviation (<0)
%MW@.chann.9:X15	STS_THLD_DONE_B3		Totalizing threshold reached
%MW@.chann.10	STATUS2_B3	Not applicable	Word containing the various Process value/ Setpoint status bits
%MW@.chann.10:X0	AT_EN_COURS_B3		Autotuning in progress (common to all 3 loops)

Address	Parameter name	Default value	Comment
%MW@.chann.10:X1	MANU/AUTO_B3		PID operating mode state
%MW@.chann.10:X2	RAISE 1_B3		Opening command
%MW@.chann.10:X3	LOWER 1_B3		Closing command
%MW@.chann.10:X4	LIM_PID_SUP_B3		The calculated PID output is greater than or equal to OUT_SUP
%MW@.chann.10:X5	LIM_PID_INF_B3		The calculated PID output is less than or equal to OUT_INF
%MW@.chann.10:X6	POT_VAL1_B3		Servo operation with feedback
%MW@.chann.10:X7	RAISE STOP1_B3		Opening step reached on Servomotor (Reserved)
%MW@.chann.10:X8	LOWER STOP1_B3		Closing step reached on Servomotor (Reserved)
%MW@.chann.10:X9	STS_TOP_NEXT_CYCLE_B3		Sampling pulse in next cycle
%MW@.chann.10:X10	STS_TOP_CURRENT_CYCLE_B3		Sampling pulse in current cycle
%MW@.chann.10:X11	OVER_TOT_WARN_B3		Overflow totalizing warning (T_MOTOR1_WARN canceled)
%MW@.chann.10:X12	INP_MINR1_WARN_B3		Control warning for INP_MINR1 and MAXR1 parameters
%MW@.chann.10:X13	SP_INF_WARN_B3		Control warning for SP_INF and SUP parameters
%MW@.chann.10:X14	CALC_SP_WARN_B3		Setpoint calculation error warning
%MW@.chann.10:X15	FLOAT_SP_WARN_B3		Floating point error in the setpoint warning
%MW@.chann.11	STATUS4	Not applicable	Word containing the precise diagnostics of the various warnings
%MW@.chann.11:X0	AT_FAILED		Autotuning failed
%MW@.chann.11:X1	AT_ABORTED		Autotuning diagnostics interrupted
%MW@.chann.11:X2	AT_ERR_PARAM		Autotuning diagnostics parameter error
%MW@.chann.11:X3	AT_ERR_PWF_OR_EFB_FAILURE		Autotuning diagnostics system error or power outage

Address	Parameter name	Default value	Comment
%MW@.chann.11:X4	AT_ERR_SATUR		Autotuning diagnostics saturation of the process value
%MW@.chann.11:X5	AT_ERR_DV_TOO_SMALL		Autotuning diagnostics process value deviation too small
%MW@.chann.11:X6	AT_ERR_TSAMP_HIGH		Autotuning diagnostics sampling period too long
%MW@.chann.11:X7	AT_ERR_INCONSISTENT_RESPONSE		Autotuning diagnostics inconsistent response
%MW@.chann.11:X8	AT_ERR_NOT_STAB_INIT		Autotuning diagnostics process value not initially stable
%MW@.chann.11:X9	AT_ERR_TMAX_TOO_SMALL		Autotuning diagnostics duration of the step function too short
%MW@.chann.11:X10	AT_ERR_NOISE_TOO_HIGH		Autotuning diagnostics process value noise too high
%MW@.chann.11:X11	AT_ERR_TMAX_TOO_HIGH		Autotuning diagnostics duration of the step function too long
%MW@.chann.11:X12	AT_WARN_OVERSHOOT		Autotuning diagnostics overshoot greater than 10%
%MW@.chann.11:X13	AT_WARN_UNDERSHOOT		Autotuning diagnostics undershoot too great
%MW@.chann.11:X14	AT_WARN_UNSYMMETRICAL_PLANT		Autotuning diagnostics process too unsymmetrical
%MW@.chann.11:X15	AT_WARN_INTEGRATING_PLANT		Autotuning diagnostics integrating process
%MW@.chann.13	Ordre Commande		
%MD@.chann.14	Paramètre Commande		

### 6.4-3 Process control language objects

Address	Parameter name	Default value	Comment
%MF@.chann.16	AT_STEP	10	Amplitude of the autotuning step function
%MF@.chann.18	AT_TMAX	100	Duration of the autotuning step function
%MF@.chann.20	AT_PERF	0.5	Autotuning stability criteria
%MF@.chann.22	T_ECH_B1	0.3	Sampling period
%MF@.chann.24	OUT_MAN_B1	Not applicable	Command value
%MF@.chann.26	DEV_B1	Not applicable	PV setpoint deviation
%MF@.chann.28	PV_B1	Not applicable	PV value in Physical scale
%MF@.chann.30	SP_B1	Not applicable	Setpoint value in Physical scale
%MF@.chann.32	PV_INF_B1	0.	Process value lower limit
%MF@.chann.34	PV_SUP_B1	100.	Process value upper limit
%MF@.chann.36	KP_B1	1.0	Proportional coefficient
%MF@.chann.38	TI_B1	0.0	Integral time
%MF@.chann.40	TD_B1	0.0	Derivative time
%MF@.chann.42	OUTBIAS_B1	0.0	Bias on PID controller output
%MF@.chann.44	INT_BAND_B1	0.0	Integral band
%MF@.chann.46	DBAND_B1	0.0	Dead band on the deviation
%MF@.chann.48	OUTRATE_B1	0.0	Limitation of output variation speed
%MF@.chann.50	OUT1_INF_B1	0.	Output 1 lower limit
%MF@.chann.52	OUT1_SUP_B1	100.0	Output 1 upper limit
%MF@.chann.54	SP_INF_B1	0.0	Setpoint lower limit
%MF@.chann.56	SP_SUP_B1	100.	Setpoint upper limit
%MF@.chann.58	PV_LL_B1	5.	PV very low threshold
%MF@.chann.60	PV_L_B1	5.	PV low threshold
%MF@.chann.62	PV_H_B1	95.	PV high threshold

Address	Parameter name	Default value	Comment
%MF@.chann.64	PV_HH_B1	95.	PV very high threshold
%MF@.chann.66	ONOFF_L_B1	-5	Low threshold of ONOFF controller
%MF@.chann.68	ONOFF_H_B1	5	High threshold of ONOFF controller
%MF@.chann.70	HYST1_B1	0.0	Hysteresis of 3-statee ONOFF controller
%MF@.chann.72	DEV_L_B1	0.	Low threshold of deviation
%MF@.chann.74	DEV_H_B1	0.	High threshold of deviation
%MF@.chann.76	THLD_B1	1E+8	Totalizing limit
%MF@.chann.78	R_RATE_B1	0.0	Speed limiter increase speed value
%MF@.chann.80	D_RATE_B1	0.0	Speed limiter decrease speed value
%MF@.chann.82	SPEED_LIM_OUT_B1	Not applicable	Speed limiter output value
%MF@.chann.84	INP_MINR1_B1	0.0	Low scale of setpoint R1
%MF@.chann.86	INP_MAXR1_B1	100.0	High scale of setpoint R1
%MF@.chann.88	T_MOTOR1_B1	10.	Opening time for valve controlled by servomotor
%MF@.chann.90	T_MINI1_B1	0.	Minimum opening time for valve controlled by Servomotor
%MF@.chann.92	KP_PREV_B1	Not applicable	Value of proportional coefficient before autotuning
%MF@.chann.94	TI_PREV_B1	Not applicable	Value of integral coefficient before autotuning
%MF@.chann.96	TD_PREV_B1	Not applicable	Value of derivative coefficient before autotuning
%MF@.chann.98	T_ECH_B2	0.3	Sampling period
%MF@.chann.100	OUT_MAN_B2	Not applicable	Command value
%MF@.chann.102	DEV_B2	Not applicable	PV setpoint deviation



Address	Parameter name	Default value	Comment
%MF@.chann.104	PV_B2	Not applicable	PV value in Physical scale
%MF@.chann.106	SP_B2	Not applicable	Setpoint value in Physical scale
%MF@.chann.108	PV_INF_B2	0.	Process value lower limit
%MF@.chann.110	PV_SUP_B2	100.	Process value upper limit
%MF@.chann.112	KP_B2	1.0	Proportional coefficient
%MF@.chann.114	TI_B2	0.0	Integral time
%MF@.chann.116	TD_B2	0.0	Derivative time
%MF@.chann.118	OUTBIAS_B2	0.0	Bias on PID controller output
%MF@.chann.120	INT_BAND_B2	0.0	Integral band
%MF@.chann.122	DBAND_B2	0.0	Dead band on the deviation
%MF@.chann.124	OUTRATE_B2	0.0	Limitation of output variation speed
%MF@.chann.126	OUT1_INF_B2	0.	Output 1 lower limit
%MF@.chann.128	OUT1_SUP_B2	100.0	Output 1 upper limit
%MF@.chann.130	SP_INF_B2	0.0	Setpoint lower limit
%MF@.chann.132	SP_SUP_B2	100.	Setpoint upper limit
%MF@.chann.134	PV_LL_B2	5.	PV very low threshold
%MF@.chann.136	PV_L_B2	5.	PV low threshold
%MF@.chann.138	PV_H_B2	95.	PV high threshold
%MF@.chann.140	PV_HH_B2	95.	PV very high threshold
%MF@.chann.142	ONOFF_L_B2	-5	Low threshold of ONOFF controller
%MF@.chann.144	ONOFF_H_B2	5	High threshold of ONOFF controller
%MF@.chann.146	HYST1_B2	0.0	Hysteresis of 3-state ONOFF controller
%MF@.chann.148	DEV_L_B2	0.	Low threshold of deviation
%MF@.chann.150	DEV_H_B2	0.	High threshold of deviation

Address	Parameter name	Default value	Comment
%MF@.chann.152	THLD_B2	1E+8	Totalizing limit
%MF@.chann.154	R_RATE_B2	0.0	Speed limiter increase speed value
%MF@.chann.156	D_RATE_B2	0.0	Speed limiter decrease speed value
%MF@.chann.158	SPEED_LIM_OUT_B2	Not applicable	Speed limiter output value
%MF@.chann.160	INP_MINR1_B2	0.0	Low scale of setpoint R1
%MF@.chann.162	INP_MAXR1_B2	100.0	High scale of setpoint R1
%MF@.chann.164	T_MOTOR1_B2	10.	Opening time for valve controlled by Servomotor
%MF@.chann.166	T_MINI1_B2	0.	Minimum opening time for valve controlled by Servomotor
%MF@.chann.168	KP_PREV_B2	Not applicable	Value of proportional coefficient before autotuning
%MF@.chann.170	TI_PREV_B2	Not applicable	Value of integral coefficient before autotuning
%MF@.chann.172	TD_PREV_B2	Not applicable	Value of derivative coefficient before autotuning
%MF@.chann.174	T_ECH_B3	0.3	Sampling period
%MF@.chann.176	OUT_MAN_B3	Not applicable	Command value
%MF@.chann.178	DEV_B3	Not applicable	PV setpoint deviation
%MF@.chann.180	PV_B3	Not applicable	PV value in Physical scale
%MF@.chann.182	SP_B3	Not applicable	Setpoint value in Physical scale
%MF@.chann.184	PV_INF_B3	0.	Process value lower limit
%MF@.chann.186	PV_SUP_B3	100.	Process value upper limit
%MF@.chann.188	KP_B3	1.0	Proportional coefficient
%MF@.chann.190	TI_B3	0.0	Integral time
%MF@.chann.192	TD_B3	0.0	Derivative time
%MF@.chann.194	OUTBIAS_B3	0.0	Bias on PID controller output

Address	Parameter name	Default value	Comment
%MF@.chann.196	INT_BAND_B3	0.0	Integral time
%MF@.chann.198	DBAND_B3	0.0	Dead band on the deviation
%MF@.chann.200	OUTRATE_B3	0.0	Limitation of output variation speed
%MF@.chann.202	OUT1_INF_B3	0.	Output 1 lower limit
%MF@.chann.204	OUT1_SUP_B3	100.0	Output 1 upper limit
%MF@.chann.206	SP_INF_B3	0.0	Setpoint lower limit
%MF@.chann.208	SP_SUP_B3	100.	Setpoint upper limit
%MF@.chann.210	PV_LL_B3	5.	PV very low threshold
%MF@.chann.212	PV_L_B3	5.	PV low threshold
%MF@.chann.214	PV_H_B3	95.	PV high threshold
%MF@.chann.216	PV_HH_B3	95.	PV very high threshold
%MF@.chann.218	ONOFF_L_B3	-5	Low threshold of ONOFF controller
%MF@.chann.220	ONOFF_H_B3	5	High threshold of ONOFF controller
%MF@.chann.222	HYST1_B3	0.0	Hysteresis of 3-state ONOFF controller
%MF@.chann.224	DEV_L_B3	0.	Low threshold of deviation
%MF@.chann.226	DEV_H_B3	0.	High threshold of deviation
%MF@.chann.228	THLD_B3	1E+8	Totalizing limit
%MF@.chann.230	R_RATE_B3	0.0	Speed limiter increase speed value
%MF@.chann.232	D_RATE_B3	0.0	Speed limiter decrease speed value
%MF@.chann.234	SPEED_LIM_OUT_B3	Not applicable	Speed limiter output value
%MF@.chann.236	INP_MINR1_B3	0.0	Low scale of setpoint R1
%MF@.chann.238	INP_MAXR1_B3	100.0	High scale of setpoint R1
%MF@.chann.240	T_MOTOR1_B3	10.	Opening time for valve controlled by Servomotor

<b>Address</b>	<b>Parameter name</b>	<b>Default value</b>	<b>Comment</b>
%MF@.chann.242	T_MINI1_B3	0.	Minimum opening time for valve controlled by Servomotor
%MF@.chann.244	KP_PREV_B3	Not applicable	Value of proportional coefficient before autotuning
%MF@.chann.246	TI_PREV_B3	Not applicable	Value of integral coefficient before autotuning
%MF@.chann.248	TD_PREV_B3	Not applicable	Value of derivative coefficient before autotuning

## 6.5 Language objects associated with a cascaded loop channel

### 6.5-1 Configuration language objects

Address	Parameter name	Default value	Comment
%KW@.chann.0	CONFIG_0_M	Not applicable	Word containing the various Process value configuration bits
%KW@.chann.0:X0	Filtrage	Absent(0)	Process value branch filtering function
%KW@.chann.0:X1	Générateur de fonction	Absent(0)	Process value branch function generator
%KW@.chann.0:X2	Totalisateur	Not applicable	Process value branch totalizing function
%KW@.chann.0:X3	Racine Carrée	Absent(0)	Process value branch root function
%KW@.chann.0:X4	Alarmes	Present	Process value branch alarm function
%KW@.chann.0:X8	PV_CLIP	Absent(0)	Peak limiting (or not) of process value
%KW@.chann.0:X9	EXTRAPOL	Non (0)	Extrapolation of function generator
%KW@.chann.0:X10	PV_UNI_BIP	Unipolar(0)	Process value type unipolar/bipolar
%KW@.chann.0:X11	PV_EXTERNE	Not selected(0)	Select standard (0) / external (1) process value
%KW@.chann.1	CONFIG_1_M	Not applicable	Word containing the various Setpoint configuration bits
%KW@.chann.1:X0	SP_Simple	Selected(1)	Type of setpoint selected : Simple
%KW@.chann.1:X1	SP_Sélection	Not selected(0)	Type of setpoint selected : Selection
%KW@.chann.1:X2	Speed_Limiteur	Not selected(0)	Setpoint speed limiter
%KW@.chann.1:X3	SP_SPP	Not selected(0)	Type of setpoint selected : Programmer
%KW@.chann.1:X4	RL/L	Remote local (0)	Speed limiter on local setpoint, or in remote/local mode
%KW@.chann.1:X8	Sel_min	Absent(0)	Function selected for Selection setpoint type
%KW@.chann.1:X9	Sel_max	Absent(0)	Function selected for Selection setpoint type

Address	Parameter name	Default value	Comment
%KW@.chann.1:X10	Sel_switch	Present on Selection	Function selected for Selection setpoint type
%KW@.chann.1:X11	R/L_INIT	Local (1)	Initial value of selected setpoint Remote/Local
%KW@.chann.1:X12	R1/R2_INIT	R1 (0)	Initial value of selected setpoint
%KW@.chann.1:X13	SP_Ratio	Not selected(0)	Type of setpoint selected : Ratio
%KW@.chann.1:X14	SP_Limiteur	Not present	Setpoint limiter (eg. Param_SP)
%KW@.chann.1:X15	SP_Folw	Non-tracking setpoint	Tracking setpoint (0)
%KW@.chann.2	CONFIG_2_M	Not applicable	Word containing the various loop controller and FF configuration bits
%KW@.chann.2:X0	PID	Present (always)	PID function of loop controller branch
%KW@.chann.2:X1	ONOFF 2	Not applicable	Controller 2-state ONOFF branch
%KW@.chann.2:X2	ONOFF 3	Not applicable	Controller 3-state ONOFF branch
%KW@.chann.2:X3	SPLRG/ChFroid	Not applicable	OR presence bits Hot/Cool and Split/Range
%KW@.chann.2:X4	Split/Range	Not applicable	Split/Range function of loop controller branch
%KW@.chann.2:X5	Chaud/Froid	Not applicable	Hot/Cool function of loop controller branch
%KW@.chann.2:X6	Alarmes_DEV	Present	Alarm function on loop controller branch deviation
%KW@.chann.2:X7	Feed Forward	Absent(0)	Presence of a Feed Forward input
%KW@.chann.2:X8	BUMP	With bumps (1)	Management of bumps on change of operating mode
%KW@.chann.2:X9	PV_DEV	On PV (0)	Type of derivative action
%KW@.chann.2:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.2:X11	REV_DIR	Inverse PID action(0)	Type of loop controller action
%KW@.chann.2:X12	MANU/AUTO_INIT	Auto (1)	Initial value of loop controller operating mode

Address	Parameter name	Default value	Comment
%KW@.chann.2:X13	Lead Lag	Absent(0)	Feed Forward branch Lead Lag function
%KW@.chann.2:X14	FF_UNI_BIP	unipolar	Type of Feed Forward PV (unipolar/bipolar)
%KW@.chann.2:X15	IMC	Absent(0)	Loop controller IMC function
%KW@.chann.3	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.7	Unité de la boucle		Loop unit
%KW@.chann.10	CONFIG_0_E	Not applicable	Word containing the various process value configuration bits
%KW@.chann.10:X0	Filtrage	Absent(0)	Process value branch filtering function
%KW@.chann.10:X1	Générateur de fonction	Not applicable	Process value branch function generator
%KW@.chann.10:X2	Totalisateur	Absent(0)	Process value branch totalizing function
%KW@.chann.10:X3	Racine Carrée	Absent(0)	Process value branch root function
%KW@.chann.10:X4	Alarmes	Present	Process value branch alarm function
%KW@.chann.10:X8	PV_CLIP	Absent(0)	Peak limiting (or not) of process value
%KW@.chann.10:X9	EXTRAPOL	Not applicable	Extrapolation of function generator
%KW@.chann.10:X10	PV_UNI_BIP	Unipolar(0)	Process value type unipolar/bipolar
%KW@.chann.10:X13	Totalisateur: Unité mesure	1	(X13=0, X14 =0): phys/ms (X13=1, X14 =0): phys/s
%KW@.chann.10:X14	Totalisateur: Unité mesure	0	(X13=0, X14 =1): phys/mn (X13=1, X14 =1): phys/h
%KW@.chann.11	CONFIG_1_E	Not applicable	Word containing the various setpoint value configuration bits
%KW@.chann.11:X0	SP_Simple	Selected(1)	Type of setpoint selected : Simple
%KW@.chann.11:X1	SP_Selection	Not applicable	Type of setpoint selected : Selection
%KW@.chann.11:X2	Speed_Limiteur	Not selected(0)	Setpoint speed limiter

Address	Parameter name	Default value	Comment
%KW@.chann.11:X3	SP_SPP	Not applicable	Type of setpoint selected : Programmer
%KW@.chann.11:X4	RL/L	Remote local (0)	Speed limiter on local setpoint, or in remote/local mode
%KW@.chann.11:X8	Sel_min	Not applicable	Function selected for Selection type setpoint
%KW@.chann.11:X9	Sel_max	Not applicable	Function selected for Selection type setpoint
%KW@.chann.11:X10	Sel_switch	Not applicable	Function selected for Selection type setpoint
%KW@.chann.11:X11	R/L_INIT	Local (1)	Initial value of selected setpoint : Remote/Local
%KW@.chann.11:X12	R1/R2_INIT	Not applicable	Initial value of selected setpoint state
%KW@.chann.11:X13	SP_Ratio	Not applicable	Type of setpoint selected : Ratio
%KW@.chann.11:X14	SP_Limiteur	Not present	Setpoint limiter (eg. Param_SP)
%KW@.chann.11:X15	SP_Folw	Non-tracking setpoint (0)	Tracking setpoint
%KW@.chann.12	CONFIG_2_E	Not applicable	Word containing the various loop controller and FF configuration bits
%KW@.chann.12:X0	PID	Present (always)	PID function of loop controller branch
%KW@.chann.12:X1	ONOFF 2	Not applicable	Controller 2-state ONOFF branch
%KW@.chann.12:X2	ONOFF 3	Not applicable	Controller 3-state ONOFF branch
%KW@.chann.12:X3	SPLRG/ChFroid	Not applicable	OR presence bits Hot/Cool and Split/Range
%KW@.chann.12:X4	Split/Range	Absent(0)	Split/Range function of loop controller branch
%KW@.chann.12:X5	Chaud/Froid	Not selected	Hot/Cool function of loop controller branch
%KW@.chann.12:X6	Alarms_DEV	Present	Alarm function on loop controller deviation
%KW@.chann.12:X7	Feed Forward	Not applicable	Presence of a Feed Forward input



Address	Parameter name	Default value	Comment
%KW@.chann.12:X8	BUMP	With bumps (1)	Management of bumps on change of operating mode
%KW@.chann.12:X9	PV_DEV	On PV (0)	Type of derivative action
%KW@.chann.12:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.12:X11	REV_DIR	Inverse PID action	Type of loop controller action
%KW@.chann.12:X12	MANU/AUTO_INIT	Manu (0)	Initial value of loop controller operating mode
%KW@.chann.12:X13	Lead Lag	Not applicable	Feed Forward branch Lead Lag function
%KW@.chann.12:X14	FF_UNI_BIP	Not applicable	Feed Forward PV type unipolar/bipolar
%KW@.chann.12:X15	IMC	Absent(0)	Loop controller branch IMC function
%KW@.chann.13	CONFIG_3_E	Not applicable	Word containing the various output configuration bits
%KW@.chann.13:X0	Servo	Not selected	Type of output selected : Servo
%KW@.chann.13:X1	Servo2	Not selected	Type of output selected : Servo
%KW@.chann.13:X2	Analogique1	Not selected	Type of output selected : Analog
%KW@.chann.13:X3	Analogique2	Not selected	Type of output selected : Analog
%KW@.chann.13:X4	PWM1	Not selected	Type of output selected : PWM
%KW@.chann.13:X5	PWM2	Not selected	Type of output selected : PWM
%KW@.chann.13:X8	POT_REV1	Direct (0)	Servo feedback direction
%KW@.chann.13:X9	POT_REV2	Direct (0)	Servo feedback direction
%KW@.chann.13:X10	POT_VAL1_INIT	No (0)	Existence of Servo feedback
%KW@.chann.13:X11	POT_VAL2_INIT	Yes(1)	Existence of Servo feedback (Reserved)
%KW@.chann.13:X12	ANALOG1_UNI_BIP	Unipolar	Analog output type unipolar/bipolar
%KW@.chann.13:X13	ANALOG2_UNI_BIP	Unipolar(0)	Analog output type unipolar/bipolar

---

<b>Address</b>	<b>Parameter name</b>	<b>Default value</b>	<b>Comment</b>
%KW@.chann.14	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.18	Unité de la boucle	Loop unit	

---

## 6.5-2 Default and diagnostic language objects

Address	Parameter name	Default value	Comment
%MW@.chann.0	Echange en cours		
%MW@.chann.1	Compte Rendu		
%MW@.chann.2	STATUS_VOIE1		Channel status defined by FM standard
%MW@.chann.2:X0	STS_DEF_EXTERNE		
%MW@.chann.2:X1	STS_DEF_DEPASS_GAMME		
%MW@.chann.2:X2	STS_DEF_BORNIER		
%MW@.chann.2:X3	STS_DEF_EXT_PROT		
%MW@.chann.2:X4	STS_DEFAULT_INTERNE		Serious internal fault
%MW@.chann.2:X5	STS_DEFAULT_CONFIG		
%MW@.chann.2:X6	STS_DEFAULT_COMMUNIC		
%MW@.chann.2:X7	WARN		Warning sigma
%MW@.chann.2:X8	STS_ERR_CALC_CORR		Master loop controller branch calculation error
%MW@.chann.2:X9	STS_ERR_FLOT_CORR		Master loop controller branch floating point error
%MW@.chann.2:X10	STS_ERR_CALC_PV		PV master branch calculation error
%MW@.chann.2:X11	STS_ERR_FLOT_PV		PV master branch floating point error
%MW@.chann.2:X12	STS_ERR_SCALE_PV		PV master branch incorrect scale
%MW@.chann.3	STATUS_VOIE2		Channel status defined by FM standard
%MW@.chann.3:X0	STS_ERR_CALC_OUT		OUT branch calculation error
%MW@.chann.3:X1	STS_ERR_FLOT_OUT		OUT branch floating point error
%MW@.chann.3:X2	STS_ERR_TH_SPLRG		SPLRG function thresholds incorrect
%MW@.chann.3:X3	STS_ERR_SCALE_OUT1		OUT1 branch incorrect scale
%MW@.chann.3:X4	STS_ERR_SCALE_OUT2		OUT2 branch incorrect scale
%MW@.chann.3:X8	STS_ERR_CALC_CORR		Slave loop controller branch calculation error

Address	Parameter name	Default value	Comment
%MW@.chann.3:X9	STS_ERR_FLOT_CORR		Slave loop controller branch floating point error
%MW@.chann.3:X10	STS_ERR_CALC_PV		PV slave branch calculation error
%MW@.chann.3:X11	STS_ERR_FLOT_PV		PV slave branch floating point error
%MW@.chann.3:X12	STS_ERR_SCALE_PV		PV slave branch incorrect scale
%MW@.chann.4	STATUS1_M		Word containing the various Process value/ Setpoint status bits
%MW@.chann.4:X0			
%MW@.chann.4:X1	STS_FORCAGE_PV		PV simulation status
%MW@.chann.4:X2	STS_PV_H_LIM		Upper limit on process value branch (PV_SUP)
%MW@.chann.4:X3	STS_PV_L_LIM		Lower limit on process value branch (PV_INF)
%MW@.chann.4:X4	STS_SP_H_LIM		Upper limit on setpoint branch
%MW@.chann.4:X5	STS_SP_L_LIM		Lower limit on setpoint branch
%MW@.chann.4:X6	STS_R_L	R/L Init	State of selected setpoint Remote/Local
%MW@.chann.4:X7	STS_R1_R2		State of selected setpoint
%MW@.chann.4:X8	STS_SIGMA_ALA		Sigma of process value alarms
%MW@.chann.4:X9	STS_HH		Very high alarm
%MW@.chann.4:X10	STS_H		High alarm
%MW@.chann.4:X11	STS_L		Low alarm
%MW@.chann.4:X12	STS_LL		Very low alarm
%MW@.chann.4:X13	STS_DEV_H		High threshold of the Process value Setpoint deviation (>0)
%MW@.chann.4:X14	STS_DEV_L		Low threshold of the Process value Setpoint deviation (>0)
%MW@.chann.4:X15			
%MW@.chann.5	STATUS2_M	Not applicable	Word containing the various loop controller status bits



Address	Parameter name	Default value	Comment
%MW@.chann.5:X0	STS_AT_EN_COURS		Autotuning in progress
%MW@.chann.5:X1	STS_TR_S1		PID in tracking mode (cascade open)
%MW@.chann.5:X3	STS_M_A		State of PID operating mode
%MW@.chann.5:X8	STS_LIM_PID_INF		Lower output limit reached
%MW@.chann.5:X9	STS_LIM_PID_SUP		Upper output limit reached
%MW@.chann.5:X10	STS_TOP_NEXT_CYCLE		Sampling pulse in next cycle
%MW@.chann.5:X11	STS_TOP_CURRENT_CYCLE		Sampling pulse in current cycle
%MW@.chann.5:X12	STS_FORCAGE_FF		FF process value simulation status
%MW@.chann.5:X13	STS_OUT_CLAMP_LOW		Master output clamped in decreasing direction
%MW@.chann.5:X14	STS_OUT_CLAMP_HIGH		Master output clamped in increasing direction
%MW@.chann.6	STATUS3_M	Not applicable	Word containing precise diagnostics for the various warnings (FF setpoint process value)
%MW@.chann.6:X0	STS_Xi_WARN		Control warning for Xi parameters
%MW@.chann.6:X1	STS_Yi_WARN		Control warning for Yi parameters
%MW@.chann.6:X2	STS_INP_MINR1_WARN		Control warning for INP_MINR1 and MAXR1 parameters
%MW@.chann.6:X3	STS_INP_MINR2_WARN		Control warning for INP_MINR2 and MAXR2 parameters
%MW@.chann.6:X4	STS_RATIO_WARN		Control warning for RATIO_MIN and MAX parameters
%MW@.chann.6:X5	STS_CALC_FF_WARN		Feed Forward calculation warning
%MW@.chann.6:X6	STS_FLOAT_FF_WARN		Feed Forward floating point warning
%MW@.chann.6:X7	STS_OUT_FF__WARN	Canceled (reserved)	Control warning for OUTFF_INF and SUP parameters

Address	Parameter name	Default value	Comment
%MW@.chann.6:X8			
%MW@.chann.6:X9	STS_SP_INF_WARN		Control warning for SP_INF and SUP parameters
%MW@.chann.6:X10	STS_CALC_SP_WARN		Setpoint calculation warning
%MW@.chann.6:X11	STS_FLOAT_SP_WARN		Setpoint floating point warning
%MW@.chann.7	STATUS1_E		Word containing the various Process value/ Setpoint status bits
%MW@.chann.7:X0	STS_HOLD_TOT		State of the totalizing function
%MW@.chann.7:X1	STS_FORCAGE_PV		Process value simulation status
%MW@.chann.7:X2	STS_PV_H_LIM		Upper limit on process value branch (PV_SUP)
%MW@.chann.7:X3	STS_PV_L_LIM		Lower limit on process value branch (PV_INF)
%MW@.chann.7:X4	STS_SP_H_LIM		Upper limit on setpoint branch (SP_SUP)
%MW@.chann.7:X5	STS_SP_L_LIM		Lower limit on setpoint branch (SP_INF)
%MW@.chann.7:X6	STS_R_L	R/L Init	State of setpoint selected Remote/Local
%MW@.chann.7:X8	STS_SIGMA_ALA		Sigma of process value alarms
%MW@.chann.7:X9	STS_HH		Very high alarm
%MW@.chann.7:X10	STS_H		High alarm
%MW@.chann.7:X11	STS_L		Low alarm
%MW@.chann.7:X12	STS_LL		Very low alarm
%MW@.chann.7:X13	STS_DEV_H		High threshold of the Process value Setpoint deviation (>0)
%MW@.chann.7:X14	STS_DEV_L		Low threshold of the Process value Setpoint deviation (<0)
%MW@.chann.7:X15	STS_THLD_DONE		Totalizer threshold reached
%MW@.chann.8	STATUS2_E	Not applicable	Word containing the various controller status bits
%MW@.chann.8:X0	STS_AT_EN_COURS		Autotuning in progress
%MW@.chann.8:X1	STS_TR_S1		Tracking Switch

Address	Parameter name	Default value	Comment
%MW@.chann.8:X2	STS_TR_S2	Not applicable	Tracking Switch (Reserved)
%MW@.chann.8:X3	STS_M_A		State of PID operating mode
%MW@.chann.8:X4	STS_RAISE1		Opening command
%MW@.chann.8:X5	STS_LOWER1		Closing command
%MW@.chann.8:X6	STS_RAISE2		Opening command for Output 2 branch
%MW@.chann.8:X7	STS_LOWER2		Closing command for Output 2 branch
%MW@.chann.8:X8	STS_LIM_PID_INF		The calculated PID output is greater than or equal to OUT_SUP
%MW@.chann.8:X9	STS_LIM_PID_SUP		The calculated PID output is less than or equal to OUT_INF
%MW@.chann.8:X10	STS_TOP_NEXT_CYCLE		
%MW@.chann.8:X11	STS_TOP_CURRENT_CYCLE		
%MW@.chann.9	STATUS3_E	Not applicable	Word containing output diagnostics
%MW@.chann.9:X0	STS_POT_VAL1		Servo operation with feedback
%MW@.chann.9:X1	STS_POT_VAL2	Not applicable	Servo operation with feedback (Reserved)
%MW@.chann.9:X2	STS_RAISESTOP1		Opening stop reached on Servomotor
%MW@.chann.9:X3	STS_LOWERSTOP1		Closing stop reached on Servomotor
%MW@.chann.9:X4	STS_RAISESTOP2		Opening stop reached on Servomotor
%MW@.chann.9:X5	STS_LOWERSTOP2		Closing stop reached on Servomotor
%MW@.chann.9:X8	STS_OVER_TOT_WARN		Totalizer output capacity overflow warning
%MW@.chann.9:X9	STS_SP_INF_WARN		Control warning for SP_INF and SUP parameters
%MW@.chann.9:X10	STS_CALC_SP_WARN		Setpoint calculation warning
%MW@.chann.9:X11	STS_FLOAT_SP_WARN		Setpoint floating point warning

Address	Parameter name	Default value	Comment
%MW@.chann.10	STATUS4	Not applicable	Word containing the autotuning diagnostics
%MW@.chann.10:X0	STS_AT_FAILED		Autotuning failed
%MW@.chann.10:X1	STS_AT_ABORTED		Autotuning diagnostics interrupted
%MW@.chann.10:X2	STS_AT_ERR_PARAM		Autotuning diagnostics parameter error
%MW@.chann.10:X3	STS_AT_ERR_PWF_OR_EFB_FAILURE		Autotuning diagnostics system error or power outage
%MW@.chann.10:X4	STS_AT_ERR_SATUR		Autotuning diagnostics saturation of the process value
%MW@.chann.10:X5	STS_AT_ERR_DV_TOO_SMALL		Autotuning diagnostics process value deviation too small
%MW@.chann.10:X6	STS_AT_ERR_TSAMP_HIGH		Autotuning diagnostics Sampling period too long
%MW@.chann.10:X7	STS_AT_ERR_INCONSISTENT_RESPONSE		Autotuning diagnostics inconsistent response
%MW@.chann.10:X8	STS_AT_ERR_NOT_STAB_INIT		Autotuning diagnostics process value initially unstable
%MW@.chann.10:X9	STS_AT_ERR_TMAX_TOO_SMALL		Autotuning diagnostics step function duration too short
%MW@.chann.10:X10	STS_AT_ERR_NOISE_TOO_HIGH		Autotuning diagnostics PV noise too high
%MW@.chann.10:X11	STS_AT_ERR_TMAX_TOO_HIGH		Autotuning diagnostics step function duration too long
%MW@.chann.10:X12	STS_AT_WARN_OVERSHOOT		Autotuning diagnostics overshoot greater than 10%
%MW@.chann.10:X13	STS_AT_WARN_UNDERSHOOT		Autotuning diagnostics undershoot too great
%MW@.chann.10:X14	STS_AT_WARN_UNSYMETRICAL_PLANT		Autotuning diagnostics non-symmetrical process
%MW@.chann.10:X15	STS_AT_WARN_INTEGRATING_PLANT		Autotuning diagnostics integrating process
%MW@.chann.11	Ordre Commande		
%MD@.chann.12	Paramètre Commande		



### 6.5-3 Process control language objects

Address	Parameter name	Default value	Comment
%MF@.chann.14	AT_STEP	10	Amplitude of autotuning step function
%MF@.chann.16	AT_TMAX	100	Duration of autotuning step function
%MF@.chann.18	AT_PERF	0.5	Autotuning stability criterion
%MF@.chann.20	T_ECH_M	0.3	Sampling period
%MF@.chann.22	OUTFF_M	Not applicable	Value of Feed Forward action in Physical scale
%MF@.chann.24	OUT_MAN_M	Not applicable	Command value
%MF@.chann.26	DEV_M	Not applicable	Process value setpoint deviation
%MF@.chann.28	PV_M	Not applicable	Process value in Physical scale
%MF@.chann.30	SP_M	Not applicable	Setpoint value in Physical scale
%MF@.chann.32	PV_INF_M	0.	Process value lower limit
%MF@.chann.34	PV_SUP_M	100.	Process value upper limit
%MF@.chann.36	KP_M	1.0	Proportional coefficient
%MF@.chann.38	TI_M	0.0	Integral time
%MF@.chann.40	TD_M	0.0	Derivative time
%MF@.chann.42	OUTBIAS_M	0.0	Bias on the PID controller output
%MF@.chann.44	INT_BAND_M	0.0	Integral band
%MF@.chann.46	DBAND_M	0.0	Dead band on the deviation
%MF@.chann.48	KD_M	10.0	Derivative filtering
%MF@.chann.50	SP_INF_M	0.0	Setpoint lower limit
%MF@.chann.52	SP_SUP_M	100.	Setpoint upper limit
%MF@.chann.54	PV_LL_M	5.	PV very low threshold
%MF@.chann.56	PV_L_M	5.	PV low threshold
%MF@.chann.58	PV_H_M	95.	PV high threshold
%MF@.chann.60	PV_HH_M	95.	PV very high threshold

Address	Parameter name	Default value	Comment
%MF@.chann.62	RATIO_M	1.0	Ratio value
%MF@.chann.64	RATIO_MIN_M	0.	Minimum Ratio value
%MF@.chann.66	RATIO_MAX_M	100	Maximum Ratio value
%MF@.chann.68	RATIO_BIAS_M	0	Ratio bias value
%MF@.chann.70	DEV_L_M	0.	Low deviation threshold
%MF@.chann.72	DEV_H_M	0.	High deviation threshold
%MF@.chann.74	T_FILT_M	0.0	Process value filtering time
%MF@.chann.76	K_FILT_M	1.0	Multiplication coefficient for process value filtering
%MF@.chann.78	FILT_OUT_M		Filter output value
%MF@.chann.80	SQRT_OUT_M		Square root output value
%MF@.chann.82	E2_IN_M	1428	Abscissa of first point of Segment S2
%MF@.chann.84	E3_IN_M	2857	Abscissa of first point of Segment S3
%MF@.chann.86	E4_IN_M	4285	Abscissa of first point of Segment S4
%MF@.chann.88	E5_IN_M	5714	Abscissa of first point of Segment S5
%MF@.chann.90	E6_IN_M	7143	Abscissa of first point of Segment S6
%MF@.chann.92	E7_IN_M	8571	Abscissa of first point of Segment S7
%MF@.chann.94	E2_OUT_M	14.28	Ordinate of first point of Segment S2
%MF@.chann.96	E3_OUT_M	28.57	Ordinate of first point of Segment S3
%MF@.chann.98	E4_OUT_M	42.85	Ordinate of first point of Segment S4
%MF@.chann.100	E5_OUT_M	57.14	Ordinate of first point of Segment S5
%MF@.chann.102	E6_OUT_M	71.43	Ordinate of first point of Segment S6
%MF@.chann.104	E7_OUT_M	8571	Ordinate of first point of Segment S7
%MF@.chann.106	R_RATE_M	0.0	Setpoint increase speed limit

Address	Parameter name	Default value	Comment
%MF@.chann.108	D_RATE_M	0.0	Setpoint decrease speed limit
%MF@.chann.110	SPEED_LIM_OUT_M		Setpoint speed limiter output value
%MF@.chann.112	INP_MINR1_M	0.0	Low scale of setpoint R1
%MF@.chann.114	INP_MAXR1_M	100.0	High scale of setpoint R1
%MF@.chann.116	INP_MINR2_M	0.0	Low scale of setpoint R2
%MF@.chann.118	INP_MAXR2_M	100.0	High scale of setpoint R2
%MF@.chann.120	T1_FF_M	0.0	Feed Forward process value filtering time
%MF@.chann.122	T2_FF_M	0.0	Feed Forward process value filtering time
%MF@.chann.124	OUT_FF_INF_M	0.	Feed Forward action lower limit
%MF@.chann.126	OUT_FF_SUP_M	100.	Feed Forward action upper limit
%MF@.chann.128	KP_PREV_M	Not applicable	Value before autotuning of proportional coefficient
%MF@.chann.130	TI_PREV_M	Not applicable	Value before autotuning of integral coefficient
%MF@.chann.132	TD_PREV_M	Not applicable	Value before autotuning of derivative coefficient
%MF@.chann.134	OUT1_E	Not applicable	Command value output 1
%MF@.chann.136	OUT2_E	Not applicable	Command value output 2
%MF@.chann.138	T_ECH_E	0.3	Sampling period
%MF@.chann.140	OUT_MAN_E	Not applicable	Command value
%MF@.chann.142	DEV_E	Not applicable	Process value setpoint deviation
%MF@.chann.144	PV_E	Not applicable	Process value in Physical scale
%MF@.chann.146	SP_E	Not applicable	Setpoint value in Physical scale
%MF@.chann.148	PV_INF_E	0.	Process value lower limit
%MF@.chann.150	PV_SUP_E	100.	Process value upper limit

Address	Parameter name	Default value	Comment
%MF@.chann.152	KP_E	1.0	Proportional coefficient
%MF@.chann.154	TI_E	0.0	Integral time
%MF@.chann.156	TD_E	0.0	Derivative time
%MF@.chann.158	OUTBIAS_E	0.0	Bias on the output of the PID controller
%MF@.chann.160	INT_BAND_E	0.0	Integral band
%MF@.chann.162	DBAND_E	0.0	Dead band on the deviation
%MF@.chann.164	KD_E	10.0	Filtering of derivative
%MF@.chann.166	OUTRATE_E	0.0	Output variation speed limit
%MF@.chann.168	OUTRATE2_E	0,0	Variation speed limit of output 2
%MF@.chann.170	OUT1_INF_E	0.	Lower limit of output 1
%MF@.chann.172	OUT1_SUP_E	100.0	Upper limit of output 1
%MF@.chann.174	SP_INF_E	0.0	Lower limit of setpoint
%MF@.chann.176	SP_SUP_E	100.	Upper limit of setpoint
%MF@.chann.178	OUT2_INF_E	0.	Lower limit of output 2
%MF@.chann.180	OUT2_SUP_E	100.	Upper limit of output 2
%MF@.chann.182	OUT1_TH1_E	0.	Threshold 1 of output 1 of Hot/Cold or Split/Range
%MF@.chann.184	OUT1_TH2_E	50.0	Threshold 2 of output 1 of Hot/Cold or Split/Range
%MF@.chann.186	OUT2_TH1_E	50.0	Threshold 1 of output 2 of Hot/Cold or Split/Range
%MF@.chann.188	OUT2_TH2_E	100	Threshold 2 of output 2 of Hot/Cold or Split/Range
%MF@.chann.190	PV_LL_E	5.	PV very low threshold
%MF@.chann.192	PV_L_E	5.	PV low threshold
%MF@.chann.194	PV_H_E	95.	PV high threshold
%MF@.chann.196	PV_HH_E	95.	PV very high threshold
%MF@.chann.198	DEV_L_E	0.	Low deviation threshold

Address	Parameter name	Default value	Comment
%MF@.chann.200	DEV_H_E	0.	High deviation threshold
%MF@.chann.202	T_E	0.0	Process value filtering time
%MF@.chann.204	K_FILT_E	1.0	Multiplication coefficient for process value filtering
%MF@.chann.206	FILT_OUT_E	Not applicable	Filter output value
%MF@.chann.208	SQRT_OUT_E	Not applicable	Square root output value
%MF@.chann.210	THLD_E	1E+8	Totalizing limit
%MF@.chann.212	R_RATE_E	0.0	Setpoint increase speed limit
%MF@.chann.214	D_RATE_E	0.0	Setpoint decrease speed limit
%MF@.chann.216	SPEED_LIM_OUT_E	Not applicable	Setpoint speed limiter output value
%MF@.chann.218	T_MOTOR1_E	10.	Opening time of the valve controlled by Servomotor
%MF@.chann.220	T_MINI1_E	0.	Minimum opening time of the valve controlled by Servomotor
%MF@.chann.222	T_MOTOR2_E	10.	Opening time of the valve controlled by Servomotor
%MF@.chann.224	T_MINI2_E	0.	Minimum opening time of the valve controlled by Servomotor
%MF@.chann.226	KP_PREV_E	Not applicable	Value before autotuning of the proportional coefficient
%MF@.chann.228	TI_PREV_E	Not applicable	Value before autotuning of the integral coefficient
%MF@.chann.230	TD_PREV_E	Not applicable	Value before autotuning of the derivative coefficient
%MF@.chann.232	KS	1.0	IMC static gain
%MF@.chann.234	T1	1.0	Time constant in OL
%MF@.chann.236	T_DELAY	0.0	Current pure delay
%MF@.chann.238	CC_PERF	0.1	OL / CL time ratio
%MF@.chann.240		Not applicable	
%MF@.chann.242		Not applicable	

Address	Parameter name	Default value	Comment
%MF@.chann.244		Not applicable	
%MF@.chann.246	reserv1_IMC_C8	Not applicable	
%MW@.chann.248	PV_SIM_M	Not applicable	Simulated process value
%MW@.chann.249	PV_SIM_E	Not applicable	Simulated process value
%MW@.chann.250	FF_SIM_M	Not applicable	Simulated feed forward input

## 6.6 Language objects associated with the autoselective loop channel

### 6.6-1 Configuration language objects

Address	Parameter name	Default value	Comment
%KW@.chann.0	CONFIG_0_C1	Not applicable	Word containing the various configuration bits for the C1 process value
%KW@.chann.0:X0	Filtrage	Absent(0)	Process value branch filtering function
%KW@.chann.0:X1	Générateur de fonction	Absent(0)	Process value branch function generator
%KW@.chann.0:X2	Totalizer	Absent(0)	Process value branch totalizer function
%KW@.chann.0:X3	Racine Carrée	Absent(0)	Process value branch square root function
%KW@.chann.0:X4	Alarmes	Present	Process value branch alarm function
%KW@.chann.0:X8	PV_CLIP	Absent(0)	Peak limiting (or not) of process value
%KW@.chann.0:X9	EXTRAPOL	No (0)	Extrapolation of function generator
%KW@.chann.0:X10	PV_UNI_BIP	Unipolar(0)	Type (uni/bipolar) of process value
%KW@.chann.0:X11	PV_EXTERNE	Absent (0)	Selection of Standard (0) / External process value (1)
%KW@.chann.0:X13	Totalisateur : Unité mesure	1	(X13=0, X14 =0): phys/ms (X13=1, X14 =0): phys/s
%KW@.chann.0:X14	Totalisateur : Unité mesure	0	(X13=0, X14 =1): phys/mn (X13=1, X14 =1): phys/h
%KW@.chann.1	CONFIG_1_C1	Not applicable	Word containing the various configuration bits for the C1 setpoint
%KW@.chann.1:X0	SP_Simple	Selected(1)	Type of setpoint selected : Simple
%KW@.chann.1:X1	SP_Sélection	Not selected(0)	Type of setpoint selected : Selection
%KW@.chann.1:X2	Speed_Limiteur	Not selected(0)	Setpoint speed limiter
%KW@.chann.1:X3	SP_SPP	Not selected(0)	Type of setpoint selected : Programmer
%KW@.chann.1:X4	RL/L	Remote Local (0)	Speed limiter either on local setpoint or in remote/local mode

Address	Parameter name	Default value	Comment
%KW@.chann.1:X8	Sel_min	Absent(0)	Function selected for Selection type of setpoint
%KW@.chann.1:X9	Sel_max	Absent(0)	Function selected for Selection type of setpoint
%KW@.chann.1:X10	Sel_switch	Present on Selection	Function selected for Selection type of setpoint
%KW@.chann.1:X11	R/L_INIT	Locale (1)	Initial value of the setpoint selected Remote/ Local
%KW@.chann.1:X12	R1/R2_INIT	R1 (0)	Initial value of the state of the setpoint selected
%KW@.chann.1:X13	SP_Ratio	Not selected(0)	Type of setpoint selected : Ratio
%KW@.chann.1:X14	SP_Limiteur	Not present	Setpoint limiter (eg Param_SP)
%KW@.chann.1:X15	SP_Folw	Non-tracking setpoint (0)	Tracking setpoint
%KW@.chann.2	CONFIG_2_C1	Not applicable	Word containing the various configuration bits of the loop controller and FF of C1
%KW@.chann.2:X0	PID	Always present	Loop controller branch PID function
%KW@.chann.2:X1	ONOFF 2	Not applicable	Controller 2-state ONOFF branch
%KW@.chann.2:X2	ONOFF 3	Not applicable	Controller 3-state ONOFF branch
%KW@.chann.2:X3	SPLRG/ChFroid	Not applicable	OR of presence bits for Hot/Cold and Split/Range
%KW@.chann.2:X4	Split/Range	Absent(0)	Split/Range function of the selected branch
%KW@.chann.2:X5	Chaud/Froid	Not selected the selected branch	Hot/Cold function of the selected branch
%KW@.chann.2:X6	Alarmes_DEV	Present	Alarm function on deviation of the loop controller branch
%KW@.chann.2:X7	Feed Forward	Absent(0)	Presence of a Feed Forward input
%KW@.chann.2:X8	BUMP	with bumps (1)	Management of bumps on change of operating mode
%KW@.chann.2:X9	PV_DEV	On PV (0)	Type of derivative action





Address	Parameter name	Default value	Comment
%KW@.chann.2:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.2:X11	REV_DIR	Inverse PID action (0)	Type of controller action
%KW@.chann.2:X12	MANU/AUTO_INIT	Auto (1)	Initial value of loop controller operating mode
%KW@.chann.2:X13	Lead Lag	Absent(0)	Lead Lag function of the Feed Forward branch
%KW@.chann.2:X14	FF_UNI_BIP	Unipolar	Type (uni/bipolar) of Feed Forward process value
%KW@.chann.2:X15	IMC	Absent (0)	Model-based controller in the loop controller branch
%KW@.chann.3	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.7	Unité de la boucle		Loop unit
%KW@.chann.10	CONFIG_0_C2	Not applicable	Word containing the various configuration bits of the C2 process value
%KW@.chann.10:X0	Filtrage	Not applicable	Process value branch filtering function
%KW@.chann.10:X1	Générateur de fonction	Not applicable	Process value branch function generator
%KW@.chann.10:X2	Totalisateur	Absent (0)	Process value branch totalizer function
%KW@.chann.10:X3	Racine Carrée	Absent(0)	Process value branch square root function
%KW@.chann.10:X4	Alarmes	Present	Process value branch alarm function
%KW@.chann.10:X8	PV_CLIP	Absent(0)	Peak limiting (or not) of process value
%KW@.chann.10:X9	EXTRAPOL	Not applicable	Extrapolation of function generator
%KW@.chann.10:X10	PV_UNI_BIP	Unipolar(0)	Type (uni/bipolar) of PV
%KW@.chann.10:X11	PV_EXTERNE	Absent (0)	Selection of Standard process value (0) / External process value (1)
%KW@.chann.10:X13	Totalisateur: Unité mesure	1	(X13=0, X14 =0): phys/ms (X13=1, X14 =0): phys/s
%KW@.chann.10:X14	Totalisateur: Unité mesure	0	(X13=0, X14 =1): phys/mn (X13=1, X14 =1): phys/h
%KW@.chann.11	CONFIG_1_C2	Not applicable	Word containing the various configuration bits of the C2 setpoint

Address	Parameter name	Default value	Comment
%KW@.chann.11:X0	SP_Simple	Selected(1)	Type of setpoint selected : Simple
%KW@.chann.11:X1	SP_Sélection	Not applicable	Type of setpoint selected : Selection
%KW@.chann.11:X2	Speed_Limiteur	Absent (0)	Setpoint speed limiter
%KW@.chann.11:X3	SP_SPP	Not applicable	Type of setpoint selected : Programmer
%KW@.chann.11:X4	RL/L	Remote Local (0)	Speed limiter either on local setpoint or in remote/local mode
%KW@.chann.11:X8	Sel_min	Not applicable	Function selected for Selection type of setpoint
%KW@.chann.11:X9	Sel_max	Not applicable	Function selected for Selection type of setpoint
%KW@.chann.11:X10	Sel_switch	Not applicable	Function selected for Selection type of setpoint
%KW@.chann.11:X11	R/L_INIT	Local (1)	Initial value of the setpoint selected Remote/Local
%KW@.chann.11:X12	R1/R2_INIT	Not applicable	Initial value of the state of the setpoint selected
%KW@.chann.11:X13	SP_Ratio	Not applicable	Type of setpoint selected : Ratio
%KW@.chann.11:X14	SP_Limiteur	Not present (0)	Setpoint limiter (eg Param_SP)
%KW@.chann.11:X15	SP_Folw	Non-tracking	Tracking setpoint setpoint (0)
%KW@.chann.12	CONFIG_2_C2	Not applicable	Word containing the various configuration bits of the loop controller and FF of C2
%KW@.chann.12:X0	PID	Present (tjs)	Loop controller branch PID function
%KW@.chann.12:X1	ONOFF 2	Not applicable	Controller 2-state ONOFF branch
%KW@.chann.12:X2	ONOFF 3	Not applicable	Controller 3-state ONOFF branch
%KW@.chann.12:X3	SPLRG/ChFroid	Not applicable	OR of presence bits for Hot/Cold and Split/Range
%KW@.chann.12:X4	Split/Range	Not applicable	Loop controller branch Split/Range function

Address	Parameter name	Default value	Comment
%KW@.chann.12:X5	Chaud/Froid	Not applicable	Hot/Cold function of the loop controller branch
%KW@.chann.12:X6	Alarms_DEV	Present	Alarm function on deviation of the loop controller branch
%KW@.chann.12:X7	Feed Forward	Not applicable	Presence of a Feed Forward input
%KW@.chann.12:X8	BUMP	with bumps (1)	Management of bumps on change of operating mode
%KW@.chann.12:X9	PV_DEV	On PV (0)	Type of derivative action
%KW@.chann.12:X10	MIX_PAR	Serial parallel PID	Mixed or parallel loop controller
%KW@.chann.12:X11	REV_DIR	Inverse PID action (0)	Type of controller action
%KW@.chann.12:X12	MANU/AUTO_INIT	Auto (1)	Initial value of loop controller operating mode
%KW@.chann.12:X13	Lead Lag	Not applicable	Lead Lag function of the Feed Forward branch
%KW@.chann.12:X14	FF_UNI_BIP	Not applicable	Type (uni/bipolar) of Feed Forward process value
%KW@.chann.12:X15	IMC	Absent (0)	Model-based controller in the loop controller branch
%KW@.chann.13	CONFIG_3_C2	Not applicable	Word containing the various output configuration bits
%KW@.chann.13:X0	Servo	Not selected	Type of output selected : Servo
%KW@.chann.13:X1	Servo2	Not selected	Type of output selected : Servo
%KW@.chann.13:X2	Analogique1	Not selected	Type of output selected : Analog
%KW@.chann.13:X3	Analogique2	Not selected	Type of output selected : Analog
%KW@.chann.13:X4	PWM1	Not selected	Type of output selected : PWM
%KW@.chann.13:X5	PWM2	Not selected	Type of output selected : PWM
%KW@.chann.13:X8	POT_REV1	Direct (0)	Servo feedback direction
%KW@.chann.13:X9	POT_REV2	Direct (0)	Servo feedback direction
%KW@.chann.13:X10	POT_VAL1_INIT	No (0)	Existence of Servo feedback

Address	Parameter name	Default value	Comment
%KW@.chann.13:X11	POT_VAL2_INIT	No (0)	Existence of Servo feedback
%KW@.chann.13:X12	ANALOG1_UNI_BIP	Unipolar (0)	Type (uni/bipolar) of analog output
%KW@.chann.13:X13	ANALOG2_UNI_BIP	Unipolar (0)	Type (uni/bipolar) of analog output
%KW@.chann.14	Nom de la boucle	Loop i where i [0;9]	Loop name
%KW@.chann.18	Unité de la boucle		Loop unit
%KW@.chann.21	CONFIG_0_G	Not applicable	Word containing the various configuration bits of the global loop
%KW@.chann.21:X0	MANU/AUTO_G_INIT	Manu(0)	Initial value of global loop operating mode
%KW@.chann.21:X1	AM_G_PID	On global loop (0)	Initial management of A/M blocks : at 0 A/M block on global loop
%KW@.chann.21:X2			at 1 : A/M blocks on each PID
%KW@.chann.21:X8	MIN_MAX	Min (0)	Initial behavior of autoselector
%KW@.chann.21:X9	AS_INIT	Present (1)	Output obtained by the secondary on init = autoselector output
%KW@.chann.21:X10	DIR1_INIT	Absent (0)	Output obtained by the secondary on init = output of PID no. 1
%KW@.chann.21:X11	DIR2_INIT	Absent (0)	Output obtained by the secondary on init = output of PID no. 2

**6.6-2 Fault and diagnostics language objects**

Address	Parameter name	Default value	Comment
%MW@.chann.0	Echange en cours		
%MW@.chann.1	Compte Rendu		
%MW@.chann.2	STATUS_VOIE1		Channel status defined by FM standard
%MW@.chann.2:X0	STS_DEF_EXTERNE		
%MW@.chann.2:X1	STS_DEF_DEPASS_GAMME		
%MW@.chann.2:X2	STS_DEF_BORNIER		
%MW@.chann.2:X3	STS_DEF_EXT_PROT		
%MW@.chann.2:X4	STS_DEFAULT_INTERNE		Serious internal fault
%MW@.chann.2:X5	STS_DEFAULT_CONFIG		
%MW@.chann.2:X6	STS_DEFAULT_COMMUNIC		
%MW@.chann.2:X7	WARN		Warning sigma
%MW@.chann.2:X8	STS_ERR_CALC_CORR_B2		Loop controller branch calculation error
%MW@.chann.2:X9	STS_ERR_FLOT_CORR_B2		Loop controller branch floating point error
%MW@.chann.2:X10	STS_ERR_CALC_PV_B2		PV branch calculation error
%MW@.chann.2:X11	STS_ERR_FLOT_PV_B2		PV branch floating point error
%MW@.chann.2:X12	STS_ERR_SCALE_PV_B2		PV1 branch incorrect scale
%MW@.chann.3	STATUS_VOIE2		Channel status defined by FM standard
%MW@.chann.3:X0	STS_ERR_CALC_OUT		OUT branch calculation error
%MW@.chann.3:X1	STS_ERR_FLOT_OUT		OUT branch floating point error
%MW@.chann.3:X2	STS_ERR_TH_SPLRG		SPLRG function thresholds incorrect
%MW@.chann.3:X3	STS_ERR_CALC_CONTRAINTE		Secondary branch calculation error
%MW@.chann.3:X4			(between selection and output branch)
%MW@.chann.3:X8	STS_ERR_CALC_CORR_B1		Loop controller branch calculation error
%MW@.chann.3:X9	STS_ERR_FLOT_CORR_B1		Loop controller branch floating point error

Address	Parameter name	Default value	Comment
%MW@.chann.3:X10	STS_ERR_CALC_PV_B1		PV branch calculation error
%MW@.chann.3:X11	STS_ERR_FLOT_PV_B1		PV branch floating point error
%MW@.chann.3:X12	STS_ERR_SCALE_PV_B1		PV branch incorrect scale
%MW@.chann.3:X13	STS_ERR_SCALE_OUT1		C1 branch incorrect scale
%MW@.chann.3:X14	STS_ERR_SCALE_OUT2		C2 branch incorrect scale
%MW@.chann.3:X15	STS_ERR_SCALE		OR of scale errors
%MW@.chann.4	STATUS0_C1		Word containing the various Process value/ Setpoint status bits of loop 1
%MW@.chann.4:X0	STS_HOLD_TOT_B1		Freezes totalizer function
%MW@.chann.4:X1	STS_PV_Sim_B1		Simulated PV
%MW@.chann.4:X2	STS_PV_H_LIM_B1		Upper limit on PV
%MW@.chann.4:X3	STS_PV_L_LIM_B1		Lower limit on PV
%MW@.chann.4:X4	STS_SP_H_LIM_B1		Upper limit on setpoint
%MW@.chann.4:X5	STS_SP_L_LIM_B1		Lower limit on setpoint
%MW@.chann.4:X6	STS_L_R_B1	R/L Init	Remote Setpoint (1) Local Setpoint (0)
%MW@.chann.4:X7	STS_R1_R2		Remote Setpoint2 (1) Remote Setpoint1 (0)
%MW@.chann.4:X8	STS_ALARMS_B1		Logic OR of PV alarms
%MW@.chann.4:X9	STS_HH_B1		Very high alarm
%MW@.chann.4:X10	STS_H_B1		High alarm
%MW@.chann.4:X11	STS_L_B1		Low alarm
%MW@.chann.4:X12	STS_LL_B1		Very low alarm
%MW@.chann.4:X13	STS_DEVH_B1		High PV Setpoint (>0) deviation alarm
%MW@.chann.4:X14	STS_DEVL_B1		Low PV Setpoint (<0) deviation alarm
%MW@.chann.4:X15	STS_THLD_DONE_B1		Totalizer threshold reached
%MW@.chann.5	STATUS1_C1	Not applicable	Word containing the various loop controller status bits of loop 1
%MW@.chann.5:X0	STS_AT_RUNNING_B1		Autotuning in progress

Address	Parameter name	Default value	Comment
%MW@.chann.5:X1	STS_M_A_B1		State of PID operating mode
%MW@.chann.5:X2	STS_FF_Sim_B1		FF process value simulation status
%MW@.chann.5:X6	STS_TOP_NEXT_CYCLE		Sampling pulse in next cycle
%MW@.chann.5:X7	STS_TOP_CURRENT_CYCLE		Sampling pulse in current cycle
%MW@.chann.5:X8	STS_TR_S		Tracking of global loop in progress
%MW@.chann.5:X9	STS_M_A		Global Manu/ Auto
%MW@.chann.5:X10	STS_RAISE1		Opening command (global loop)
%MW@.chann.5:X11	STS_LOWER1		Closing command (global loop)
%MW@.chann.5:X12	STS_RAISE2		Opening command of output 2 branch (global loop)
%MW@.chann.5:X13	STS_LOWER2		Closing command of output 2 branch (global loop)
%MW@.chann.5:X14	STS_OUT_L_LIM		Upper limit reached for the PID output selected (global loop)
%MW@.chann.5:X15	STS_OUT_H_LIM		Lower limit reached for the PID output selected (global loop)
%MW@.chann.6	STATUS2_C1	Not applicable	Word containing the precise diagnostics of the various warnings (FF setpoint process value)
%MW@.chann.6:X0	Xi_WARN		Control warning for Xi parameters
%MW@.chann.6:X1	Yi_WARN		Control warning for Yi parameters
%MW@.chann.6:X2	RATIO_WARN		Control warning for RATIO_MIN and MAX parameters
%MW@.chann.6:X3	FF_CALC_WARN		Feed Forward calculation warning
%MW@.chann.6:X4	FF_FLOAT_WARN		Feed Forward floating point warning

Address	Parameter name	Default value	Comment
%MW@.chann.6:X5	OUT_FF__WARN	Canceled (reserved)	Control warning for OUTFF_INF and SUP parameters
%MW@.chann.6:X8	INP_MINR1_B1_WARN		Control warning for INP_MINR1 and MAXR1 parameters
%MW@.chann.6:X9	INP_MINR2_B1_WARN		Control warning for INP_MINR2 and MAXR2 parameters
%MW@.chann.6:X10	SP_INF_B1_WARN		Control warning for SP_INF and SUP parameters
%MW@.chann.6:X11	SP_CALC_B1_WARN		Setpoint calculation warning
%MW@.chann.6:X12	SP_FLOAT_B1_WARN		Setpoint floating point warning
%MW@.chann.6:X13	OVER_TOT_B1_WARN		Totalizer overflow warning
%MW@.chann.7	STATUS0_C2	Not applicable	Word containing the various process value/ setpoint status bits of loop 2
%MW@.chann.7:X0	STS_HOLD_TOT_B2		Freezes totalizer function
%MW@.chann.7:X1	STS_PV_Sim_B2		Process value simulation status
%MW@.chann.7:X2	STS_PV_H_LIM_B2		Upper limit on process value branch (PV_SUP)
%MW@.chann.7:X3	STS_PV_L_LIM_B2		Lower limit on process value branch (PV_INF)
%MW@.chann.7:X4	STS_SP_H_LIM_B2		Upper limit on setpoint branch
%MW@.chann.7:X5	STS_SP_B_LIM_B2		Lower limit on setpoint branch
%MW@.chann.7:X6	STS_L_R_B2	R/L Init	State of the setpoint selected Remote/Local
%MW@.chann.7:X8	STS_ALARMS_B2		Process value alarms sigma
%MW@.chann.7:X9	STS_HH_B2		Very high alarm
%MW@.chann.7:X10	STS_H_B2		High alarm
%MW@.chann.7:X11	STS_L_B2		Low alarm
%MW@.chann.7:X12	STS_LL_B2		Very low alarm



Address	Parameter name	Default value	Comment
%MW@.chann.7:X13	STS_DEVH_B2		High threshold of Process value Setpoint deviation (>0)
%MW@.chann.7:X14	STS_DEVL_B2		Low threshold of Process value Setpoint deviation (<0)
%MW@.chann.7:X15	STS_THLD_DONE_B2		Totalizer threshold reached
%MW@.chann.9	STATUS1_C2	Not applicable	Word containing the various loop controller / setpoint status bits of loop 2
%MW@.chann.8:X0	STS_AT_RUNNING_B2		Autotuning in progress
%MW@.chann.8:X1	STS_M_A_B2		State of PID operating mode
%MW@.chann.8:X8	INP_MINR1_B2_WARN		Control warning for INP_MINR1 and MAXR1 parameters
%MW@.chann.8:X9			
%MW@.chann.8:X10	SP_INF_B2_WARN		Control warning for SP_INF and SUP parameters
%MW@.chann.8:X11	SP_CALC_B2_WARN		Setpoint calculation warning
%MW@.chann.8:X12	SP_FLOAT_B2_WARN		Setpoint floating point warning
%MW@.chann.8:X13	OVER_TOT_B2_WARN		Totalizer overflow warning
%MW@.chann.9	STATUS_GLOBAL	Not applicable	Word containing the various bits of the output
%MW@.chann.9:X0	STS_POT_VAL1		Servo operation with feedback (global loop)
%MW@.chann.9:X1	STS_POT_VAL2		Servo operation with feedback (global loop)
%MW@.chann.9:X2	STS_RAISE STOP1		Opening stop reached on Servomotor (global loop)
%MW@.chann.9:X3	STS_LOWER STOP1		Closing stop reached on Servomotor (global loop)
%MW@.chann.9:X4	STS_RAISE STOP2		Opening stop reached on Servomotor (global loop)

Address	Parameter name	Default value	Comment
%MW@.chann.9:X5	STS_LOWER STOP2		Closing stop reached on Servomotor (global loop)
%MW@.chann.9:X8	STS_AS		Selector set to autoselector
%MW@.chann.9:X9	STS_DIR1		Selector set to output of PID1
%MW@.chann.9:X10	STS_DIR2		Selector set to output of PID2
%MW@.chann.9:X11	STS_SEL_PID1		At 1 ; Output selected = output of PID1
%MW@.chann.9:X12			At 0 : Output selected = output of PID2
%MW@.chann.10	STATUS_AT	Not applicable	Word containing the autotuning diagnostics
%MW@.chann.10:X0	AT_FAILED		Autotuning failed
%MW@.chann.10:X1	AT_ABORTED		Autotuning diagnostics interrupted
%MW@.chann.10:X2	AT_ERR_PARAM		Autotuning diagnostics parameter error
%MW@.chann.10:X3	AT_ERR_PWF_OR_EFB_FAILURE		Autotuning diagnostics system error or power outage
%MW@.chann.10:X4	AT_ERR_SATUR		Autotuning diagnostics saturation of the process value
%MW@.chann.10:X5	AT_ERR_DV_TOO_SMALL		Autotuning diagnostics process value deviation too small
%MW@.chann.10:X6	AT_ERR_TSAMP_HIGH		Autotuning diagnostics Sampling period too long
%MW@.chann.10:X7	AT_ERR_INCONSISTENT_RESPONSE		Autotuning diagnostics inconsistent response
%MW@.chann.10:X8	AT_ERR_NOT_STAB_INIT		Autotuning diagnostics process value initially unstable
%MW@.chann.10:X9	AT_ERR_TMAX_TOO_SMALL		Autotuning diagnostics step function duration too short
%MW@.chann.10:X10	AT_ERR_NOISE_TOO_HIGH		Autotuning diagnostics PV noise too high

Address	Parameter name	Default value	Comment
%MW@.chann.10:X11	AT_ERR_TMAX_TOO_HIGH		Autotuning diagnostics step function duration too long
%MW@.chann.10:X12	AT_WARN_OVERSHOOT		Autotuning diagnostics overshoot greater than 10%
%MW@.chann.10:X13	AT_WARN_UNDERSHOOT		Autotuning diagnostics undershoot too great
%MW@.chann.10:X14	AT_WARN_UNSYMMETRICAL_PLANT		Autotuning diagnostics non-symmetrical process
%MW@.chann.10:X15	AT_WARN_INTEGRATING_PLANT		Autotuning diagnostics integrating process
%MW@.chann.11	Ordre Commande		
%MD@.chann.12	Paramètre Commande		

### 6.6-3 Process control language objects

Address	Parameter name	Default value	Comment
%MF@.chann.14	AT_STEP	10	Amplitude of autotuning step function
%MF@.chann.16	AT_TMAX	100	Duration of autotuning step function
%MF@.chann.18	AT_PERF	0.5	Autotuning stability criterion
%MF@.chann.20	T_ECH	0.3	Sampling period (common to both PID)
%MF@.chann.22	OUT1	Not applicable	Command value output 1
%MF@.chann.24	OUT2	Not applicable	Command value output 2
%MF@.chann.26	OUTD	Not applicable	Global loop command variation value
%MF@.chann.28	OUT_MAN	Not applicable	Global command value (Value of the output of the loop controller selected after processing with OUTFRATE and limitations)
%MF@.chann.30	OUTFF_C1	Not applicable	Value of Feed Forward action in Physical scale
%MF@.chann.32	OUT_MAN_C1	Not applicable	Command value (output of loop controller 1)
%MF@.chann.34	DEV_C1	Not applicable	Process value setpoint deviation
%MF@.chann.36	PV_C1	Not applicable	Process value in Physical scale
%MF@.chann.38	SP_C1	Not applicable	Setpoint value in Physical scale
%MF@.chann.40	PV_INF_C1	0.	Process value lower limit
%MF@.chann.42	PV_SUP_C1	100.	Process value upper limit
%MF@.chann.44	KP_C1	1.0	Proportional coefficient
%MF@.chann.46	TI_C1	0.0	Integral time
%MF@.chann.48	TD_C1	0.0	Derivative time
%MF@.chann.50	OUTBIAS_C1	0.0	Bias on the PID controller output

Address	Parameter name	Default value	Comment
%MF@.chann.52	INT_BAND_C1	0.0	Integral band
%MF@.chann.54	DBAND_C1	0.0	Dead band on the deviation
%MF@.chann.56	KD_C1	10	Derivative filtering
%MF@.chann.58	SP_INF_C1	0.0	Setpoint lower limit
%MF@.chann.60	SP_SUP_C1	100.	Setpoint upper limit
%MF@.chann.62	PV_LL_C1	5.	PV very low threshold
%MF@.chann.64	PV_L_C1	5.	PV low threshold
%MF@.chann.66	PV_H_C1	95.	PV high threshold
%MF@.chann.68	PV_HH_C1	95.	PV very high threshold
%MF@.chann.70	RATIO_C1	1.0	Ratio value
%MF@.chann.72	RATIO_MIN_C1	0.	Minimum Ratio value
%MF@.chann.74	RATIO_MAX_C1	100	Maximum Ratio value
%MF@.chann.76	RATIO_BIAS_C1	0	Ratio bias value
%MF@.chann.78	DEV_L_C1	-5	Low deviation threshold
%MF@.chann.80	DEV_H_C1	5	High deviation threshold
%MF@.chann.82	T_C1	0.0	Process value filtering time
%MF@.chann.84	K_FILT_C1	1.	Multiplication coefficient for process value filtering
%MF@.chann.86	FILT_OUT_C1	Not applicable	Filter output value
%MF@.chann.88	SQR_OUT_C1	Not applicable	Square root output value
%MF@.chann.90	E2_IN_C1	1428	Value of input of Segment S2
%MF@.chann.92	E3_IN_C1	2857	Value of input of Segment S3
%MF@.chann.94	E4_IN_C1	4285	Value of input of Segment S4
%MF@.chann.96	E5_IN_C1	5714	Value of input of Segment S5
%MF@.chann.98	E6_IN_C1	7143	Value of input of Segment S6
%MF@.chann.100	E7_IN_C1	8571	Value of input of Segment S7

Address	Parameter name	Default value	Comment
%MF@.chann.102	E2_OUT_C1	14.28	Value of output of Segment S2
%MF@.chann.104	E3_OUT_C1	28.57	Value of output of Segment S3
%MF@.chann.106	E4_OUT_C1	42.85	Value of output of Segment S4
%MF@.chann.108	E5_OUT_C1	57.14	Value of output of Segment S5
%MF@.chann.110	E6_OUT_C1	71.43	Value of output of Segment S6
%MF@.chann.112	E7_OUT_C1	8571	Value of output of Segment S7
%MF@.chann.114	THLD_C1	1E+8	Totalizer limit
%MF@.chann.116	R_RATE_C1	0.0	Setpoint increase speed limit
%MF@.chann.118	D_RATE_C1	0.0	Setpoint decrease speed limit
%MF@.chann.120	SPEED_LIM_OUT_C1	Not applicable	Setpoint speed limiter output value
%MF@.chann.122	INP_MINR1_C1	0.0	Low scale of setpoint R1
%MF@.chann.124	INP_MAXR1_C1	100.0	High scale of setpoint R1
%MF@.chann.126	INP_MINR2_C1	0.0	Low scale of setpoint R2
%MF@.chann.128	INP_MAXR2_C1	100.0	High scale of setpoint R2
%MF@.chann.130	T1_FF_C1	0.0	Feed Forward process value filtering time
%MF@.chann.132	T2_FF_C1	0.0	Feed Forward process value filtering time
%MF@.chann.134	OUT_FF_INF_C1	0.	Feed Forward process value lower limit
%MF@.chann.136	OUT_FF_SUP_C1	100.	Feed Forward process value upper limit
%MF@.chann.138	KP_PREV_C1	Not applicable	Value before autotuning of proportional coefficient
%MF@.chann.140	TI_PREV_C1	Not applicable	Value before autotuning of integral coefficient
%MF@.chann.142	TD_PREV_C1	Not applicable	Value before autotuning of derivative coefficient

Address	Parameter name	Default value	Comment
%MF@.chann.144	OUT_MAN_C2	Not applicable	Command value of PID no. 2
%MF@.chann.146	DEV_C2	Not applicable	Process value setpoint deviation
%MF@.chann.148	PV_C2	Not applicable	Process value in Physical scale
%MF@.chann.150	SP_C2	Not applicable	Setpoint value in Physical scale
%MF@.chann.152	PV_INF_C2	0.	Process value lower limit
%MF@.chann.154	PV_SUP_C2	100.	Process value upper limit
%MF@.chann.156	KP_C2	1.0	Proportional coefficient
%MF@.chann.158	TI_C2	0.0	Integral time
%MF@.chann.160	TD_C2	0.0	Derivative time
%MF@.chann.162	OUTBIAS_C2	0.0	Bias on the PID controller output
%MF@.chann.164	INT_BAND_C2	0.0	Integral band
%MF@.chann.166	DBAND_C2	0.0	Dead band on the deviation
%MF@.chann.168	SP_INF_C2	0.0	Setpoint lower limit
%MF@.chann.170	SP_SUP_C2	100.	Setpoint upper limit
%MF@.chann.172	PV_LL_C2	5.	PV very low threshold
%MF@.chann.174	PV_L_C2	5.	PV low threshold
%MF@.chann.176	PV_H_C2	95.	PV high threshold
%MF@.chann.178	PV_HH_C2	95.	PV very high threshold
%MF@.chann.180	DEV_L_C2	-5	Low deviation threshold
%MF@.chann.182	DEV_H_C2	5	High deviation threshold
%MF@.chann.184	SQR_OUT_C2	Not applicable	Square root output value
%MF@.chann.186	THLD_C2	1E+8	Totalizer limit
%MF@.chann.188	R_RATE_C2	0.0	Setpoint increase speed limit
%MF@.chann.190	D_RATE_C2	0.0	Setpoint decrease speed limit
%MF@.chann.192	SPEED_LIM_OUT_C2	Not applicable	Setpoint speed limiter output value

Address	Parameter name	Default value	Comment
%MF@.chann.194	INP_MINR1_C2	0.0	Low scale of setpoint R1
%MF@.chann.196	INP_MAXR1_C2	100.0	High scale of setpoint R1
%MF@.chann.198	KP_PREV_C2	Not applicable	Value before autotuning of proportional coefficient
%MF@.chann.200	TI_PREV_C2	Not applicable	Value before autotuning of integral coefficient
%MF@.chann.202	TD_PREV_C2	Not applicable	Value before autotuning of derivative coefficient
%MF@.chann.204	OUTRATE	0.0	Output 1 speed limit
%MF@.chann.206	OUTRATE2	0.0	Output 2 speed limit
%MF@.chann.208	OUT1_INF	0.	Lower limit of output 1
%MF@.chann.210	OUT1_SUP	100.0	Upper limit of output 1
%MF@.chann.212	OUT2_INF	0.	Lower limit of output 2
%MF@.chann.214	OUT2_SUP	100.0	Upper limit of output 2
%MF@.chann.216	OUT1_TH1	0.	Threshold 1 of output 1 of Hot/Cold or Split/Range
%MF@.chann.218	OUT1_TH2	50.0	Threshold 2 of output 1 of Hot/Cold or Split/Range
%MF@.chann.220	OUT2_TH1	50.0	Threshold 1 of output 2 of Hot/Cold or Split/Range
%MF@.chann.222	OUT2_TH2	100.	Threshold 2 of output 2 of Hot/Cold or Split/Range
%MF@.chann.224	T_MOTOR1	10.	Opening time of valve controlled by Servomotor
%MF@.chann.226	T_MINI1	0.	Minimum opening time of valve controlled by Servomotor
%MF@.chann.228	T_MOTOR2	10.	Opening time of valve controlled by Servomotor
%MF@.chann.230	T_MINI2	0.	Minimum opening time of valve controlled by Servomotor



Address	Parameter name	Default value	Comment
%MF@.chann.232	KS	1.0	IMC static gain
%MF@.chann.234	T1	1.0	Time constant in OL
%MF@.chann.236	T_DELAY	0.0	Current pure delay
%MF@.chann.238	CC_PERF	0.1	OL / CL time ratio
%MF@.chann.240		Not applicable	
%MF@.chann.242		Not applicable	
%MF@.chann.244		Not applicable	
%MF@.chann.246		Not applicable	
%MW@.chann.248	PV_C1 Simulée	Not applicable	Simulated process value
%MW@.chann.249	PV_C2 Simulée	Not applicable	Simulated process value
%MW@.chann.250	FF_C1 Simulée	Not applicable	Simulated feed forward input

## 6.7 Language objects associated with the setpoint programmer

### 6.7-1 Configuration language objects

Address	Parameter name	Default value	Comment
%KW@.chann.0	CONFIG_1		Word containing the various configuration bits of profile 1
%KW@.chann.0:X0	Palier garanti	no (0)	Enable guaranteed dwell time function (0 : no, 1 : yes)
%KW@.chann.0:X1	Type de palier	0	Type of maintain on guaranteed dwell time : 2 bits
%KW@.chann.0:X2	Type de palier	0	
%KW@.chann.0:X3	Démarrage	With bump (0)	Starting with bump (0 : SP0) or bumpless (1 : PV)
%KW@.chann.0:X4	Réitération	Not continuous (0)	Continuous reiteration of profile (1) or not (0)
%KW@.chann.0:X5	Type de reiteration	With bump (0)	Reiteration with bump (0 : SPi) or bumpless (1 : PV)
%KW@.chann.1	CONFIG_2		Word containing the various configuration bits of profile 2
%KW@.chann.1:X0	Palier garanti	no (0)	Enable guaranteed dwell time function (0 : no, 1 : yes)
%KW@.chann.1:X1	Type de palier	0	Type of maintain on guaranteed dwell time : 2 bits
%KW@.chann.1:X2	Type de palier	0	
%KW@.chann.1:X3	Démarrage	With bump (0)	Starting with bump (0 : SP0) or bumpless (1 : PV)
%KW@.chann.1:X4	Réitération	Not continuous (0)	Continuous reiteration of profile (1) or not (0)
%KW@.chann.1:X5	Type de réitération	With bump (0)	Reiteration with bump (0 : SPi) or bumpless (1 : PV)
%KW@.chann.2	CONFIG_3		Word containing the various configuration bits of profile 3
%KW@.chann.2:X0	Palier garanti	No (0)	Enable guaranteed dwell time function (0 : no, 1 : yes)

Address	Parameter name	Default value	Comment
%KW@.chann.2:X1	Type de palier	0	Type of maintain on guaranteed dwell time : 2 bits
%KW@.chann.2:X2	Type de palier	0	
%KW@.chann.2:X3	Démarrage	With bump (0)	Starting with bump (0 : SP0) or bumpless (1 : PV)
%KW@.chann.2:X4	Réitération	Not continuous (0)	Continuous reiteration of profile (1) or not (0)
%KW@.chann.2:X5	Type de réitération	With bump (0)	Reiteration with bump (0 : SPi) or bumpless (1 : PV)
%KW@.chann.3	CONFIG_4		Word containing the various configuration bits of profile 4
%KW@.chann.3:X0	Palier garanti	No (0)	Enable guaranteed dwell time function (0 : no, 1 : yes)
%KW@.chann.3:X1	Type de palier	0	Type of maintain on guaranteed dwell time : 2 bits
%KW@.chann.3:X2	Type de palier	0	
%KW@.chann.3:X3	Démarrage	With bump (0)	Starting with bump (0 : SP0) or bumpless (1 : PV)
%KW@.chann.3:X4	Réitération	Not continuous (0)	Continuous reiteration of profile (1) or not (0)
%KW@.chann.3:X5	Type de réitération	With bump (0)	Reiteration with bump (0 : SPi) or bumpless (1 : PV)
%KW@.chann.4	CONFIG_5		Word containing the various configuration bits of profile 5
%KW@.chann.4:X0	Palier garanti	No (0)	Enable guaranteed dwell time function (0 : no, 1 : yes)
%KW@.chann.4:X1	Type de palier	0	Type of maintain on guaranteed dwell time : 2 bits
%KW@.chann.4:X2	Type de palier	0	
%KW@.chann.4:X3	Démarrage	With bump (0)	Starting with bump (0 : SP0) or bumpless (1 : PV)
%KW@.chann.4:X4	Réitération	Not continuous (0)	Continuous reiteration of profile (1) or not (0)

Address	Parameter name	Default value	Comment
%KW@.chann.4:X5	Type de réitération	With bump (0)	Reiteration with bump (0 : SPi) or bumpless (1 : PV)
%KW@.chann.5	CONFIG_6		Word containing the various configuration bits of profile 6
%KW@.chann.5:X0	Palier garanti	No(0)	Enable guaranteed dwell time function (0 : no, 1 : yes)
%KW@.chann.5:X1	Type de palier	0	Type of maintain on guaranteed dwell time : 2 bits
%KW@.chann.5:X2	Type de palier	0	
%KW@.chann.5:X3	Démarrage	With bump (0)	Starting with bump (0 : SP0) or bumpless (1 : PV)
%KW@.chann.5:X4	Réitération	Not continuous (0)	Continuous reiteration of profile (1) or not (0)
%KW@.chann.5:X5	Type de réitération	With bump (0)	Reiteration with bump (0 : SPi) or bumpless (1 : PV)
%KW@.chann.6	USED_PF1	1	Number of 1st segment of profile 1
%KW@.chann.7	USED_PF2	9	Number of 1st segment of profile 2
%KW@.chann.8	USED_PF3	17	Number of 1st segment of profile 3
%KW@.chann.9	USED_PF4	25	Number of 1st segment of profile 4
%KW@.chann.10	USED_PF5	33	Number of 1st segment of profile 5
%KW@.chann.11	USED_PF6	41	Number of 1st segment of profile 6
%KW@.chann.12	NB_SEG_PF1	8	Number of segments used in profile 1
%KW@.chann.13	NB_SEG_PF2	8	Number of segments used in profile 2
%KW@.chann.14	NB_SEG_PF3	8	Number of segments used in profile 3
%KW@.chann.15	NB_SEG_PF4	8	Number of segments used in profile 4
%KW@.chann.16	NB_SEG_PF5	8	Number of segments used in profile 5

Address	Parameter name	Default value	Comment
%KW@.chann.17	NB_SEG_PF6	8	Number of segments used in profile 6
%KW@.chann.18	NO_SEG_RT1	1	Number of reiteration start segment for profile 1
%KW@.chann.19	NO_SEG_RT2	9	Number of reiteration start segment for profile 2
%KW@.chann.20	NO_SEG_RT3	17	Number of reiteration start segment for profile 3
%KW@.chann.21	NO_SEG_RT4	25	Number of reiteration start segment for profile 4
%KW@.chann.22	NO_SEG_RT5	33	Number of reiteration start segment for profile 5
%KW@.chann.23	NO_SEG_RT6	41	Number of reiteration start segment for profile 6
%KW@.chann.24	CONF_SEG1	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.25	CONF_SEG2	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.26	CONF_SEG3	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.27	CONF_SEG4	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.28	CONF_SEG5	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.29	CONF_SEG6	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.30	CONF_SEG7	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.31	CONF_SEG8	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment

Address	Parameter name	Default value	Comment
%KW@.chann.32	CONF_SEG9	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.33	CONF_SEG10	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.34	CONF_SEG11	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.35	CONF_SEG12	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.36	CONF_SEG13	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.37	CONF_SEG14	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.38	CONF_SEG15	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.39	CONF_SEG16	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.40	CONF_SEG17	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.41	CONF_SEG18	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.42	CONF_SEG19	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.43	CONF_SEG20	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.44	CONF_SEG21	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.45	CONF_SEG22	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment

Address	Parameter name	Default value	Comment
%KW@.chann.46	CONF_SEG23	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.47	CONF_SEG24	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.48	CONF_SEG25	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.49	CONF_SEG26	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.50	CONF_SEG27	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.51	CONF_SEG28	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.52	CONF_SEG29	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.53	CONF_SEG30	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.54	CONF_SEG31	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.55	CONF_SEG32	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.56	CONF_SEG33	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.57	CONF_SEG34	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.58	CONF_SEG35	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.59	CONF_SEG36	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment

Address	Parameter name	Default value	Comment
%KW@.chann.60	CONF_SEG37	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.61	CONF_SEG38	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.62	CONF_SEG39	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.63	CONF_SEG40	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.64	CONF_SEG41	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.65	CONF_SEG42	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.66	CONF_SEG43	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.67	CONF_SEG44	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.68	CONF_SEG45	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.69	CONF_SEG46	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.70	CONF_SEG47	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.71	CONF_SEG48	0	Outputs (bits 8-15) PG (bit 5) Type (bit 4) Unit (bits 0-3) of segment
%KW@.chann.72	SPP_NAME1		8 characters on 4 times 2 bytes
%KW@.chann.73	SPP_NAME2		
%KW@.chann.74	SPP_NAME3		
%KW@.chann.75	SPP_NAME4		



**6.7-2 Fault and diagnostic language objects**

Address	Parameter name	Default value	Comment
%MW@.chann.0	EXCH		Current exchange (for IOB)
%MW@.chann.1	CR_EXCH		Current exchange report (for IOB)
%MW@.chann.2	STATUS1		The first 6 bits are standard to all IOB
%MW@.chann.2:X7	WARN		Warning sigma
%MW@.chann.2:X8	ERR_CALC		Calculation error
%MW@.chann.2:X9	ERR_FLOAT		Floating point error
%MW@.chann.3	STATUS2		Status of control outputs, SPP freezing, state of profile
%MW@.chann.3:X0	STOR0	0	Status of control output 0
%MW@.chann.3:X1	STOR1	0	Status of control output 1
%MW@.chann.3:X2	STOR2	0	Status of control output 2
%MW@.chann.3:X3	STOR3	0	Status of control output 3
%MW@.chann.3:X4	STOR4	0	Status of control output4
%MW@.chann.3:X5	STOR5	0	Status of control output 5
%MW@.chann.3:X6	STOR6	0	Status of control output 6
%MW@.chann.3:X7	STOR7	0	Status of control output 7
%MW@.chann.3:X8	STS_SPP_HOLD	0	Freezing of setpoint programmer function
%MW@.chann.3:X9	STS_INIT	1	1 : all profiles are in INIT
%MW@.chann.3:X10	STS_RUN	0	1 : current profile is in RUN
%MW@.chann.3:X11	STS_STOP	0	1 : current profile is in STOP
%MW@.chann.3:X12	STS_HOLD_PG	0	1 : the guaranteed dwell time is inhibited

Address	Parameter name	Default value	Comment
%MW@.chann.3:X15	STS_ERR_SEG	0	Parameter error on current segment
%MW@.chann.4	STATUS3		Indication of warnings on profiles 1 to 4
%MW@.chann.4:X0	WRN1_RMP_SP	Not applicable	A ramp of profile 1 has 2 identical setpoints
%MW@.chann.4:X1	WRN1_RMP_0	Not applicable	A ramp of profile 1 has a zero speed
%MW@.chann.4:X2	WRN1_PLR_SP	Not applicable	A dwell step of profile 1 has 2 various setpoints
%MW@.chann.4:X3	WRN1_PLR_THLD	Not applicable	A guaranteed dwell time of profile 1 has a zero THLD?
%MW@.chann.4:X4	WRN2_RMP_SP	Not applicable	A ramp of profile 2 has 2 identical setpoints
%MW@.chann.4:X5	WRN2_RMP_0	Not applicable	A ramp of profile 2 has a zero speed
%MW@.chann.4:X6	WRN2_PLR_SP	Not applicable	A dwell step of profile 2 has 2 various setpoints
%MW@.chann.4:X7	WRN2_PLR_THLD	Not applicable	A guaranteed dwell time of profile 2 has a zero THLD?
%MW@.chann.4:X8	WRN3_RMP_SP	Not applicable	A ramp of profile 3 has 2 identical setpoints
%MW@.chann.4:X9	WRN3_RMP_0	Not applicable	A ramp of profile 3 has a zero speed
%MW@.chann.4:X10	WRN3_PLR_SP	Not applicable	A dwell step of profile 3 has 2 various setpoints
%MW@.chann.4:X11	WRN3_PLR_THLD	Not applicable	A guaranteed dwell time of profile 3 has a zero THLD?
%MW@.chann.4:X12	WRN4_RMP_SP	Not applicable	A ramp of profile 4 has 2 identical setpoints
%MW@.chann.4:X13	WRN4_RMP_0	Not applicable	A ramp of profile 4 has a zero speed
%MW@.chann.4:X14	WRN4_PLR_SP	Not applicable	A dwell step of profile 4 has 2 various setpoints
%MW@.chann.4:X15	WRN4_PLR_THLD	Not applicable	A guaranteed dwell time of profile 4 has a zero THLD?
%MW@.chann.5	STATUS4		Indication of warnings on profiles 5 and 6
%MW@.chann.5:X0	WRN5_RMP_SP	Not applicable	A ramp of profile 5 has 2 identical setpoints

Address	Parameter name	Default value	Comment
%MW@.chann.5:X1	WRN5_RMP_0	Not applicable	A ramp of profile 5 has a zero speed
%MW@.chann.5:X2	WRN5_PLR_SP	Not applicable	A dwell step of profile 5 has 2 various setpoints
%MW@.chann.5:X3	WRN5_PLR_THLD	Not applicable	A guaranteed dwell time of profile 5 has a zero THLD?
%MW@.chann.5:X4	WRN6_RMP_SP	Not applicable	A ramp of profile 6 has 2 identical setpoints
%MW@.chann.5:X5	WRN6_RMP_0	Not applicable	A ramp of profile 6 has a zero speed
%MW@.chann.5:X6	WRN6_PLR_SP	Not applicable	A dwell step of profile 6 has 2 various setpoints
%MW@.chann.5:X7	WRN6_PLR_THLD	Not applicable	A guaranteed dwell time of profile 6 has a zero THLD?
%MW@.chann.7	CMD_ORDER		Command order (single word)
%MD@.chann.8	CMD_PARAM		Command parameter (double word)
%MW@.chann.10	CUR_PF	Not applicable	Number of current profile
%MW@.chann.11	SEG_OUT	Not applicable	Number of current segment
%MW@.chann.12	CUR_ITER	Not applicable	Number of current iteration
%MW@.chann.13	NB_RT_PF1	1	Number of reiteration of profile 1
%MW@.chann.14	NB_RT_PF2	1	Number of reiteration of profile 2
%MW@.chann.15	NB_RT_PF3	1	Number of reiteration of profile 3
%MW@.chann.16	NB_RT_PF4	1	Number of reiteration of profile 4
%MW@.chann.17	NB_RT_PF5	1	Number of reiteration of profile 5
%MW@.chann.18	NB_RT_PF6	1	Number of reiteration of profile 6

### 6.7-3 Process control language objects

Address	Parameter name	Default value	Comment
%MF@.chann.20	SP	Not applicable	Value of calculated setpoint (output)
%MF@.chann.22	TOTAL_TIME	Not applicable	Value of total time elapsed (freezes included)
%MF@.chann.24	CUR_TIME	Not applicable	Value of time elapsed on the current segment (freezes included)
%MF@.chann.26	THLD_PF1	0.0	Value of guaranteed dwell time threshold of profile 1
%MF@.chann.28	THLD_PF2	0.0	Value of guaranteed dwell time threshold of profile 2
%MF@.chann.30	THLD_PF3	0.0	Value of guaranteed dwell time threshold of profile 3
%MF@.chann.32	THLD_PF4	0.0	Value of guaranteed dwell time threshold of profile 4
%MF@.chann.34	THLD_PF5	0.0	Value of guaranteed dwell time threshold of profile 5
%MF@.chann.36	THLD_PF6	0.0	Value of guaranteed dwell time threshold of profile 6
%MF@.chann.38	SP0_PF1	0.0	Value of initial setpoint of profile 1
%MF@.chann.40	SP0_PF2	0.0	Value of initial setpoint of profile 2
%MF@.chann.42	SP0_PF3	0.0	Value of initial setpoint of profile 3
%MF@.chann.44	SP0_PF4	0.0	Value of initial setpoint of profile 4
%MF@.chann.46	SP0_PF5	0.0	Value of initial setpoint of profile 5
%MF@.chann.48	SP0_PF6	0.0	Value of initial setpoint of profile 6
%MF@.chann.50	SP1	0.0	Setpoint to be reached by segment 1
%MF@.chann.52	VAL1	0.0	Time or speed value for segment 1
%MF@.chann.54	SP2	0.0	Setpoint to be reached by segment 2
%MF@.chann.56	VAL2	0.0	Time or speed value for segment 2

Address	Parameter name	Default value	Comment
%MF@.chann.58	SP3	0.0	Setpoint to be reached by segment 3
%MF@.chann.60	VAL3	0.0	Time or speed value for segment 3
%MF@.chann.62	SP4	0.0	Setpoint to be reached by segment 4
%MF@.chann.64	VAL4	0.0	Time or speed value for segment 4
%MF@.chann.66	SP5	0.0	Setpoint to be reached by segment 5
%MF@.chann.68	VAL5	0.0	Time or speed value for segment 5
%MF@.chann.70	SP6	0.0	Setpoint to be reached by segment 6
%MF@.chann.72	VAL6	0.0	Time or speed value for segment 6
%MF@.chann.74	SP7	0.0	Setpoint to be reached by segment 7
%MF@.chann.76	VAL7	0.0	Time or speed value for segment 7
%MF@.chann.78	SP8	0.0	Setpoint to be reached by segment 8
%MF@.chann.80	VAL8	0.0	Time or speed value for segment 8
%MF@.chann.82	SP9	0.0	Setpoint to be reached by segment 9
%MF@.chann.84	VAL9	0.0	Time or speed value for segment 9
%MF@.chann.86	SP10	0.0	Setpoint to be reached by segment 10
%MF@.chann.88	VAL10	0.0	Time or speed value for segment 10
%MF@.chann.90	SP11	0.0	Setpoint to be reached by segment 11
%MF@.chann.92	VAL11	0.0	Time or speed value for segment 11
%MF@.chann.94	SP12	0.0	Setpoint to be reached by segment 12
%MF@.chann.96	VAL12	0.0	Time or speed value for segment 12

Address	Parameter name	Default value	Comment
%MF@.chann.98	SP13	0.0	Setpoint to be reached by segment 13
%MF@.chann.100	VAL13	0.0	Time or speed value for segment 13
%MF@.chann.102	SP14	0.0	Setpoint to be reached by segment 14
%MF@.chann.104	VAL14	0.0	Time or speed value for segment 14
%MF@.chann.106	SP15	0.0	Setpoint to be reached by segment 15
%MF@.chann.108	VAL15	0.0	Time or speed value for segment 15
%MF@.chann.110	SP16	0.0	Setpoint to be reached by segment 16
%MF@.chann.112	VAL16	0.0	Time or speed value for segment 16
%MF@.chann.114	SP17	0.0	Setpoint to be reached by segment 17
%MF@.chann.116	VAL17	0.0	Time or speed value for segment 17
%MF@.chann.118	SP18	0.0	Setpoint to be reached by segment 18
%MF@.chann.120	VAL18	0.0	Time or speed value for segment 18
%MF@.chann.122	SP19	0.0	Setpoint to be reached by segment 19
%MF@.chann.124	VAL19	0.0	Time or speed value for segment 19
%MF@.chann.126	SP20	0.0	Setpoint to be reached by segment 20
%MF@.chann.128	VAL20	0.0	Time or speed value for segment 20
%MF@.chann.130	SP21	0.0	Setpoint to be reached by segment 21
%MF@.chann.132	VAL21	0.0	Time or speed value for segment 21
%MF@.chann.134	SP22	0.0	Setpoint to be reached by segment 22
%MF@.chann.136	VAL22	0.0	Time or speed value for segment 22

Address	Parameter name	Default value	Comment
%MF@.chann.138	SP23	0.0	Setpoint to be reached by segment 23
%MF@.chann.140	VAL23	0.0	Time or speed value for segment 23
%MF@.chann.142	SP24	0.0	Setpoint to be reached by segment 24
%MF@.chann.144	VAL24	0.0	Time or speed value for segment 24
%MF@.chann.146	SP25	0.0	Setpoint to be reached by segment 25
%MF@.chann.148	VAL25	0.0	Time or speed value for segment 25
%MF@.chann.150	SP26	0.0	Setpoint to be reached by segment 26
%MF@.chann.152	VAL26	0.0	Time or speed value for segment 26
%MF@.chann.154	SP27	0.0	Setpoint to be reached by segment 27
%MF@.chann.156	VAL27	0.0	Time or speed value for segment 27
%MF@.chann.158	SP28	0.0	Setpoint to be reached by segment 28
%MF@.chann.160	VAL28	0.0	Time or speed value for segment 28
%MF@.chann.162	SP29	0.0	Setpoint to be reached by segment 29
%MF@.chann.164	VAL29	0.0	Time or speed value for segment 29
%MF@.chann.166	SP30	0.0	Setpoint to be reached by segment 30
%MF@.chann.168	VAL30	0.0	Time or speed value for segment 30
%MF@.chann.170	SP31	0.0	Setpoint to be reached by segment 31
%MF@.chann.172	VAL31	0.0	Time or speed value for segment 31
%MF@.chann.174	SP32	0.0	Setpoint to be reached by segment 32
%MF@.chann.176	VAL32	0.0	Time or speed value for segment 32

Address	Parameter name	Default value	Comment
%MF@.chann.178	SP33	0.0	Setpoint to be reached by segment 33
%MF@.chann.180	VAL33	0.0	Time or speed value for segment 33
%MF@.chann.182	SP34	0.0	Setpoint to be reached by segment 34
%MF@.chann.184	VAL34	0.0	Time or speed value for segment 34
%MF@.chann.186	SP35	0.0	Setpoint to be reached by segment 35
%MF@.chann.188	VAL35	0.0	Time or speed value for segment 35
%MF@.chann.190	SP36	0.0	Setpoint to be reached by segment 36
%MF@.chann.192	VAL36	0.0	Time or speed value for segment 36
%MF@.chann.194	SP37	0.0	Setpoint to be reached by segment 37
%MF@.chann.196	VAL37	0.0	Time or speed value for segment 37
%MF@.chann.198	SP38	0.0	Setpoint to be reached by segment 38
%MF@.chann.200	VAL38	0.0	Time or speed value for segment 38
%MF@.chann.202	SP39	0.0	Setpoint to be reached by segment 39
%MF@.chann.204	VAL39	0.0	Time or speed value for segment 39
%MF@.chann.206	SP40	0.0	Setpoint to be reached by segment 40
%MF@.chann.208	VAL40	0.0	Time or speed value for segment 40
%MF@.chann.210	SP41	0.0	Setpoint to be reached by segment 41
%MF@.chann.212	VAL41	0.0	Time or speed value for segment 41
%MF@.chann.214	SP42	0.0	Setpoint to be reached by segment 42
%MF@.chann.216	VAL42	0.0	Time or speed value for segment 42



Address	Parameter name	Default value	Comment
%MF@.chann.218	SP43	0.0	Setpoint to be reached by segment 43
%MF@.chann.220	VAL43	0.0	Time or speed value for segment 43
%MF@.chann.222	SP44	0.0	Setpoint to be reached by segment 44
%MF@.chann.224	VAL44	0.0	Time or speed value for segment 44
%MF@.chann.226	SP45	0.0	Setpoint to be reached by segment 45
%MF@.chann.228	VAL45	0.0	Time or speed value for segment 45
%MF@.chann.230	SP46	0.0	Setpoint to be reached by segment 46
%MF@.chann.232	VAL46	0.0	Time or speed value for segment 46
%MF@.chann.234	SP47	0.0	Setpoint to be reached by segment 47
%MF@.chann.236	VAL47	0.0	Time or speed value for segment 47
%MF@.chann.238	SP48	0.0	Setpoint to be reached by segment 48
%MW@.chann.240	VAL48	0.0	Time or speed value for segment 48

## 6.8 Tables of exchanges for operation

### 6.8-1 Table of multiplexed parameters for a loop

This table is used by pages FAV, TUNE and ATUNE (for an XBTF-01) and by page TUNE-AT (for an XBTF-02). The first 4 words (not multiplexed) are used by monitoring pages.

Rank	Parameter	Exchange
%MWn+0	<p><b>Number of selected loop (1 word)</b>            From 0 to 29. This word is used to select the loop managed by the multiplexed table if the special XBT table is not configured. Otherwise it is not used.            This word is controlled according to word %MWn+5 or is written directly. In the case of an overshoot beyond the last loop or before the first, the word is managed according to the "roller" principle. On initialization it is set to 0.</p>	PLC<->XBT
%MWn+1	<p><b>Identifier (1 word)</b></p>	PLC->XBT
%MDn+2	<p><b>Number of paragraphs indicator (1 double word)</b>            Each bit is associated with a loop. A bit at 0 indicates that the loop only has one output. A bit at 1 indicates that the loop has 2 outputs.</p>	PLC->XBT
%MWn+4	<p><b>Write access prohibited (1 word)</b>            Write access to this table is only taken into account by the PLC if this word is at 0. This word is managed by the user application. (Default = 0)</p>	PLC->XBT
%MWn+5	<p><b>Increment/Decrement loop number (1 word)</b>            Dynamic buttons in XBT pulse mode raise the word bits controlling incrementation or decrementation of the loop number.            X0 : incrementation of loop number (XBT-F01)            X1 : decrementation of loop number (XBT-F01)            X2 : incrementation of loop number (XBT-F02)            X3 : decrementation of loop number (XBT-F02)            X15 : memorization of change of loop (internal management)  <i>These bits are processed by the channel containing the loop currently selected. The bit is taken into account on a rising edge.</i></p>	PLC<->XBT  internal PLC
%MWn+6	<p><b>Command word for toggle buttons (1 word)</b>            Each bit of the word is used to send a command to the selected loop, on change of state.            X0 : 0= change to local setpoint; 1= change to remote setpoint            X1 : 0= change to manual mode; 1= change to automatic mode            X2 : 0= stop autotuning; 1= launch autotuning            X3 : return to previous setting            X4 : acknowledge autotuning diagnostics            X5 : 0= select remote setpoint 1; 1= select remote setpoint 2</p>	PLC<->XBT

	<p>X6 :0= deactivation of output RAISE1; 1= activation of output RAISE1                  X7 : 0= deactivation of output LOWER1; 1= activation of output LOWER1                  X14 : Save parameters  <i>The selected loop takes the command into account on a rising or falling edge. The associated buttons are in toggle mode. The word is updated by the PLC depending on the current state of the loop (for bits used to send 2 distinct commands).</i></p>	
%MWn+7	<p><b>Command word for pulse buttons (1 word)</b>                  Each bit of the word is used to send a command to the selected loop. The first 4 bits are associated with dynamic buttons. The following are used for opening :</p> <p>X0 : Change setpoint mode (remote-&gt;local or local-&gt;remote depending on current mode)                  X1 : change operating mode (manu-&gt;auto or auto-&gt;manu depending on current operating mode)                  X2 : start or stop autotuning, depending on whether Autotuning is in progress or not                  X3 : return to previous setting                  X4 : acknowledge autotuning diagnostics                  X5 : select remote setpoint 1                  X6 : select remote setpoint 2                  X7 : activation of output RAISE1                  X8 : deactivation of output RAISE1                  X9 : activation of output LOWER1                  X10 : deactivation of output LOWER1                  X14 : Save parameters  <i>The selected loop takes the command into account on a rising edge. The buttons are in pulse mode.</i></p>	PLC<-XBT
%MWn+8	<p><b>Loop label (8 bytes)</b>  <i>Updated by the selected loop when the loop is selected</i></p>	PLC->XBT
%MWn+12	<p><b>Loop unit (6 bytes)</b>  <i>Updated by the selected loop when the loop is selected</i></p>	PLC->XBT
%MWn+15	<p>Loop identifier (1 word)                  1H : single/process loop : nothing                  2H : master cascade : CASCADE M                  3H : slave cascade : CASCADE S                  4H : autoselector, main loop : AUTOSELECTEUR 0                  5H : autoselector, secondary : AUTOSELECTEUR 1  <i>Updated by the selected loop when the loop is selected. used to display the nature of the loop int he various screens.</i></p>	PLC->XBT
%MWn+16	<p><b>Controller identifier (1 word)</b>                  xx1H : PID                  xx2H : simple PID</p>	PLC->XBT

xx3H : ONOFF2

xx4H : ONOFF3

xx5H : IMC.

*Updated by the selected loop when the loop is selected*

%MWn+17

**Alarm word (1 word)**

PLC->XBT

Each bit defines a various alarm :

X0 : STS\_SIGMA\_ALA (sum of alarms)

X1 : STS\_HH (PV very high threshold overshoot)

X2 : STS\_H (PV high threshold overshoot)

X3 : STS\_L (PV low threshold overshoot)

X4 : STS\_LL (PV very low threshold overshoot)

X5 : STS\_DEV\_H (positive deviation threshold overshoot)

X6 : STS\_DEV\_L (negative deviation threshold overshoot)

X14: AT\_NON\_AUTORISE

X15: NB\_BARGRAPHER\_OUT (0=1 barg.; 1=2 barg.)

*This word is updated on all cycles*

%MWn+18

PV in scale 0-10000 (1 word)

PLC->XBT

%MWn+19

SP in scale 0-10000 (1 word)

PLC->XBT

%MWn+20

OUT1 in scale 0-10000 (1 word)

PLC->XBT

%MWn+21

OUT2 in scale 0-10000 (1 word)

PLC->XBT

%MFn+22

**Zone in fast read-only (6 floating point words)**

PLC->XBT

%MFn+22 OUT\_MAN,

%MFn+24 PV,

%MFn+26 SP,

%MFn+28 OUT1,

%MFn+30 OUT2,

%MFn+32 STATUS

Contents of STATUS :

least significant (%MWn+30)

X0 : STS\_M\_A (0=manu, 1=auto)

X1 : STS\_TR\_S1 (1=tracking)

X2 : STS\_AT\_RUNNING (1=Autotuning in progress)

X3 : STS\_R\_L (0=remote, 1=local)

X4 : STS\_RAISE1 (output 1 of ONOFF or of SERVO)

X5 : STS\_LOWER1 (output 2 of ONOFF3 or of SERVO)

X6 : STS\_RAISE2 (output 1 of SERVO2)

X7 : STS\_LOWER2 (output 2 of SERVO2)

X8 : STS\_R1\_R2 (0=SP1 is selected, 1=SP2 is selected)

X9 : STS\_AS (1=autoselector in autoselection mode)

X10 : STS\_DIR1 (1=autoselector in direct main loop mode)

X11 : STS\_DIR2 (1=autoselector in direct secondary loop mode)

X12 : STS\_SEL\_PID1 (0=output of PID2 selected,  
1= output of PID1 selected)



	<p>most significant (%MWn+31) = Autotuning diagnostics (list of values)</p> <p>1=Autotuning in progress (AT_EN_COURS)</p> <p>2=Autotuning interrupted (<i>by the user or program</i>) (AT_ANNULE)</p> <p>3=AT: parameter error (<i>parameter incorrect, or value modified while autotuning in progress</i>) (AT_ERR_PARAM)</p> <p>4=AT: power outage (<i>or system error</i>) (AT_PWFAL)</p> <p>5=AT: PV or OV saturation (AT_SATUR)</p> <p>6=AT:deviation too small (AT_DV_FAIBLE)</p> <p>7=AT:under-sampling (AT_SOUS_ECH)</p> <p>8=AT: inconsistent response (AT_INCOHER)</p> <p>9=AT: PV unstable on initialization (AT_INSTB_INIT)</p> <p>10=AT: TMAX too small (AT_TMAX_PTIT)</p> <p>11=AT: noise too high (AT_BRUIT_GRD)</p> <p>12=AT_TMAX too high (AT_TMAX_GRD)</p> <p>13=AT: process insufficiently damped (AT_PROC_DEP)</p> <p>14=AT: too great an undershoot (AT_PROC_NMP)</p> <p>15=AT: non-symmetrical process (AT_PROC_DIS)</p> <p>16=AT: integrating process (AT_PROC_INT)</p> <p>%MFn+34 SPEED_LIM_OUT</p> <p><i>Updated on all cycles, for multiplexed screens</i></p>	
<hr/>		
%MFn+36	<b>Loop controller adjustment zone (20 floating point words)</b>	PLC<->XBT
%MFn+36	T_ECH,	
%MFn+38	OUT1_INF (read-only),	
%MFn+40	OUT1_SUP (read-only),	
%MFn+42	SP_INF,	
%MFn+44	SP_SUP,	
%MFn+46	OUT2_INF (read-only),	
%MFn+48	OUT2_SUP (read-only),	
%MFn+50	PV_INF (read only),	PLC<->XBT
%MFn+52	PV_SUP(read only),	
%MFn+54	KP (PID) / ONOFF_L (ONOFF) / KS (IMC),	
%MFn+56	TI (PID) / ONOFF_H (ONOFF) / T1 (IMC),	
%MFn+58	TD (PID) / HYST (ONOFF3) / T_DELAY (IMC),	
%MFn+60	OUTBIAS (PID) / - / CL_PERF (IMC),	
%MFn+62	INT_BAND (PID) / - / -,	
%MFn+64	DBAND (PID, IMC)	
%MFn+66	KD (PID except single PID)	
%MFn+68	OUTRATE1 (PID, IMC),	
%MFn+70	OUTRATE2,	
%MFn+72	PV_L,	
%MFn+74	PV_H	
	<p>Zone tested by checksum every second at the start of processing. If changes are made, the modified parameters are written to the loop parameters.</p> <p><i>The entire zone is updated every second at the end of processing from the loop parameters.</i></p>	
<hr/>		
%MFn+76	<b>Autotuning adjustment zone (6 floating point words)</b>	PLC<->XBT
%MFn+76	AT_STEP,	

---

%MFn+78	AT_TMAX,
%MFn+80	AT_PERF,
%MFn+82	KP_PREV (PID) / KS_PREV (IMC), (read only)
%MFn+84	T1_PREV (PID) / T1_PREV (IMC), (read only)
%MFn+86	TD_PREV (PID) / T_DELAY_PREV (IMC) (read only)

This zone is only managed if the autotuning function exists (PID, IMC).

Zone tested by checksum every second at the start of processing. If changes are made, the modified parameters are written to the loop parameters.

*The entire zone is updated every second at the end of processing from the loop parameters.*

---

%MFn+88	<b>Opening adjustment zone (8 floating point words)</b>	PLC<->XBT
%MFn+88	OUT1_TH1,	
%MFn+90	OUT1_TH2,	
%MFn+92	OUT2_TH1,	
%MFn+82	OUT2_TH2,	
%MFn+84	P_LL,	
%MFn+86	P_HH,	
	DEV_L,	
	DEV_H,	

Zone tested by checksum every second at the start of processing. If changes are made, the modified parameters are written to the loop parameters.

*The entire zone is updated every second at the end of processing from the loop parameters.*

### 6.8-2 Table of periodic data

This table is used by monitoring and TREND pages.

Rank	Parameter	Exchange
%MFn+0	<b>Loop 1 data (6 floating point words)</b>	PLC->XBT
%MFn+0	OUT_MAN,	
%MFn+2	PV,	
%MFn+4	SP,	
%MFn+6	OUT1,	
%MFn+8	OUT2,	
%MFn+10	STATUS	
	The STATUS word is identical to that of the multiplexed table. <i>This zone is updated on all cycles.</i>	
%MFn+12	<b>Loop 2 data (6 floating point words)</b>	PLC->XBT
%MFn+24	<b>Etc, according to the number of loops configured for XBT</b>	PLC->XBT

This table occupies 12 words x (number of loops) configured for XBT, with a maximum of 192 words (%MW) for 16 loops.

Note : The OUT1 and OUT2 fields can be found in the multiplexed table as well as in the periodic table.

If the loop has only one output, the output is in OUT1 and the associated bargraph is magenta.

In the case of a hot-cool, the cool output is stored in OUT2 and the hot output in OUT1. The process control channel variables are therefore reversed. This allows the hot output to be magenta and the cool output blue.

### 6.8-3 Table of alarms (loop only)

This table can be found in the XBT dialog table

Rank	Parameter	Exchange
%MFn+0	<b>Loop 1 alarm word (1 byte)</b>	PLC->XBT
	Each bit defines a various alarm :	
	X0 : STS_SIGMA_ALA (sum of alarms)	
	X1 : STS_HH (PV very high threshold overshoot)	
	X2 : STS_H (PV high threshold overshoot)	
	X3 : STS_L (PV low threshold overshoot)	
	X4 : STS_LL (PV very low threshold overshoot)	
	X5 : STS_DEV_H (positive deviation threshold overshoot)	
	X6 : STS_DEV_L (negative deviation threshold overshoot)	
	<i>This word is updated on all cycles. It is identical to that in the multiplexed zone.</i>	
%MFn+24	<b>Loop 2 alarm word (1 byte)</b>	PLC->XBT
%MFn+24	Etc, according to the number of loops configured for XBT	PLC->XBT

This table occupies one byte per loop configured for XBT, with alignment on an even number. It therefore occupies a maximum of 8 words (%MW) for 16 loops.

## 6.8-4 XBT special table

Rank	Parameter	Exchange
%MWn+0	<p><b>Number of selected loop (1 word)</b>            From 0 to 29. This word is used to select the loop managed by the multiplexed table.            This word is controlled according to word %MWn+5. In the case of an overshoot beyond the last loop or before the first, the word is managed according to the "roller" principle. On initialization it is set to 0.            This word can also be directly written.</p>	PLC<->XBT
%MWn+1	<p><b>Loop 1 monitoring screen status (1 word)</b>            This word is used to display a list of possible states :            X0: 0= the loop does not exist (the integer word is therefore null); 1= the loop exists            X1: 0=loop in manu; 1=loop in automatic            X2: high alarm on PV            X3: low alarm on PV            X4: alarm on deviation              Note : X2 and X3 are mutually exclusive.</p>	PLC->XBT
%MDn+2	<p><b>Loop 2 monitoring screen status (1 word)</b>            Etc. Up to loop 16</p>	PLC->XBT
%MWn+17	<p><b>Loop 1 label (8 bytes)</b>  <i>Updated on initialization</i></p>	PLC->PLC
%MWn+21	<p><b>Loop 2 label (8 bytes)</b>  <i>Updated on initialization</i>            Etc. Up to loop 16</p>	PLC->XBT
%MWn+81	<p><b>Loop 1 unit (6 bytes)</b>  <i>Updated on initialization</i></p>	PLC->XBT
%MWn+84	<p><b>Loop 1 identifier (1 word)</b>            1H : single/process loop : nothing            2H : master cascade : CASCADE M            3H : slave cascade : CASCADE S            4H : autoselector, main loop : AUTOSELECTEUR 0            5H : autoselector, secondary : AUTOSELECTEUR 1  <i>Updated by the selected loop when the loop is selected</i></p>	PLC->XBT
%MFn+85 %MFn+85 %MFn+87	<p><b>Loop 1 scale parameters (2 floating points)</b>            PV_INF            PV_SUP</p>	PLC->XBT
%MFn+89	<p><b>Command word for loop 1 toggle buttons (1 word)</b>            Each bit of the word is used to send a command to the selected loop, on change of state.            X0 : 0= change to local setpoint; 1= change to remote setpoint            X1 : 0= change to manual mode; 1= change to automatic mode</p>	PLC<->XBT





	<p>X2 : 0= stop autotuning; 1= launch autotuning)                  X3 : return to previous setting                  X4 : acknowledge autotuning diagnostics                  X5 : 0= select remote setpoint 1; 1= select remote setpoint 2                  X6 :0= deactivation of output RAISE1; 1= activation of output RAISE1                  X7 : 0= deactivation of output LOWER1; 1= activation of output LOWER1                  X14 : Save parameters  <i>The selected loop takes the command into account on a rising or falling edge. The associated buttons are in toggle mode. The word is updated by the PLC depending on the current state of the loop (for bits used to send 2 distinct commands).</i></p>	
%MWn+90	<p><b>Command word for loop 1 pulse buttons (1 word)</b>                  Each bit of the word is used to send a command to the selected loop. The first 4 bits are associated with dynamic buttons. The following are used for opening :                  X0 : Change setpoint mode (remote-&gt;local or local-&gt;remote depending on current mode)                  X1 : change operating mode (manu-&gt;auto or auto-&gt;manu depending on current operating mode)                  X2 : start or stop autotuning, depending on whether Autotuning is in progress or not                  X3 : return to previous setting                  X4 : acknowledge autotuning diagnostics                  X5 : select remote setpoint 1                  X6 : select remote setpoint 2                  X7 : activation of output RAISE1                  X8 : deactivation of output RAISE1                  X9 : activation of output LOWER1                  X10 : deactivation of output LOWER1                  X14 : Save parameters  <i>The selected loop takes the command into account on a rising edge. The buttons are in pulse mode.</i></p>	PLC<->XBT
%MWn+91 • • •	<p><b>Loop 2 unit (6 bytes)</b>  <b>Loop 2 identifier (1 word)</b>  <b>Loop 2 scale parameters (2 floating points)</b>  <b>Command word for loop 2 toggle buttons (1 word)</b>  <b>Command word for loop 2 pulse buttons (1 word)</b>  <b>Etc. Up to loop 16</b></p>	PLC<->XBT
%MWn+241	<b>Programmer 1 label (8 bytes)</b>	PLC->XBT
%MWn+245	<b>SPP 2 label (8 bytes)</b> <b>Etc. Up to loop 16</b>	PLC->XBT

This table occupies 281 words, regardless of the number of loops and SPPs configured.

---

### 6.8-5 Default addresses

Table	Start address	End address	Max. size (%MW)
Alarm table	%MW3228	%MW3235	8
Programmer multiplexed table	%MW3350	%MW3449	100
Loop periodic table	%MW3500	%MW3691	192
Loop multiplexed table	%MW3700	%MW3803	104
XBT table	%MW3810	%MW4090	281

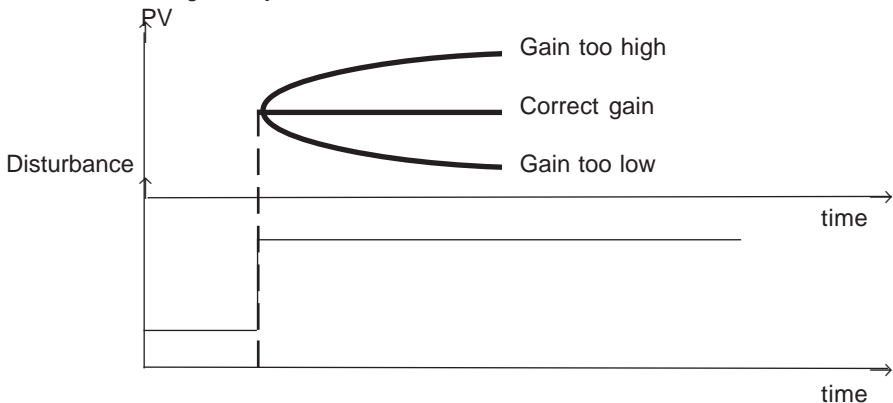
Note : the XBT application dialog table starts at %MW3227.

## 7.1 Debugging the feedforward

To tune the feedforward branch, change the loop controller to manual mode.

### 7.1-1 Adjusting the gain

The gain on the feedforward branch is made with the scale values. It must be adjusted so that the disturbance measured at the feedforward input is completely compensated. To do this, ( $T1\_FF = 0$ ,  $T2\_FF = 0$ ) is required, a disturbance step function must be executed and the gain adjusted in stabilized state.



**Example :** Disturbance variation : 5% -> variation of PV : - 10%  
 and Command variation : 5% -> variation of process value : 7%  
 The selected gain will be :  $(- 10/5) / (7/5) = 1.4$

For a feedforward input of between 0 and 10000 and  $FF\_INF = 0.0$  then  
 $FF\_SUP = - 140.0$  for a command scale ( $OUT\_INF = 0.0$ ,  $OUT\_SUP = 100$ )

### 7.1-2 Adjusting the lead-lag

Initially give  $T1\_FF$  the value of the process time constant, and give  $T2\_FF$  the time constant of the disturbance.

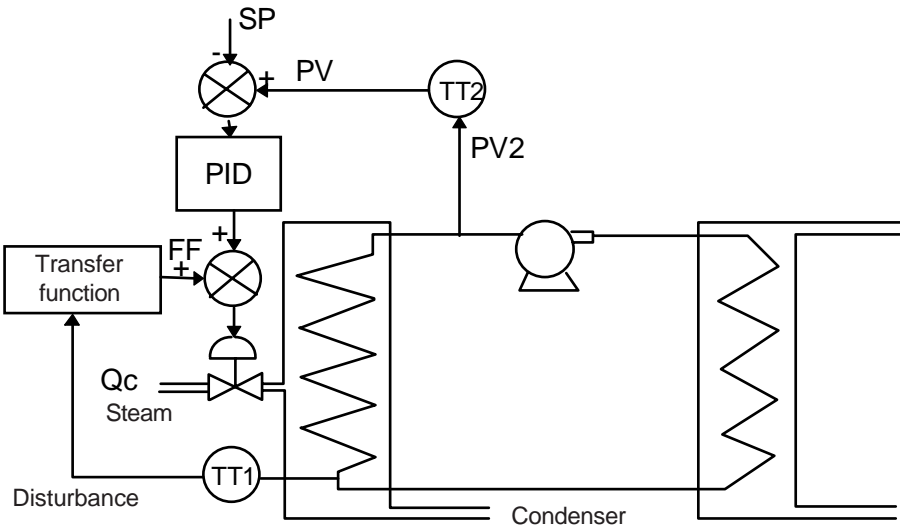
Execute a disturbance step function :

- If the overshoot is positive, decrease  $T1\_FF$ , and if the overshoot is negative, increase  $T1\_FF$ .
- If the overshoot starts positively, increase  $T2\_FF$ , and if the overshoot starts negatively, decrease  $T2\_FF$ .

Example :

C

To adjust the temperature PV2 at the output of the secondary circuit of an exchanger. A PID controller controls the hot air inlet valve according to PV2 and setpoint SP. The cold water temperature is a measurable disturbance in terms of this process control. The use of the Feedforward function enables the system to react as soon as the temperature of the cold water changes, and not once PV2 has decreased.



The following hypotheses are used :

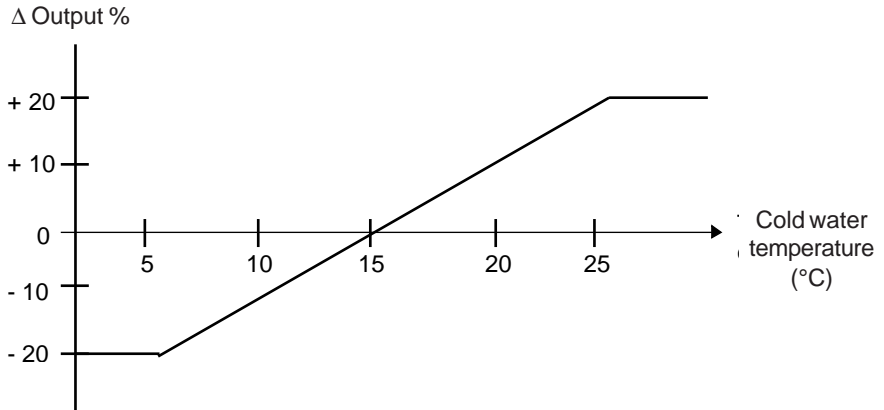
- The temperature at the outlet of the condenser (cold water temperature) varies between 5°C and 25°C, with an average value of 15°C.
- A variation,  $DT$ , of this temperature integrally affects the output temperature of the exchanger.
- To compensate for a 5°C rise (or fall) in temperature of the exchanger output, it is necessary to close (or open) the steam control valve by 10%.

The feedforward input parameters should therefore be adjusted so that the effect of the cold water temperature on the steam flow control valve is :

- zero at 15°C

- In a ratio of 10% / 5°C between 5 and 25°C. This is shown in the following diagram :

The following adjustment should therefore be made :



## 7.2 Debugging the PID function

There are a large number of methods for adjusting the parameters of a PID controller. The one suggested here is the Ziegler and Nichols method, of which there are two versions :

- closed loop adjustment
- open loop adjustment

Before using either of these methods, the direction of action of the PID controller must be determined :

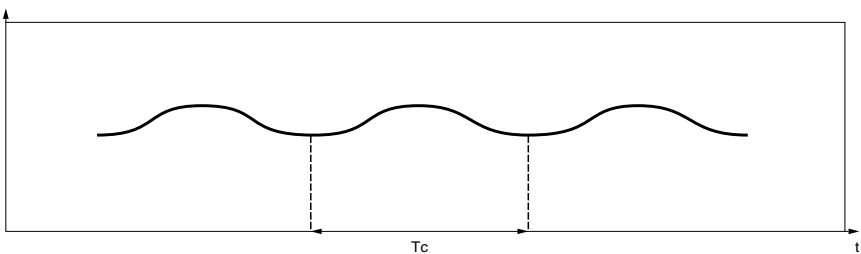
- If an increase of the OUT output causes process value PV to increase, the PID should be set to inverse mode.
- If, on the other hand, this causes PV to decrease, the PID controller should be set to direct mode.

### 7.2-1 Closed loop adjustment

The principle consists of using proportional control ( $T_I = 0$ ,  $T_D = 0$ ) to influence the process by increasing the gain until it starts to oscillate after having applied a step function on the PID controller setpoint.

Simply by reading the value of the critical gain ( $K_{pc}$ ) which caused the undamped oscillation, together with the oscillation period ( $T_c$ ), the values giving optimum adjustment of the loop controller can be deduced.

Process value



Depending on the type of loop controller (PID or PI), the coefficients are adjusted with the values given below :

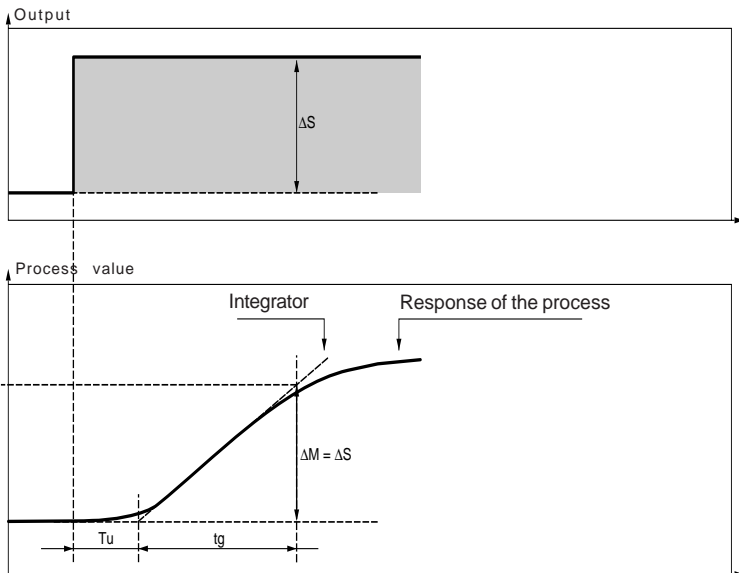
	$K_p$	$T_i$	$T_d$
<b>PID</b>	$\frac{K_{pc}}{1.7}$	$\frac{T_c}{2}$	$\frac{T_c}{8}$
<b>PI</b>	$\frac{K_{pc}}{2.22}$	$0.83 * T_c$	X

where  $K_p$  = proportional gain,  $T_i$  = integration time and  $T_d$  = derivative time.

This method of adjustment provides very dynamic control which may cause undesirable overshoots when setpoints are changed. In this case, lower the value of the gain until the required behavior is achieved.

### 7.2-2 Open loop adjustment

As the loop controller is in manual mode, a step function is applied on the output and the start of the process response is similar to a pure delay integrator.



The point at which the straight line representing the integrator intersects the time axis determines the time  $T_u$ .

The time  $T_g$  is then defined as the time required for the controlled variable (process value) to vary by the same amplitude (in % of scale) as the loop controller output.

Depending on the type of loop controller (PID or PI), the coefficients are adjusted with the values opposite.

	$K_p$	$T_i$	$T_d$
PID	$1.2 T_g/T_u$	$2 T_u$	$0.5 T_u$
PI	$0.9 T_g/T_u$	$3.3 T_u$	X

This method of adjustment again provides very dynamic control which may cause undesirable overshoots when setpoints are changed. In this case, lower the value of the gain until the required behavior is achieved.

The advantage of this method lies in the fact that it does not require any hypothesis on the type and order of the process. It can be applied just as easily to stable processes as to genuinely integrating processes. It is particularly useful for slow processes as the user only needs the start of the response to adjust the coefficients  $K_p$ ,  $T_i$  and  $T_d$ .

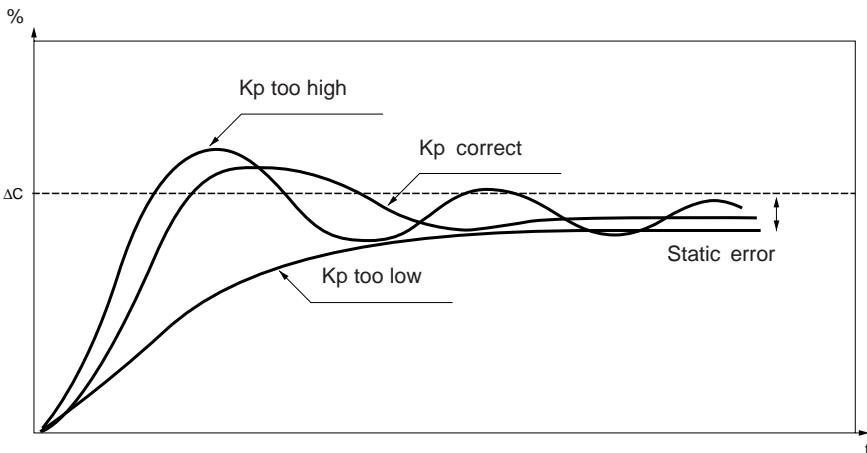
### 7.2-3 Roles and effects of the parameters of a PID controller

#### Proportional action

Proportional action is used to alter the speed of response of the process. The higher the gain, the faster the response and the lower the static error (with pure proportional action), but the poorer the stability.

A good compromise must be found between speed and stability.

#### Effect of proportional action on the response of the process to a step function



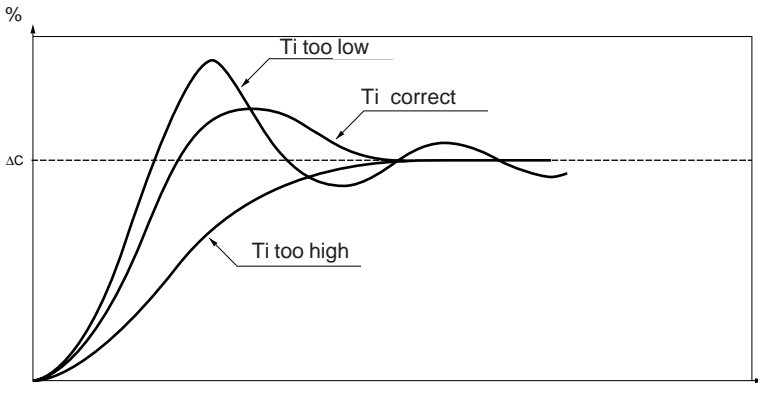


### Integral action

Integral action is used to remove the static error (deviation between the process value and the setpoint). The higher the integral action ( $T_i$  low), the faster the response and the poorer the stability.

A good compromise must be found between speed and stability.

#### Effect of integral action on the response of the process to a step function

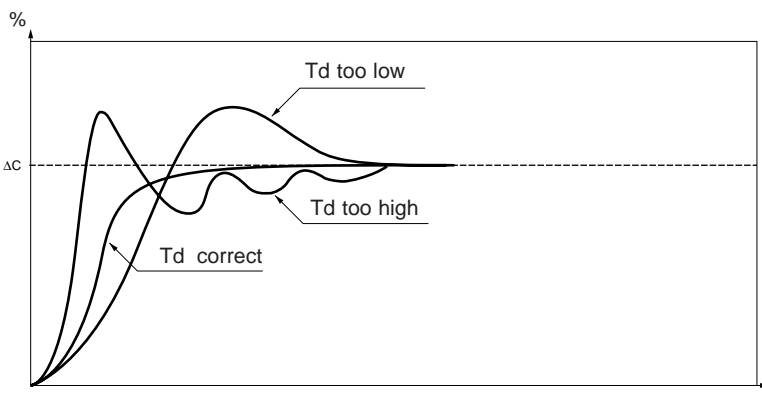


Reminder : Low  $T_i$  means a high integral action.

### Derivative action

Derivative action is anticipative. In fact, it adds a term which takes account of the speed of variation of the deviation, which makes anticipation possible by accelerating the response of the process when the deviation increases, and slowing it down when the deviation decreases. The higher the derivative action ( $T_d$  high), the faster the response. Here too, a good compromise must be found between speed and stability.

#### Effect of derivative action on the response of the process to a step function



---

## Limits of PID control

If the process is compared with a first order with pure delay, with transfer function :

$$H(p) = \frac{Ke^{-\tau p}}{1 + \theta p}$$

where :

- $\tau$  = delay of the model
- $\theta$  = time constant of the model

the performance of the process control depends on the  $\tau / \theta$  ratio.

PID control is suitable in the following area :

$$2 \leq \frac{\theta}{\tau} \leq 20$$

For  $\theta / \tau < 2$ , ie. fast loops ( $\theta$  low) or processes with a high delay ( $\tau$  high) PID control is no longer suitable, and more sophisticated algorithms must be used : IMC model-based controller.

For  $\theta / \tau > 20$ , threshold control with hysteresis is adequate (OnOFF controllers).

### 7.3 Debugging the model-based controller

The model of the process must first be identified. A graphic method based on the indexed response of the process can be used (for example, the Broïda method which directly provides the parameters of a first order model with pure delay). Once the parameters of the model are known, the adjustment can be refined by changing the RM to automatic.

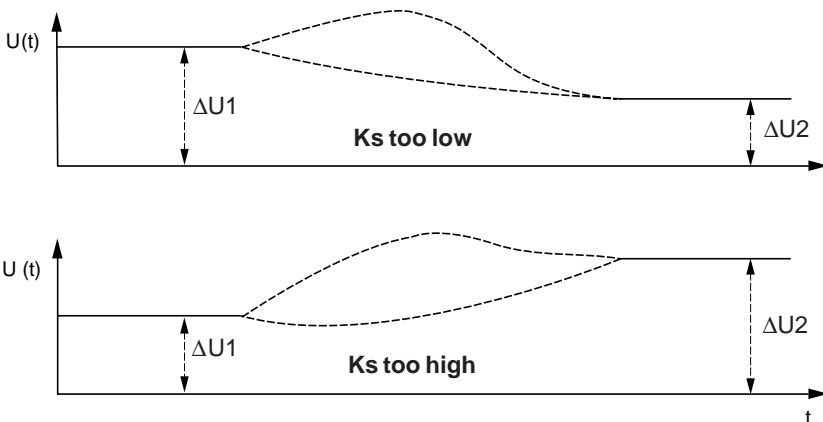
To check whether the model is suitable for the process, set  $CL\_PERF = 1.0$  (closed loop time constant = open loop time constant).

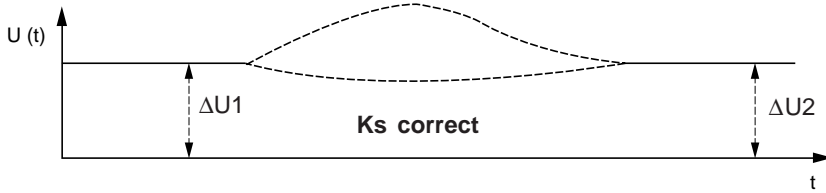
Take the process to the operating point and set the loop controller to automatic, and execute a setpoint step function  $\Delta C$ . Under these conditions, if the model parameters are correct, the process value should be the same as the setpoint with no overshoot, and the  $OUT\_MAN$  control signal must be practically a step function. If this result is not obtained, it is because the model parameters are not suitable for the process. A correction must then be made, that is, the static gain, the dead time and the time constant must be adapted.

#### 7.3-1 Adjusting the static gain (Ks)

During the setpoint step function, observe the recording of the control signal  $U(t)$ . If the static gain is correct, the variation amplitude  $\Delta U1$  should be equal to  $\Delta U2$ . If this is not the case, correct the static gain by applying the formula :

$$\text{Correct } K_s = \text{Test } K_s \cdot \Delta U1 / \Delta U2$$

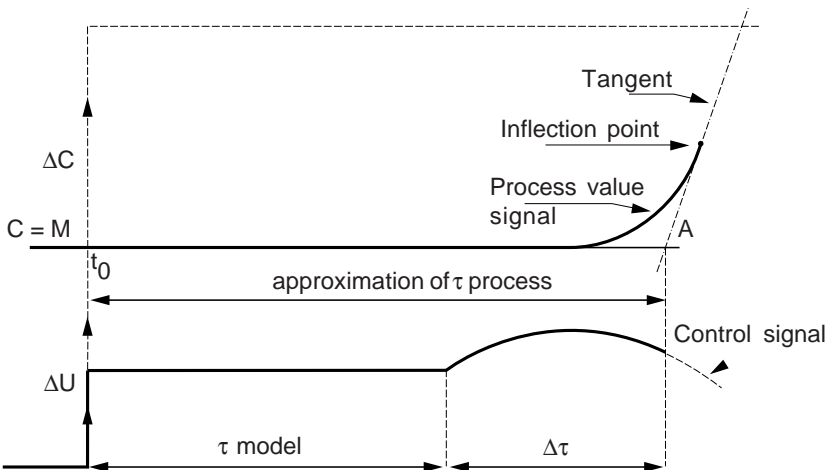




### 7.3-2 Adjusting the dead time or delay ( $T\_DELAY$ )

To perform this adjustment, observe the process value signals and the control signals of the adjustment device during a recording. Let us call  $\tau$  the delay of the model or the process. Only the start of recordings of the signals can be used :

- If  $\tau$  model is less than  $\tau$  process

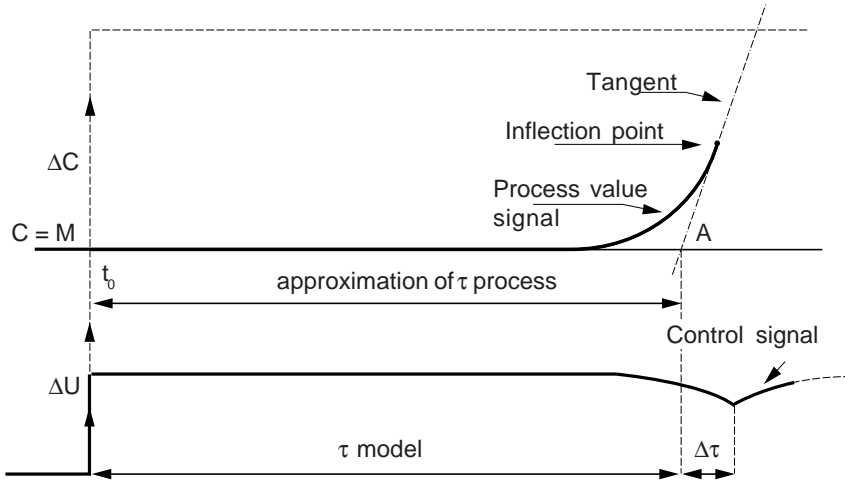


The control signal "starts" with a higher value in relation to  $\Delta U$  on a positive setpoint step function  $\Delta C$ .

Trace the tangent to the first inflection point on the process value signal, which intersects the time axis at point A. Value A is very similar to the process delay time.

$T\_DELAY$  then takes this value A

- $\tau$  model is greater than  $\tau$  process



The control signal "starts" with a lower value in relation to  $\Delta U$  on a positive setpoint step function  $\Delta C$ .

Trace the tangent to the first inflection point on the process value signal and read the value of  $\Delta\tau$ .

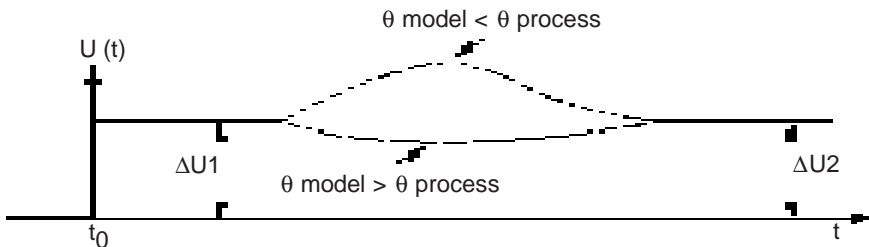
$T\_DELAY$  then takes the value at point A

**Note**

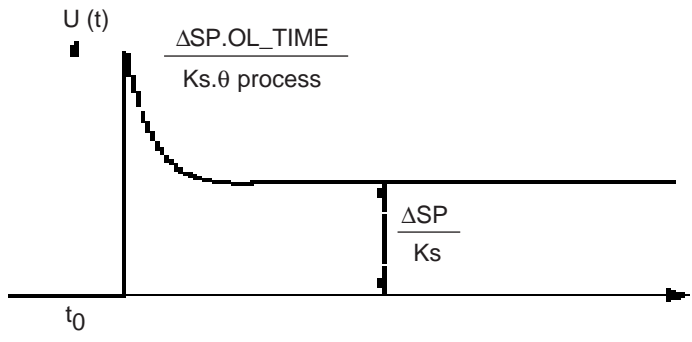
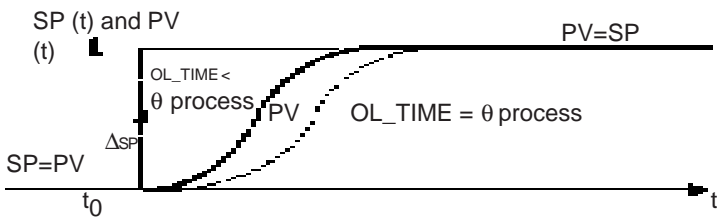
The delay and the static gain can both be adjusted in the same test.

**7.3-3 Adjusting the time constant**

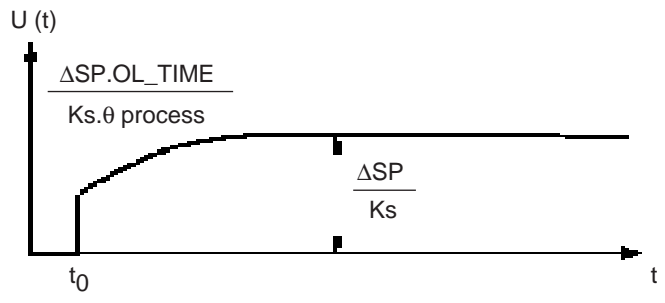
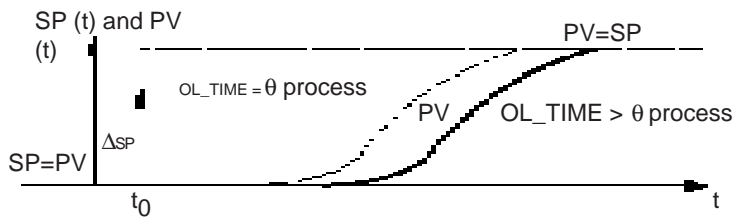
After adjusting the dead time and the static gain, adjust the time constant of the model by successive approaches, observing the recording of the control signal  $U(t)$ . Let us call  $\theta$  the time constant.



• speed of the signals when  $OL\_TIME < \theta$  process



• speed of the signals when  $OL\_TIME > \theta$  process



---

After determining the model to be used, the closed loop time constant still needs to be chosen. Its value depends on the response speed of the required closed loop.

For processes suitable for a first order model with delay, by selecting a ratio of time constants CL\_PERF between 1.05 and 1.15, the system response is improved without the risk of destabilizing the process ( $CL\_PERF = OL\_TIME / \text{required closed loop time constant}$ ).

Any increase in CL\_PERF corresponds to an increase in the response speed (at the cost of a more significant action of the adjustment device), and also to an increased sensitivity to modeling errors.





### Symboles

1st order filtering 2/6, 2/32  
 3 single loops 1/3

### A

Alarm on deviation 2/56  
 Alarm on level 2/37  
 Algorithms 1/14  
 Auto 2/67  
 Autoselective loop 1/1  
 Autotune 1/6

### B

Branches 2/1

### C

Cascaded loop 1/1  
 Channel 2/3  
 Closed loop 7/4  
 Compatibility 1/14  
 Configuration 1/5

### D

Dead band 2/69  
 Debug 1/5  
 Delay 2/72  
 Direct action 2/67  
 Dwell step 2/21

### E

Exchange zones 4/1  
 Execution check 2/8  
 External process value 2/7

### F

FeedForward 2/4, 7/1  
 Front panel 1/8  
 Function generator 2/6, 2/34  
 Functions 2/1

### G

Gradient 2/73  
 Guaranteed dwell time 1/11, 2/22

### H

High scale 2/29  
 Hot/Cool 2/6

### I

ID 2/29  
 IMC 2/14  
 Integral band 2/69  
 Inverse action 2/67

### L

Lead-lag 2/6  
 Local 4/11  
 Local setpoint 2/9  
 Low scale 2/29

### M

Magelis 1/8  
 Main loop 2/2  
 Manu 1/6  
 Master loop 2/2  
 Mixed 2/64  
 Mixed PID 2/66  
 Model-based controller 1/3, 2/72  
 Multiplexed table 4/2

### O

OnOff 2 2/6  
 OnOff 3 2/6  
 Open loop 7/5

### P

Parallel 2/64  
 Parallel PID 2/65  
 Performance 1/14  
 Periodic table 4/2  
 PID 2/6  
 PL7-PRO-DYN 1/7  
 Position feedback 2/18, 2/94  
 Post-processing 5/1  
 Pre-processing 5/1  
 Process loop 1/1  
 Profile 2/3  
 Pulsed output 2/6  
 PWM 2/17

**R**

Ramp	2/21
Ratio	2/6, 2/42
Remote	4/11
Remote setpoint	2/9
Runtime screens	1/7

**S**

Sampling period	2/3
Saving	3/5
Saving data	3/5
Scale	2/45
Scale limiter	2/36
Scaling	2/6
Secondary	1/1, 2/2
Secondary loop	2/2
Segments	1/11
Selection	2/6, 2/44
Sending commands	1/6
Servo	2/6
Servomotor	2/6, 2/17
Setpoint limiter	2/47
Setpoint programmer	1/1
Simple setpoint	2/9
Simulation	1/6
Single process control loop	1/1
Slave loop	2/2
Speed limiter	2/6, 2/50
Split range	2/6
Square root	2/6, 2/33
Standard process value	2/7
Symbolization	2/104
Symbolize	2/1
Synchronization	5/1

**T**

Threshold limiter	2/6
Totalizing	2/39
Tracking	2/73
Tracking setpoint	2/6, 2/49
Tuning	1/5

**W**

Warnings	2/15
----------	------

**X**

XBT	1/2
XBTF	1/2
XBTF	1/8
XBTF01	4/5
XBTF02	4/5
XBTL1000	1/2

<b>Section</b>	<b>Page</b>
<b>1 PL7 programming</b>	<b>1/1</b>
1.1 Module operation	1/1
1.2 Module configuration	1/3
1.2-1 Configuration parameters	1/4
1.2-2 Default configuration	1/12
1.3 Programming the weighing function	1/13
1.3-1 Language objects associated with the weighing function	1/13
1.3-2 Programming aspects associated with the weighing function	1/15
<b>2 Debugging</b>	<b>2/1</b>
2.1 Introduction	2/1
2.2 Sending commands to the weighing module	2/5
2.2-1 Calibration	2/7
2.2-2 Saving adjustments in the processor	2/12
2.2-3 Tare	2/13
2.2-4 Zero reset	2/15
2.2-5 Order to return to gross weight	2/17
2.2-6 Order to display the manual tare for 3 seconds	2/18
2.2-7 Orders to enable and disable thresholds	2/19
2.3 Adjustments	2/21
2.3-1 PL7 instructions used for adjustment	2/22
2.3-2 Adjustment parameters	2/24
2.3-3 Adjustment procedures	2/26
2.3-4 Reading configuration parameters	2/27

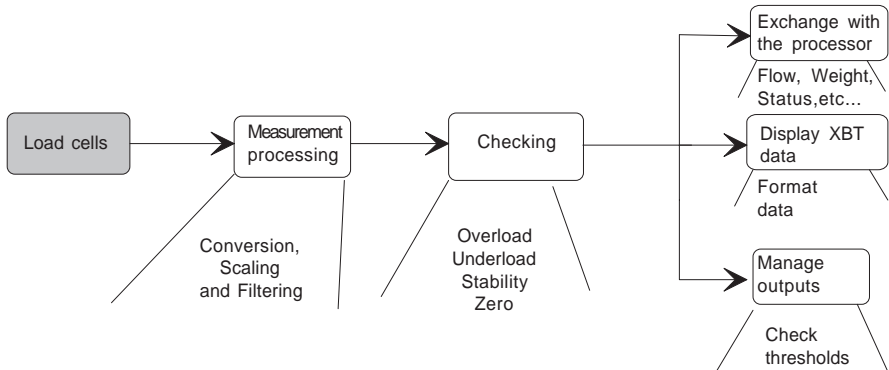
<b>Section</b>	<b>Page</b>
<b>3 Operation</b>	<b>3/1</b>
3.1 Weighing operation using PL7	3/1
3.2 The display report	3/4
3.3 Operating mode	3/5
<b>4 Adjustment protection</b>	<b>4/1</b>
4.1 Adjustment protection procedure	4/1
4.2 Metrology legal requirements and regulations	4/3
4.2-1 CE type approval	4/3
4.2-2 Approval for national model	4/3
4.2-3 Class of apparatus	4/4
<b>5 Programming examples</b>	<b>5/1</b>
5.1 Example of tare operation	5/1
5.2 Example of batching	5/2
<b>6 Appendix</b>	<b>6/1</b>
6.1 Technical characteristics	6/1
6.2 Standards	6/2
6.3 Approvals	6/2
6.4 Recommendations for installing an analog measurement system	6/3
<b>7 Index</b>	<b>7/1</b>

## 1.1 Module operation

To operate the **weighing module**, it must first be **configured**.

In a PLC environment, the module, like the other modules, has a range of data unique to it. This data is used for exchanges (reports and commands) with the processor.

The following operating diagram shows the processing executed by the module and indicates all the elements which require configuration.



### Measurement processing

The signal from the load cells is :

- converted,
- the measurement is filtered in accordance with the selection made on the parameter settings screen,
- scaled, with scale characteristics determined following calibration.

### Measurement verification

The measurement resulting from processing is subjected to the following verifications :

- overload check,
- underload check,
- stability check, defined via a stability range and a stability period,
- check for presence of zero in the zone.

---

### **Data exchanges with the processor**

The module receives and processes the commands from the processor (zero reset, semi-automatic tare, etc).

It also prepares data in "legal" format (ie. as required by law) for display on the TSX XBT H100.

It returns various data to the processor such as gross weight, net weight, flow rate, tare and status.

### **Data display**

The TSX XBT H100 displays the weight or the manual tare in the unit selected during configuration with 4 additional data items : net weight, stability, presence of zero in the zone and the weight unit.

### **Output management**

The module can manage 2 discrete outputs directly and control them according to the thresholds transmitted to the module by the application program.

The elements used in management are :

- the switching thresholds,
- the direction of weight change (Weighing or Downweighing),
- the output switching logic.

## 1.2 Module configuration

The configuration data is used to define metrological characteristics and adapt the module operation to its intended application.

The weighing module configuration screen appears below (for more information on screen access refer to the PL7 Junior software documentation).

The screenshot shows the configuration interface for the TSX ISP Y100 module. The window title is "TSX ISP Y100 [RACK 0 POSITION 2]". The "Configuration" dropdown is selected. The "Designation" is "1 WEIGHT MEASURE INPUT".

**Symbol:**  
 Channel: 0 | Function: Weighing | Task: MAST

**Metrological Data**  
 Unit: kg | Max Range (MR): 150.00 kg | Scale Division (d): 0.01 kg | Overload Threshold: +9d

**Zero**  
 Extent of Range: ±2xMR |  Zero Tracking

**Data Format**  
 Legal |  High Resolution

**Stability**  
 Extent of Range: 3 % d | Time: 1 s

**Sensor Supply**  
 Continuous |  Switched

**Filtering**  
 Coefficient: 4

**Flow**  
 Calculate on: 4 Measures

**Tare**  
 Predefined | Value: 0.01 kg

**Threshold Check**  
 Active | Direction:  Weighing |  Downweighing  
 Outputs Active Phase 1:  S0 |  S0 and S1  
 Cut-off Points: Low Flow (LF): 0.0000 kg | High Flow (HF): 0.0000 kg  
 LF Mask Time: 0 s

The weighing module has 2 categories of parameters :

- user-defined parameters which are generally modified during setup and then locked,
- operating parameters which are always accessible.

---

## 1.2-1 Configuration parameters

The configuration screen contains the following data associated with the weigher and its processing :

### Task

The user may choose between :

- Mast task
- Fast task

### Note:

For more details, refer to the TLX DS PL7J 10E documentation.

### Metrological data

Metrological Data	
Unit :	kg ▾
Max Range (MR):	150 .00 kg
Scale Division (d):	0.01 ▾ kg
Overload Threshold :	.9d ▾

- **Unit :**

From a predefined list, the user may choose grams, kilograms, tonnes, pounds (1 lb = 453g), ounces (1 oz = 28.35g) or no unit.

- **Maximum range (MR) :**

This is the maximum load which can be weighed using the instrument, without taking into account the weight of the empty load receptacle (in legal format, see: 'Data format' parameter).

- **Scale division (sd) :**

The scale division value is in the form  $1, 2$  or  $5$  multiplied by  $10^n$  ( $n$  being a positive or negative integer or zero with  $|n| \leq 3$ ).

### Note:

In industrial use, if because of the weigher installation environment, a resolution of greater than 3000 points is selected, installation precautions for operation in harsh environments must be taken.

It will not be possible to enter a resolution of greater than 50 000 points on the programming screen.

In other words, the following inequality must be observed :

Maximum range (MR) < 50 000 x Scale division.



- **Overload threshold**

This threshold is the value above which the display unit can no longer display the weight (the overload is indicated by '>' on the display unit).

It may have the following values : +9 scale divisions or +2% of the maximum range or +5% of the maximum range.

Example :

The maximum range is set at 150 Kg and the scale division at 10 g : depending on user choice the operating limit will be :

Overload threshold	Corresponding limit
9 scale divisions	Max. range + 9 sd or 150.09 Kg
+2%	102% of max. range or 153 Kg
+5%	105% of max. range or 157.5 Kg

**Note :**

Parameters cannot be set for the overload threshold. It defines the tolerated limit for the indication below zero. It is -2% of the maximum range (underload is then indicated by a line of '<' on the display unit).

**Zero**

**Zero**

**Extent of Range:**    ±2%MF ▾

**Zero Tracking**

- **Extent of the range :**

Any offset from zero may be corrected in the measurement if this range is not exceeded.

It is defined as a % of the maximum range. It may take the following values : ±2%, ±5%.

- **Zero tracking** (optional) : this function is used to compensate slow drifts from zero within the extent of the range (2% of the maximum range). This option is not recommended for automatic installations.

**Note :**

A slow drift is distinguished from a genuine weighing operation as follows :

Any weight variation with an amplitude of less than one-half of a scale division where the frequency of repetition is low enough to maintain measurement stability is considered to be a drift.

Any correction made by the function is limited to ±2% of the maximum range of the weigher. Once this limit is exceeded, automatic correction no longer functions.

## Data format

**Data Format**

**Legal**

**High Resolution**

The weight value displayed or entered by the user

- either as a physical unit with a fixed decimal point : **legal format**
- or as one-hundredth of a physical unit with a fixed decimal point : **high resolution**

### Comment :

A **physical unit with a fixed decimal point** is an integer expressed as a unit of weight for which a decimal point is required.  
The position of the decimal point is given by the power to the tenth of the scale division.

### Example :

Legal format :

The value 3014 signifies 301.4 kg if the scale division is  $2 \cdot 10^{-1}$  kg.

High resolution format :

The value 301403 signifies 301.403 kg if the scale division is  $2 \cdot 10^{-1}$  kg.

This unit offers greater precision but is not accepted by the French Metrology Department.

## Stability

**Stability**

**Extent of Range:**  % d

**Time:**  s

- **Extent of the range:**

A weight may not be measured immediately after the load is received because of the inevitable oscillations affecting the mechanical parts.

The stability range represents the amplitude below which the measurement is considered to be stable.

It can be set at 2, 3, 4, 6 or 8 quarters of a scale division.

- **Time :**

The stability time represents the period during which the measurement must remain within the stability range to be considered stable. It can be set at 0.4, 0.5, 0.7 or 1 second.

---

## Load cell power supplies

**Sensor Supply**

Continuous

Switched

These parameters determine the load cell power supplied by the module at 10 volts (continuous or switched current).

### Note :

A switched power supply has the advantage of canceling out any offset voltages within the measurement system, especially those due to thermocouple effects. This option is not implemented on the module version.

## Filtering

**Filtering**

**Coefficient**

Filtering is on the measurement input of the load cells.

Two types of filter are available :

- sliding average filters (from 1 to 11) where the measurement is an average of the last n values,
- second order filters (from 12 to 19) referenced by their cut-off frequencies.

---

The user selects the filter value from the list below :

Value	Type of filtering	Characteristics
0	none	unfiltered
1	sliding average	averaged over last 2 measurements
2	sliding average	averaged over last 3 measurements
3	sliding average	averaged over last 4 measurements
4	sliding average	averaged over last 5 measurements
5	sliding average	averaged over last 8 measurements
6	sliding average	averaged over last 16 measurements
7	sliding average	averaged over last 25 measurements
8	sliding average	averaged over last 32 measurements
9	sliding average	averaged over last 40 measurements
10	sliding average	averaged over last 50 measurements
11	sliding average	averaged over last 64 measurements
12	second order filter	cut-off frequency 15 Hz
13	second order filter	cut-off frequency 10 Hz
14	second order filter	cut-off frequency 8 Hz
15	second order filter	cut-off frequency 6 Hz
16	second order filter	cut-off frequency 4 Hz
17	second order filter	cut-off frequency 2 Hz
18	second order filter	cut-off frequency 1 Hz
19	second order filter	cut-off frequency 0.8 Hz

## Flow rate

The image shows a control panel for flow rate measurement. It features a label 'Flow' above a 'Calculate on' field containing the number '4' and a dropdown arrow. To the right is a 'Measures' label.

The user may select the number of measurements (one measurement is performed every 20 milliseconds) to calculate the flow rate.

The flow rate is calculated in accordance with the following formula :

$$\text{Flow rate}_n = (\text{Val}_n - \text{Val}_{n-\beta})$$

This is the difference in weight for a number of configured measurements.

With  $\beta$  being the number of measurements for calculating the flow rate,  $\text{Val}_n$  the unfiltered weight at instant  $n$  and  $\text{Val}_{n-\beta}$  the unfiltered weight at instant  $n-\beta$ .

Operation :

At any given moment the flow rate is calculated and is implicitly transmitted to the processor as the weight measurement to be used in correcting thresholds. The flow rate is always calculated in high resolution format. This calculation can be made over 2, 4, 8, 16, 32 or 64 measurements.

The default number of measurements is 4.

## Tare

The image shows a control panel for tare configuration. It has a 'Tare' label above a checkbox labeled 'Predefined' which is checked. Below the checkbox is a 'Value:' field containing '0.01' followed by the unit 'kg'.

The tare is the weight measurement memorized on the last semi-automatic tare command. However, the user may, if necessary, manually introduce a tare value. This tare value is referred to as “predefined” or “manual” and may be transmitted to the module. It is expressed in legal format (physical unit with a fixed decimal point).

The tare must of necessity be positive or zero and less than the Maximum Range.

When such a device is used, the “predefined” tare indicator (PT) is set. It is disabled when a Tare order is executed.

### Note:

The entry range extends from 0 to 65 535 : if the user requires a larger tare, he must modify the scale division and then enter the tare.

### Threshold check (optional)

The threshold check manages the discrete outputs of the module. The High Flow cut-off point is associated with output Q0 : the Low Flow cut-off point is associated with output Q1.

Threshold Check			
<input type="checkbox"/> <b>Active</b>	Direction:	<input checked="" type="radio"/> Weighing	<input type="radio"/> Downweighing
Outputs Active Phase 1:		<input checked="" type="radio"/> S0	<input type="radio"/> S0 and S1
Cut-off Points:	Low Flow (LF)	<input type="text" value="0.0000"/>	kg
	High Flow (HF)	<input type="text" value="0.0000"/>	kg
LF Mask Time:		<input type="text" value="0"/>	s

- **Active**

Discrete output management is operating if this box is checked.  
By default, it is not checked.

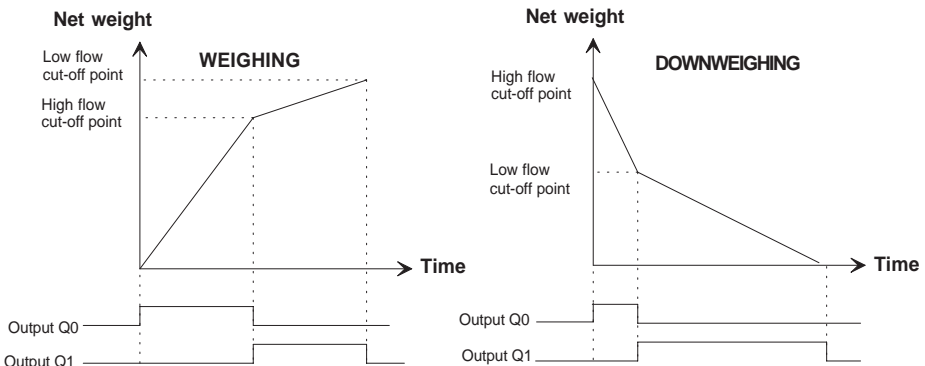
- **Direction**

The direction of detection corresponds to the direction in which thresholds are taken into account, either in **Weighing** (filling) or **Downweighing** (emptying).  
The theory is that of exceeding a greater value when weighing or a lesser value when downweighing.  
Weighing is selected by default.

- **Active outputs phase 1**

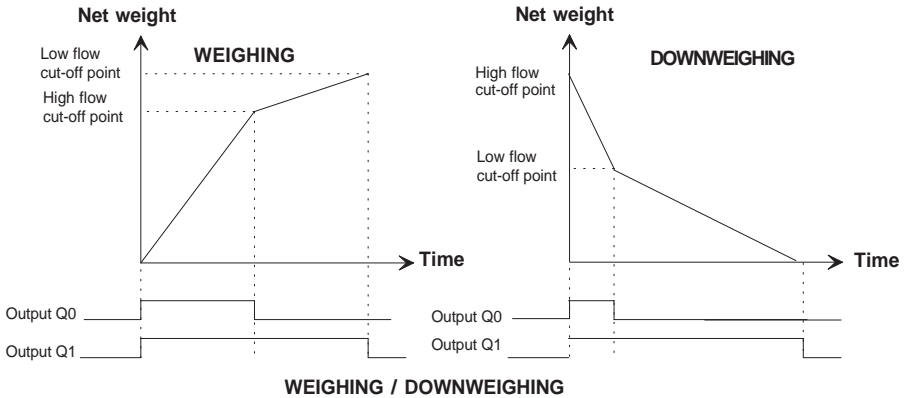
The choice is between Q0 only or Q0 and Q1 simultaneously. By default, the module activates Q0 alone in the first phase.

Active output phase 1 (Q0):



WEIGHING / DOWNWEIGHING

Active output phase 1 (Q0 and Q1):



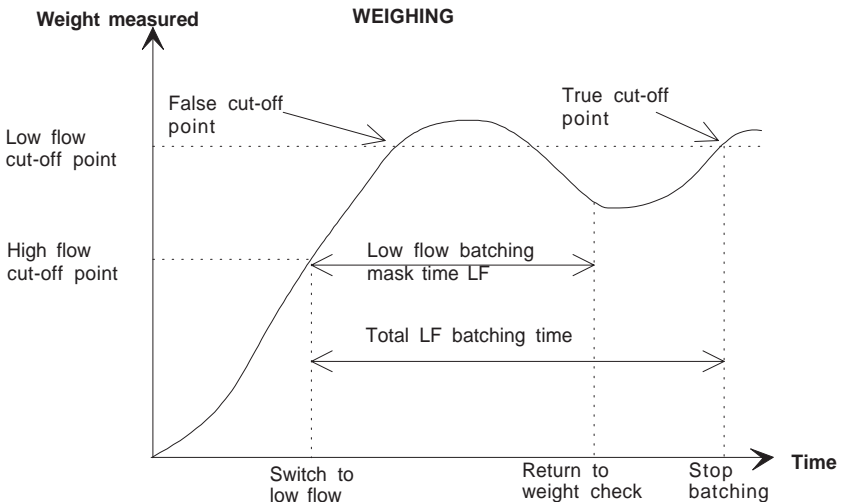
### Cut-off points

Measurement may be associated with two thresholds for batching : one High Flow cut-off point and one Low Flow cut-off point. Depending on the logic defined, outputs Q0 and Q1 reset when these thresholds are exceeded.

The permitted threshold values are between 0 and the maximum range. They are expressed in high resolution (one-hundredth of a physical unit with a fixed decimal point).

### LF (low flow) mask time

This defines the time after the high flow cut-off point during which the module is no longer checking Weight/Thresholds. This is to mask the overshoot due to the fall of the product. The permitted values are between 0 and 1.5 seconds in steps of 1/10<sup>th</sup> of a second. The default time is zero.



## 1.2-2 Default configuration

The table below shows the default configuration for the module :

Parameters	Default configuration	Possible	Unit
Task	Mast	Mast Fast	
Data format	Legal format	Legal format High resolution	
Metrology / Unit	kg	kg g t lb oz no unit	kilogram gram tonne pound (= 453g) ounce (= 28.35g)
Metrology / Maximum range	150	from 0 to 65 535	in the unit of weight selected
Metrology / Scale division	$1 \cdot 10^{-2}$	1, 2 or $5 \cdot 10^n$	in the unit of weight selected
Metrology / Overload threshold	+9 scale divisions	+9 scale divisions +2% +5%	scale divisions % of Max. Range % of Max. Range
Filtering / Coefficient	4	from 0 to 19	
Flow rate / Calculate on	4	2, 4, 8, 16, 32 or 64	measurements
Tare	Not predefined	Predefined or not	in the unit of weight selected
Stability / Extent of the range	3	2, 3, 4, 6 or 8	1/4 scale division
Stability / Time	1	0.4, 0.5, 0.7 or 1	second
Zero / Zero tracking	Inactive	Inactive or active	
Zero / Extent of the range	2% MR	$\pm 2\%$ MR, $\pm 5\%$ MR	
Cell power supply	Continuous	Continuous	
Threshold check	Inactive	Inactive or Active	
Output logic	Weighing	Weighing or Downweighing	
Active outputs	Q0	Q0 or (Q0 and Q1)	
Cut-off point	0	from 0 to Max. Range	in $1/100^{\text{th}}$ of the unit selected
Mask time	0	0 to 1.5 seconds in steps of 0.1 s.	in seconds



### 1.3 Programming the weighing function

Accessing the weighing function via the control system, as with other modules, is via objects (bits, words, etc, associated with the module).

#### 1.3-1 Language objects associated with the weighing function

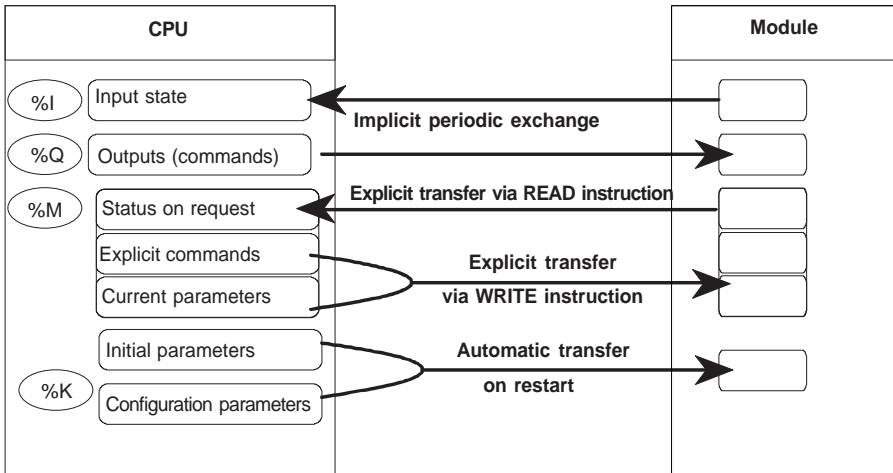
The configuration of a weighing module in a given position, generates a set of language objects which can be manipulated via the application program.

The syntax of these objects is structured in the form:

%	I, Q, M, K	X, W, D	xy	• i	• r
<b>Sign</b> IEC 1131	<b>Type of object</b> I = input Q = output M = internal word K = constant word	<b>Format</b> X = boolean W = word D = double word	<b>Position</b> x = rack number y = position number in rack	<b>Channel no.</b> i = 0 to 127 or MOD	<b>Rank</b> r = 0 to 255 or ERR

The data exchange principle is as follows :

Certain data is exchanged implicitly for each channel on each task scan : others are exchanged explicitly (specific exchange instruction). The model below summarizes the exchanges between the module and the processor:



## Implicit exchange objects

This data, exchanged at the end of each PLC scan, mainly concerns measurement data. The table below lists the names and significance of this data.

Object address	Object contents
%Ixy.MOD.ERR	Module error bit
%IDxy.0.0	Weight (GROSS or NET)
%IDxy.0.2	Flow rate
%IWxy.0.4	Data on the measured value
%IDxy.0.5	Tare value
%IDxy.0.7	Offset memory (zero offset)
%Ixy.0.ERR	Measurement channel error bit

## Explicit exchange objects

This data is updated via command and adjustment functions.

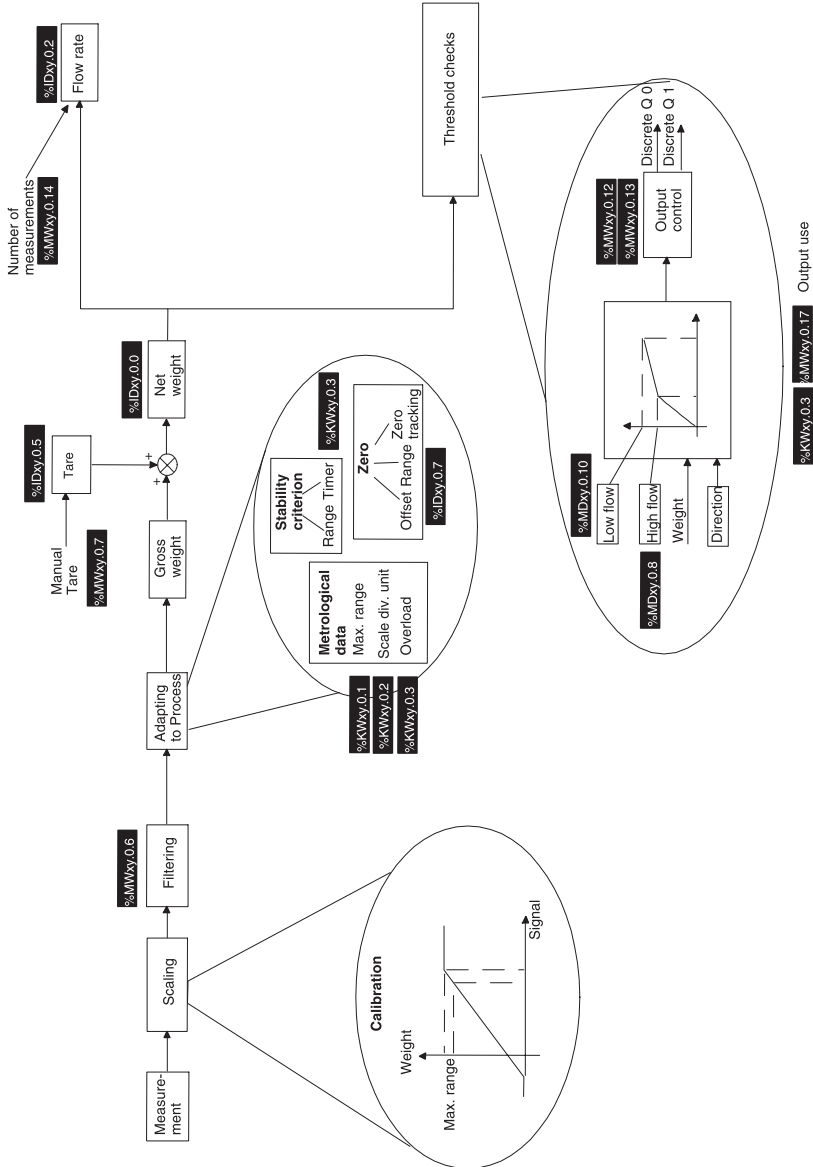
Object address	Object contents
%MWxy.MOD.2	Module status
%CHxy.0	Structure of data attached to the channel
%MWxy.0.0	Exchange in progress
%MWxy.0.1	Exchange report
%MWxy.0.2	Channel status
%MWxy.0.3	Command order (calibration, tare, zero reset, etc)
%MDxy.0.4	Control parameters
%MWxy.0.6	Filtering coefficient
%MWxy.0.7	Manual tare value
%MDxy.0.8	High flow cut-off point Q0 (batching)
%MDxy.0.10	Low flow cut-off point Q1 (batching)
%MWxy.0.12	Logic of outputs Q0 and Q1
%MWxy.0.13	LF mask time
%MWxy.0.14	Number of measurements to calculate flow rate

## Constants

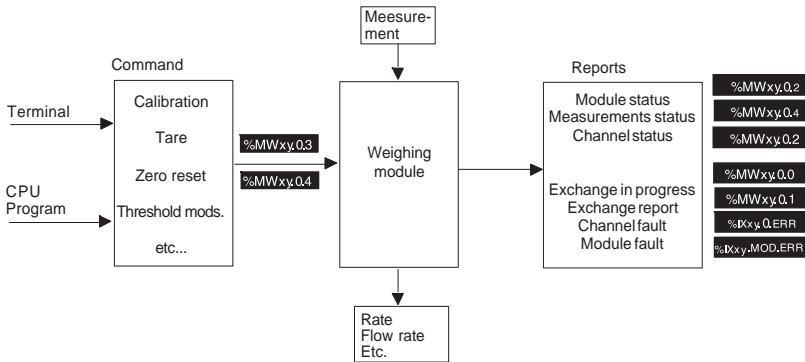
Object address	Object contents
%KWxy.0.0	Maximum range (configuration parameter)
%KWxy.0.1	Unit + scale division
%KWxy.0.2	Stability + Zero + Overload + Output use + Format

### 1.3-2 Programming aspects associated with the weighing function

The flowchart shows the sequencing of the various functions executed by the module.



Using the PLC program or directly via the terminal, the user may send commands to the weighing module :



## Accessing measurements

The numeric values, weight measurement (GROSS or NET) and flow rate, are placed in **2 double word input registers (%ID)**. They are complemented by 1 measurement status word (%IW), 1 tare value double word (%ID) and 1 offset memory double word (%ID) (zero offset).

Register address	Register significance
%IDxy.0.0	Weight value (GROSS or NET)
%IDxy.0.2	Flow rate
%IWxy.0.4	Measurement status : stability, zero, etc
%IDxy.0.5	Tare value
%IDxy.0.7	Offset memory (zero offset)

This data is automatically returned to the processing unit at the start of the task associated with the channel, whether the **task** is in **Run** or **Stop**.

The data is directly accessible :

- from the application via operator dialogue (access to the PLC memory image objects),
- from the terminal using the animation tables.

The **weight** (see section 3 3.2, Module Configuration)

Example :

Legal format : %IDxy.0.0 = 3014 signifies (if the scale division is  $2 \cdot 10^{-1}$  kg) that the weight is 301.4 kg.

High resolution : %IDxy.0.0 = 301403 signifies (if the scale division is  $2 \cdot 10^{-1}$  kg) that the weight is 301.403 kg.

By default, if no tare order has been executed, the weight is expressed as a GROSS weight. It becomes a NET weight when a tare order is executed or when a tare is introduced manually.

**The flow rate** (see chapter 3 section 3.2, Module configuration)

Example :

$%IDxy.0.2 = 450\ 000$  signifies, if the scale division is  $1.10^{-2}$  kg, that a weight difference of 45 Kg has been measured between n measurements (sampling every 20 ms). The number n of measurements is defined by the user (see module configuration).

### The Measurement status word

The data word is coded as follows :

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
			Manual tare	Zero tracking	Zero	instability	NET weight
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Fault / Supervis.	Calibration	Processing in progress	Protected module	Over Satur	Under Satur	Q1	Q0

$%IWxy.0.4:X0$  is the image of output Q0.

$%IWxy.0.4:X1$  is the image of output Q1.

$%IWxy.0.4:X2$  indicates an excessively low voltage. The measurement is abnormal and there is a strong possibility of a sensor or wiring fault.

$%IWxy.0.4:X3$  indicates an excessive voltage on the module input.

$%IWxy.0.4:X4$  indicates a sealed module.

$%IWxy.0.4:X5$  indicates processing in progress (tare, zero reset, etc).

$%IWxy.0.4:X6$  indicates calibration during processing.

$%IWxy.0.4:X7$  indicates a fault during the command.

$%IWxy.0.4:X8$  indicates a NET weight measurement.

---

%IWxy.0.4:X9 indicates measurement **instability**. It is set when the measurement is outside the stability range during the time defined. The extent of the stability range and the time are defined during configuration.

%IWxy.0.4:X10 indicates zero. It is set when the zero offset is no greater than +/- 1/4 of a scale division.

%IWxy.0.4:X11 indicates that zero tracking is active.

%IWxy.0.4:X12 indicates a predefined or manual tare (language element specific to the module, accessible in read-only). It is set when the tare is not the result of a tare order but of a user entry.

%IWxy.0.4:X13 to X15 are not used.

### **The tare value**

This word is used to display the current tare value in the same format as the weight and is memorized by the module.

### **The offset memory**

This word is used to display the current offset in high resolution format and is memorized by the module.

It is reset to 0 at each calibration.

**Validity conditions for measurements and module :**

A **channel fault** bit is associated with the channel. To ensure that numerical values are valid, it is necessary to check the fault bit.

**Note:**

The fault bit goes to 1 when an error condition appears on the channel (underload/overload, etc). For more details about the fault, check the channel status.

In addition, there is an error detection bit **at module level**. This bit goes to 1 when the channel is faulty. In the case of this module, it is always equivalent to the previous one.

**Accessing the module fault bit :**

%I xy.MOD.ERR	Module fault bit
---------------	------------------

**Accessing the channel fault bit:**

%I xy.0.ERR	Module measurement channel fault bit
-------------	--------------------------------------

In the case of the weighing module, the module and channel data is identical.

**Behavior of fault bits :**

Depending on the type and seriousness of the faults, the corresponding fault bit may be transient (resets to 0 when the fault disappears) or memorized (stays at 1 even if the fault disappears).

Memorized faults	Transient faults :
- Internal fault	- Range exceeded fault
	- Application fault
	- Configuration fault
	- Communication fault
	- Protected module, parameter refused
	- Underload fault
	- Overload fault
	- Module not calibrated
	- Module tare being executed
	- Module zero reset being executed
	- Module calibration being executed
	- Module forced calibration being executed
	- Operating fault
	- Module missing



---

## Accessing the module status

A status word is also associated with the module.

<b>%MWxy.MOD.2</b>	Module status
--------------------	---------------

The following status word bits concern the various types of fault. In the event of a fault, the corresponding bit is set to 1.

Module status address : <b>%MWxy.MOD.2</b>	
Bit no :	Role
0	Internal fault : module is inoperative
1	Operating fault : communication or application fault
2	Terminal block fault : not used
3	Self-tests in progress on the module : not used
4	Reserved
5	Configuration fault : recognized module is not the required module
6	Module missing fault : module missing or off
7	Down_fault : not used

### Accessing the various status words :

The status words are accessible by launching an explicit read operation via the READ\_STS instruction. The syntax is as follows :

Read module channel status : READ\_STS %CH xy.0

Read module status : READ\_STS %CH xy.MOD

Access to status words is conditional on a module fault or channel fault.



**Accessing channel status bits :**

<b>%MWxy.0.2</b>	Measurement channel status
------------------	----------------------------

The following status word bits concern the various types of fault and channel status. In the event of a fault, the corresponding bit is set to 1.

Channel status address : <b>%MWxy.0.2</b>	
Bit no :	Role
0	External fault : overload or underload during calibration
1	Range overrun fault (1)
2	External fault : measurement module saturated
3	External fault : module sealed, configuration refused
4	Internal fault : module is inoperative
5	Configuration fault : the module present is not the module declared during configuration
6	Communication fault with the processor
7	Application fault
8	Protected module fault, parameter refused : the module refuses the parameter (if it would influence the measurement)
9	Module not calibrated
10	Overload fault
11	Underload fault
12	Tare mode
13	Zero mode
14	Calibration mode
15	Forced calibration mode

- (1) This bit is only activated when the **gross filtered and measured weight** exceeds the overload threshold or is below the underload threshold. The two faults are distinguished by specific faults: underload fault or overload fault.

**Comment :**

Internal fault : any internal fault detected on the module sets the discrete outputs to their fallback values (electrical 0).

---

D

## 2.1 Introduction

To debug the weighing function, calibration must first be performed.

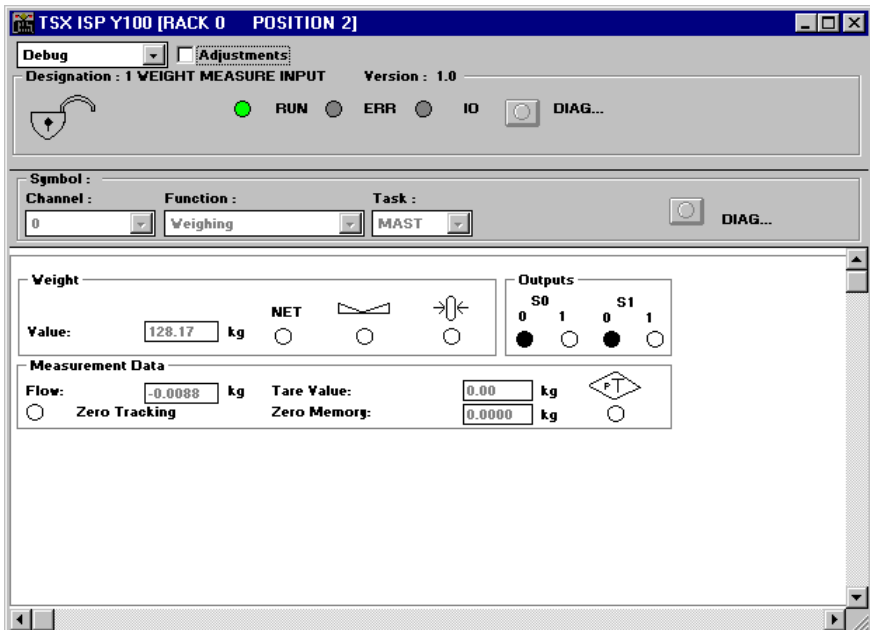
The commands available on the module are :

- calibration,
- zero reset,
- tare,
- temporary display of manual tare,
- measurement freeze,
- enable thresholds,
- disable thresholds.

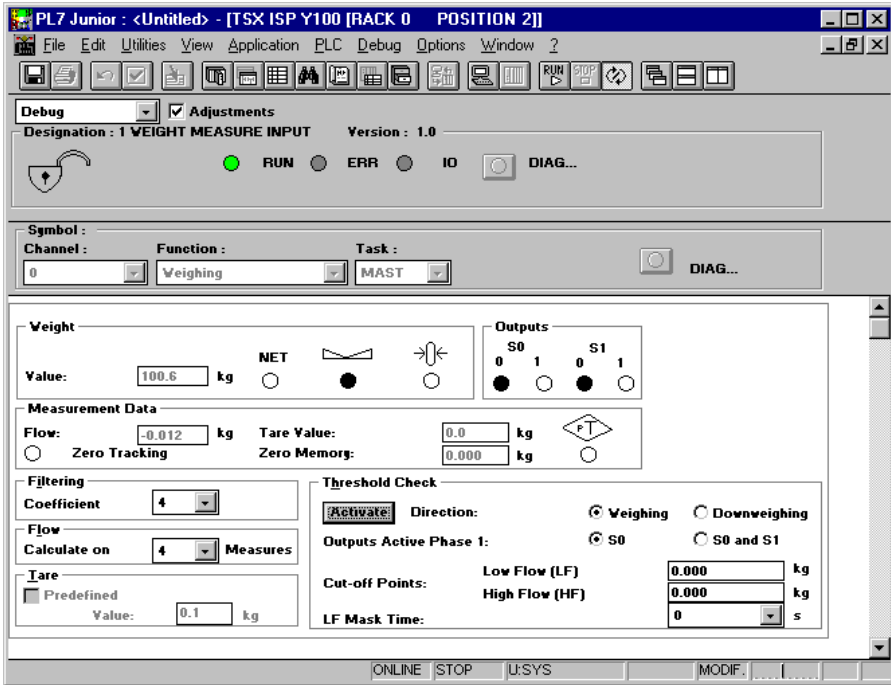
The adjustments affect :

- filtering,
- the manual tare value,
- the modification of threshold values,
- the output control logic,
- the LF mask time,
- the number of measurements used to calculate the flow rate.

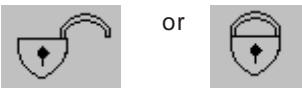
The operator can display the status of the main data on the following screen :



By selecting the “adjust” box, the screen displays access and additional data for executing this function.



The part of the screen with a gray background gives data on the module status.



or

Indicates whether or not the module is sealed (lock closed = sealed) (see section 6).



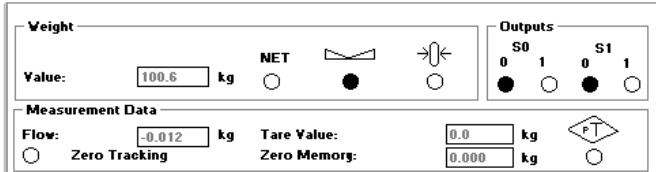
The first indicator light shows the module operating mode. The second signals an internal error and the third an external fault (see section 2.3, Display of module status).



This indicator light turns red in the event of a fault associated with the weighing function. Access to the fault details is via the Diag button located beneath it.

This screen consists of two 2 distinct parts :

- A **dynamic display zone** for the main data associated with weighing.





It shows dynamic data relating to :

- **Weight :**

**Value:** The value of the current weight in the defined unit. In the event of the module detecting a fault in the analog measurement system, ERR is displayed on the screen.

**NET** The Net Weight indicator light comes on if the module returns net weight data : otherwise the data is gross weight.

 The Stable Measurement indicator light shows that the measurement is within the defined stability range.

 The Zero Zone indicator light comes on if the weight measurement is within the zero range.

- **Outputs :**

- The indications correspond to the physical states of outputs Q0 and Q1.

- **Measurement Data :**

- the flow rate, indicated in units per measurement period (20 ms),
- the current tare value,
- the zero memory value corresponding to the zero offset since the last calibration,
- the TW indicator light shows that the tare value has been introduced manually and not been measured,
- the Zero tracking indicator light shows that function parameters have been set.

**Note :**

In the case of invalid data, ERR is displayed before the value.

---

- A parameter adjustment zone.

The image shows a software interface for parameter adjustment. It is divided into two main sections: 'Filtering' and 'Threshold Check'.  
The 'Filtering' section contains:

- A 'Coefficient' dropdown menu with the value '4' selected.
- A 'Flow' section with a 'Calculate on' dropdown menu set to '4' and the label 'Measures'.
- A 'Tare' section with a 'Predefined' checkbox and a 'Value' input field containing '0.1' followed by 'kg'.

The 'Threshold Check' section contains:

- An 'Activate' checkbox.
- A 'Direction' section with radio buttons for 'Weighing' (selected) and 'Downweighing'.
- An 'Outputs Active Phase 1' section with radio buttons for 'S0' (selected) and 'S0 and S1'.
- 'Cut-off Points' for 'Low Flow (LF)' and 'High Flow (HF)' both set to '0.000' kg.
- 'LF Mask Time' set to '0' s.

This is used to access modification and display of the following parameters :

- **Filtering :**

The operator may modify the value of the input measurement filter coefficient. He may select a value from 0 (unfiltered) to 19 (see section 3.2.1, configuration parameters).

**Note :**

The stronger the filtering, the longer the response time.

- **Flow rate :**

The operator may modify the number of measurements used to calculate the flow rate. The selection lists the values 2, 4, 8, 16, 32 and 64.

- **Tare :**

The operator may introduce a predefined tare by checking the corresponding box and entering this tare in the unit defined.

- **Threshold check :**

These parameters are only displayed if the 'threshold check' option has been activated during configuration. All parameters are taken into account once the command in the 'Edit' menu has been enabled.

- **Activate** : the operator may activate outputs Q0 and Q1.

- **Weighing/Downweighing direction** : used to modify the direction in which thresholds are taken into account.

- **Active outputs phase 1 : 'Q0' or 'Q0 and Q1'** : used to select the active outputs during the first batching phase.

- **Low flow (LF) and High flow (HF) cut-off points**: used to modify the values of these thresholds (see section 3.1.1, configuration parameters).

- **LF mask time** : used to modify the mask time delay when low flow commences.

(For more information on these parameters, see section 3.2.1, configuration parameters).

## 2.2 Sending commands to the weighing module

To transmit a command to the module, it is necessary to :

- first, select the type of command by setting to 1 the corresponding bit in the word “Command Type”,
- second, enter the command parameter, if required (eg : standard load in high resolution format).

### Language elements used to send the commands

The module command orders can be accessed in the PLC memory via the following language elements :

Command bits	Corresponding commands
%MWxy.0.3:X0	Save calibration values in module.
%MWxy.0.3:X1	Zero load calibration.
%MWxy.0.3:X2	Standard load calibration (normal condition).
%MWxy.0.3:X3	Cancel command.(1)
%MWxy.0.3:X4	Tare order.
%MWxy.0.3:X5	Zero reset order.
%MWxy.0.3:X6	GROSS weight return order.
%MWxy.0.3:X7	Display the manual tare for 3 seconds.
%MWxy.0.3:X8	Enable thresholds.
%MWxy.0.3:X9	Disable thresholds.
%MWxy.0.3:X10	Forced calibration.
%MWxy.0.3:X11	Save adjustment coefficients in processor.
%MWxy.0.3:X12	Standard load calibration in downgraded conditions (standard < 70% of maximum range).
%MWxy.0.3:X13 to X15	Not used

(1) Canceling commands only affects the following current commands : Calibration, Zero reset, Tare.

---

### **Sending commands to the module :**

Commands are sent to the module using the WRITE\_CMD instruction with the following syntax :

**WRITE\_CMD %CHxy.0**

This instruction sends the order to the module and waits for it to be acknowledged. This waiting time may require several task scans.

The module can only interpret one command at a time. If a command is required, and the previous command is still current, the latter is refused. There may never be more than one bit set to 1 in the command word.

### **Checking parameters have been taken into account**

As several task scans may be required for the module to take commands into account, two standardized memory words check the exchanges : %MWxy.0.0 and %MWxy.0.1

The first word, %MWxy.0.0 indicates that an exchange is in progress.  
The second word, %MWxy.0.1 gives the exchange report.

Bits %MWxy.0.0:X1 and %MWxy.0.1:X1 are associated with commands.

Bit %MWxy.0.0:X1 indicates that the command has been sent to the module.

Bit %MWxy.0.1:X1 shows whether the command has been accepted by the module.

The application fault bit of channel status %MWxy.0.2:X7 signals that a command or parameter has been refused.



---

### 2.2-1 Calibration

Calibration of the analog measurement system consists of making a weight value correspond to an electrical signal from the load cells. This is done on site, during installation, and is absolutely necessary to ensure that the measurement is valid.

All uncalibrated modules are in channel fault mode. The first calibration must be made in full (zero load and standard load), otherwise the data returned has no significance.

Calibration is not possible if the PLC processor is equipped with a Flash-Eprom type memory card (TSX MFP 032P or TSX MFP 064P or TSX MFP 0128P).

It is possible to perform recalibration during the life of the module. The electronic characteristics do not require regular recalibration. However, the legal requirements or mechanical characteristics of the application may require this calibration, especially for commercial transactions.

Three commands are described in this paragraph :

- normal calibration (the calibration function must be executed with a standard load  $\geq 70\%$  the maximum range),
- downgraded calibration (if, for whatever reason, the calibration cannot be executed in the conditions described above),
- forced calibration (to recover adjustments executed on a different module for maintenance or duplication purposes).

#### Calibration principle :

Calibration is executed in two stages :

- Zero (zero load) determines the offset,
- Standard load, used to define the analog measurement system gain.

#### Calibration operating mode

In calibration mode, the channel is signaled as faulty.

#### Calibration method

Calibration may be executed on a PL7 station connected to the PLC by means of the calibration screen.

It may be also be executed via a man-machine interface using PL7 language instructions.

#### Calibration precautions

Any change in the load cell supply requires full recalibration. This calibration forces outputs Q0 and Q1 to 0.

#### Note :

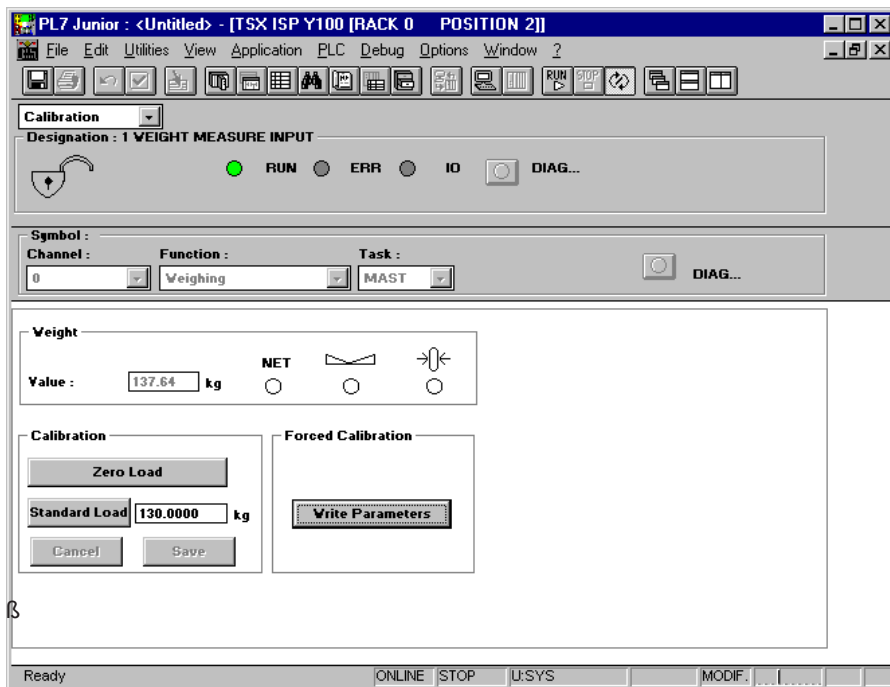
Calibration is independent of the configured filter, but takes into account the Metrological Data, Stability and Load Cell Power Supply parameters from the Configuration screen.

## Calibration screen

To facilitate this operation, a calibration screen can be accessed from the operating screen.

The upper part displays the weight and its characteristics (Net or Gross) : the lower left of the screen is used to execute calibration and the lower right of the screen is used to force calibration.

D





**Calibration procedure using the calibration screen**

User operations with the screen	Module behavior
Power up the rack.	The product is initialized, performs self-tests and receives its configuration.
Enter data in the Calibration screen via the command status screen	
Check that the weigher is empty	
Execute <b>zero load calibration</b> (take load receptacle into account). The zero load calibration command is executed by selecting the load under “Zero load” on the calibration screen.  ↓	This phase requires approximately one minute. The “Zero load” changes to reverse video during this phase and an hour-glass icon appears. The module changes to channel fault and no measurements are valid : TheCalibration_in_progress indicator light changes state, The module indicates that the zero load reference has been acquired and processes the reports :
Position the <b>standard load</b>	
Enter the value of the standard load and execute the standard load calibration command. The standard load calibration is ordered by selecting the load under “Standard load” on the calibration screen.  ↓  If the measurement is stable (1), the standard load calibration is executed	This phase requires approximately one minute. The module verifies the standard load in relation to the maximum range (2). The Calibration_in_progress indicator light changes state, The module reads the standard load reference and processes and determines the reports.

(1) If the measurement is unstable, a message signals the fault : “Provisional adjustment, unstable weigher”. Accept : Accept the measurement and save it. Cancel : Quit the current phase.

(2) If the Standard load / Maximum Range ratio is less than 70%, the message “Calibration conditions not satisfactory, standard load is too low” alerts the operator. The operator may accept the adjustment under these conditions by selecting the “Accept” key or recommence calibration using the “Quit” key.

**Note :**

The stability check is only possible if a zero load calibration and a standard load calibration have been executed.

Partial calibration phases are possible, but do not enable the optimum characteristics of the analog measurement system to be used.

<p>Save or Cancel the calibration procedure :</p> <p>Save</p> <p>Cancel</p>	<p>→ the module and the processor take into account the parameters resulting from the calibration. These parameters are saved in the module. During the write phase, the measurement This fault disappears as soon as the write operation is executed (channel fault and calibration in progress disappear). The measurement is valid.</p> <p>→ the module reverts to its previous parameters. The application fault disappears. The measurement is valid.</p>
<p>Quit the calibration screen : "status/command" button</p> <p>⇓</p> <p>Return to the command status screen</p>	

### Comment :

The procedure can be stopped at any time by pressing 'Cancel'. The module reverts to its previous parameters. The current calibration parameters are lost.

The validation of the procedure is effective only if the calibration has in fact been executed in the module. If there is a calibration problem (standard load reference values outside permitted limits, etc), the screen cannot be validated. Either the error must be corrected or the procedure must be canceled via "Cancel".

### Forced calibration

This function responds to the need for rapid maintenance procedures (immediate module replacement).

It consists of loading the adjustments made using one module into another module. These adjustments are saved at the time of the last calibration executed on the replaced module.

Using the 'Load' key, the user imports the adjustments stored in the processor. This operation may also be performed by setting the associated command bit, %MWxy.0.3:X10, to 1.

### Note :

The next calibration must be made in full (zero load and standard load).

If a module fails and the operator wishes to replace it quickly, he may force the calibration of the previous module.

## Data associated with the calibration

Several language elements are used to create and supervise the calibration mechanism. The calibration screen facilitates this procedure, but it may also be directly performed using reserved data.

- **A command word :**

This is word %MWxy.0.3 : this word is updated during calibration.

- **Save calibration** (%MWxy.0.3:X0) : order (if =1) sent to the module to validate and save the parameters determined in the calibration phase.
- **Standard load** (%MWxy.0.3:X2) : order (if =1) sent to the module to calibrate the channel using the standard load.
- **Zero load** (%MWxy.0.3:X1) : order (if =1) sent to the module to calibrate the channel using the zero load.

- **A control parameter** containing the value of the **standard load** : this is word %MD xy.0.4 (high resolution format).

- **A status bit**

- **Calibration in progress** : status bit sent by the module which changes state at the beginning and end of calibration.

Bit 9 (uncalibrated module fault) of word %MWxy.0.2 is set, until the first calibration order.

The table below shows the data involved during calibration :

Type of command	Associated data
Save calibration in module	%MWxy.0.3:X0
Zero load	%MWxy.0.3:X1
Standard load (normal)	%MWxy.0.3:X2
Forced calibration	%MWxy.0.3:X10
Save calibration in processor	%MWxy.0.3:X11
Standard load (downgraded)	%MWxy.0.3:X12
Control parameter	Associated data
Value of the standard load	%MD xy.0.4
Report	Associated data
Calibration in progress %IW xy.0.4:X6	
Instability	%IW xy.0.4:X9
Overload or underload during calibration.	%MWxy.0.2:X0
Uncalibrated module	%MWxy.0.2:X9
Calibration mode	%MWxy.0.2:X14
Forced calibration mode	%MWxy.0.2:X15

---

## 2.2-2 Saving adjustments in the processor

If the calibration screen is used during calibration, the "Save" key saves the parameters both in the module and in the processor if the application is not stored in a write-protected cartridge.

A command exists which enables current weighing module adjustment parameters to be saved directly in the processor.

This function is triggered by setting bit **%MWxy.0.3:X11** to 1.

D



### 2.2-3 Tare

When a load, called the tare, is placed on the load receptacle, this function sets its measured NET weight to zero. An offset value can then be applied to the measurement to ensure it conforms to the value required by the user.

The tare value is accessible in the PLC memory : it is stored in word %ID xy.0.5, in the weight format.

It may be saved by the application.

**Note :**

When no tare operation has been executed, the NET weight is equivalent to the GROSS weight.

**Tare execution conditions**

The acceptance conditions for execution of the Tare command are as follows :

- The measurement is stable.
- The measurement is less than the maximum range.
- The measurement must be positive.

**Tare procedure using Tare command**

User operations	Module behavior
1 - Enter WRITE_CMD setting the tare order (%MWxy.0.3:X4 = 1).	
2 - Confirm execution, application in RUN.	<p>The module switches to tare mode and sends the Processing_in_progress report %IWxy.0.4:X5 = 1.</p> <p>Proceed to tare acquisition.</p> <p><b>Note :</b> the weight is measured and memorized in the associated object %IDxy.0.5. It will be subtracted from any subsequent GROSS weight measurement to determine the NET weight. End acquisition : Processing_in_progress = 0</p>
3 - Check successful execution of command : Status of Processing_in_progress %IWxy.0.4:X5	

---

The module remains in Processing\_in\_progress state as long as the acceptance conditions are not fulfilled or no cancel command order is received.

**Comment :**

All tare values are deleted when the configuration is changed. Execution of a Tare command deletes any tare entered in manual mode (WRITE\_PARAM) and resets the “manual” tare indicator to zero.

Similarly, a GROSS weight return order enables any tare to be deleted. No acceptance conditions are necessary.

The table below summarizes the data used for a tare.

Command	Associated data
Tare order	%MWxy.0.3:X4
Display	Associated data
Tare value	%IDxy.0.5
Tare in progress	%IWxy.0.4:X5

**Example :**

Send a tare order to the weighing module at slot 2 of rack 0

```
LD TRUE
S %MW 2.0.3:X4
[WRITE_CMD %CH2.0]
```

This order involves :

- Sending the command.
- Setting bit %MW2.0.0:X1 to 1 to indicate that the command is currently being sent. This bit remains at 1 until the module sends a report. The bit then returns to 0 and the exchange report bit becomes significant.
- Exchange report bit %MW2.0.1:X1 sets to 1 in the event of problems during the exchange. Value 0 indicates that the command has been accepted by the module.

%IW2.0.4:X5 remains at 1 (processing in progress) as long as the acceptance conditions are not fulfilled (for example awaiting measurement stability). The application fault bit for the channel status is set to 1 (module currently executing command). As for all commands, the order may be canceled by sending the command “cancel current command”.



**2.2-4 Zero reset**

This function consists of setting the measured weight to zero and canceling any tare. The zero indicator light is then set.

It is controlled by the Zero Reset command.

Corrections to the measurement are stored in word %IDxy.0.7 in high resolution format. They may be saved by the application. This parameter is reset to zero on each calibration.

**Conditions for executing zero reset**

The acceptance conditions for executing zero reset are as follows :

- The measurement is in GROSS weight.
- The measurement is stable.
- The measurement must be within the extent of the zero range as defined during configuration.

**Zero Reset procedure via the Zero Reset command**

User operations	Module behavior
1 - Enter WRITE_CMD setting the Zero Reset order (%MWxy.0.3:X5 = 1). 2 - Confirm execution with application in RUN.	The module switches to zero_reset mode and sends the Processing_in_progress report %IWxy.0.4:X5 = 1.  The module then acquires the measurement and memorizes the new value in the offset memory %IDxy.0.7. Processing_in_progress = 0 signifies the end of the procedure.
3 - Check successful execution of command : Status of Processing_in_progress	

**Note :**

The module remains in the Processing\_in\_progress state as long as the acceptance conditions are not fulfilled or no cancel command order is received.

All zero resets are deleted when the configuration is changed.



---

The table below summarizes the data used for a zero reset.

Command	Associated data
Zero reset order	%MWxy.0.3:X5
Display	Associated data
Offset memory	%IDxy.0.7
Report	Associated data
Processing in progress	%IWxy.0.4:X5Command

### Example :

Send a Zero\_Reset order to the weighing module at slot 2 of rack 0

```
LD TRUE
S %MW 2.0.3:X5
[WRITE_CMD %CH2.0]
```

This order involves :

- Sending the command.
- Setting bit %MW2.0.0:X1 to 1 to indicate that the command is currently being sent. This bit remains at 1 until the module sends a report. The bit then returns to 0 and the exchange report bit becomes significant.
- Exchange report bit %MW2.0.1:X1 sets to 1 in the event of problems during the exchange. Value 0 indicates that the command has been accepted by the module.

%IW2.0.4:X5 remains at 1 (processing in progress) as long as the acceptance conditions are not fulfilled (for example awaiting measurement stability). The status channel application fault bit is set to 1 (module currently executing command).

As for all commands, the order may be canceled by sending the command "cancel current command".

**2.2-5 Order to return to gross weight**

This function consists of canceling the tare value so that the current weight equals the gross weight.

The current weight is stored in word %IDxy.0.0 in the format defined during configuration.

**Conditions for executing the return to gross weight**

This command requires no special execution conditions.

**Procedure to return to gross weight via the return to gross weight command**

User operations	Module behavior
1 - Enter WRITE_CMD setting the Return to gross weight order (%MWxy.0.3:X6 = 1).	
2 - Confirm execution with application in RUN.	The module switches to return to gross weight mode. The module proceeds to reset the tare to zero The Net indicator = 0 signifies the end of the procedure (%IWxy.0.4:X8 = 0).
3 - Check successful execution of command : Status of Net indicator	

The table below summarizes the data used for a zero reset.

Command	Associated data
Return to gross weight order	%MWxy.0.3:X6
Display	Associated data
Weight measured	%IDxy.0.0
Current tare value	%IDxy.0.5
Report	Associated data
Processing in progress	%IWxy.0.4:X5
Gross weight	%IWxy.0.4:X8 = 0



---

## 2.2-6 Order to display the manual tare for 3 seconds

This function displays the manual tare on the display unit for 3 seconds.

### Execution conditions

This command requires a manual tare to have been configured.

### Procedure to temporarily display manual tare

User operations	Module behavior
1 - Enter WRITE_CMD setting the order to temporarily display manual tare (%MWxy.0.3:X7 = 1).	
2 - Confirm execution with application in RUN.	The module manages data normally. Only the values displayed on the TSX XBT H100 indicate the manual tare.
3 - At the end of 3 seconds, the display unit once more shows current values.	

The table below summarizes the data used for displaying the tare.

Command	Associated data
Order to display the tare	%MWxy.0.3:X7
Display	Associated data
The data on the display unit indicates the manual tare	

### 2.2-7 Orders to enable and disable thresholds

These functions are mainly used to coordinate the control of outputs in relation to the control system managed by the processor. The control thresholds option must have previously been confirmed on the configuration screen.

#### Operating principle

Action on the outputs is executed via the “Validate Thresholds” command. When this command is executed, the threshold control cycle is triggered. It stops when the condition corresponding to the Low flow cut-off point is reached.

A disable command may, if necessary, stop the control cycle and reset outputs Q0 and Q1 to 0.

#### Procedure to Enable thresholds

User operations	Module behavior
1- Enter a WRITE_CMD to set the threshold enable order. (%MWxy.0.3:X8 = 1).	
2 - With the application in RUN, modify the threshold values, output logic and mask time as required (see adjustment).  End of adjustment	Modifications taken into account by the module
3 - Launch the threshold enable operation using the WRITE_CMD instruction	The module interprets the request, sets outputs Q0 and Q1 and the corresponding image bits : %IWxy.0.4:X0 and %IWxy.0.4:X1.

---

## Procedure to Disable thresholds

User operations	Behavior of the module
1 - Enter a WRITE_CMD to set the threshold disable order. (%MWxy.0.3:X9 = 1).	
2 - Confirm execution with application in RUN.	The module sets the outputs to rest state and sets the image bits to 0.

## Language elements associated with threshold control

The table below summarizes the data used to enable and disable thresholds :

Command	Associated data
Enable threshold order	%MWxy.0.3:X8
Disable threshold order	%MWxy.0.3:X9
Display	Associated data
Current flow rate	%IDxy.0.2
High flow threshold	%MDxy.0.8
Low flow threshold	%MDxy.0.10
Output logic	%MWxy.0.12.
LF masking time	%MWxy.0.13.
Current position of Q0	%IWxy.0.4:X0
Current position of Q1	%IWxy.0.4:X1

## 2.3 Adjustments

### Adapting the process and customizing the measurement

Depending on the operation to be performed or the product to be processed, the process may require modification of output behavior, modification in the number of samples used to calculate the flow rate, etc.

These modifications are performed using the following data (see section 2.3.2):

Adjustable parameters :	Corresponding data
Filtering coefficient	%MWxy.0.6
“Manual” tare value	%MWxy.0.7
Cut-off points (thresholds)	%MDxy.0.8 & %MDxy.0.10
Q0 and Q1 output logic	%MWxy.0.12
LF mask time	%MWxy.0.13
Number of measurements used to calculate the flow rate	%MWxy.0.14

The user may :

- Modify an adjustment parameter via a program,
- Send the adjustment parameters to the module,
- Check that the module is taking the parameters into account,
- Read the value of the adjustment parameters in the module and thus update the PLC memory,
- Save the adjustment parameters,
- Restore the value of the saved parameters to the PLC memory.

The instructions used for these operations are as follows :

Instruction	Function performed
WRITE_PARAM %CH xy.0	Sends the contents of the parameters on the previous table to the weighing module
READ_PARAM %CH xy.0	Reads the adjustment parameters in the module and updates the table previously mentioned.
SAVE_PARAM %CH xy.0	Saves the adjustment parameter values in the processor memory zone. These parameter values are those used on a PLC cold restart.
RESTORE_PARAM %CH xy.0	Used to reload the adjustment parameters with the values entered during module configuration or on the last SAVE_PARAM

The module can process several adjustments simultaneously.

---

### 2.3-1 PL7 instructions used for adjustment

The user may access the CPU data memory via PL7. To perform adjustment operations, access to the module data itself is required. Access is via specific instructions which provide the following functions.

D

#### Send the adjustment parameters to the module :

The **module** channel parameters are sent using the **WRITE\_PARAM** instruction with the following syntax :

**WRITE\_PARAM %CH xy.0**

This instruction sends the contents of the parameters to the module and waits for acknowledgment. This may require several task scans.

#### Checking parameters have been taken into account

As several task scans may be required for the module to take commands into account, two memory words are used to check the exchanges :

**%MWxy.0.0 and %MWxy.0.1**

- The first word **%MWxy.0.0** indicates that an exchange is in progress,
- The second word **%MWxy.0.1** gives the exchange report,
- Bits with the number 2 are associated with adjustment parameters :
  - Bit **%MWxy.0.0:X2** indicates that adjustment parameters have been sent to the module,
  - Bit **%MWxy.0.1:X2** shows whether the adjustment parameters have been accepted by the module.

#### Example :

Write the parameters of the module at slot 2 of rack 0:

**WRITE\_PARAM %CH2.0** involves :

- Sending the adjustment parameters,
- Setting bit **%MW2.0.0:X2** to 1 to indicate that the adjustment parameters are currently being sent. This bit remains at 1 until the module sends a report. The bit then returns to 0 and the exchange report bit becomes significant.
- Exchange report bit **%MW2.0.1:X2** sets to 1 in the event of problems during the exchange. Value 0 indicates that the data has been accepted by the module.



If the module is protected (sealed), modification of the filtering coefficient is not authorized. If a `WRITE_PARAM` instruction is sent to the module with a filtering coefficient which is different from the current coefficient, the application fault (illegal parameters received) and the sealed module fault are set. The module continues to use the current filtering coefficient.

### Reading the adjustment parameters values :

The `READ_PARAM` instruction is used to read the **module** adjustment parameters and update the PLC memory. Reading the adjustment parameters may require several task cycles.

The adjustment parameters for the module channel are read via the `READ_PARAM` instruction with the following syntax :

```
READ_PARAM %CH xy.0
```

### Saving adjustment parameters :

The `SAVE_PARAM` instruction is used to copy the current values of the module adjustment parameters to the back-up zone defined in the processor memory. The back-up zone is not accessible using the language .

The execution of this instruction may require several task scans.

Module adjustment parameters are saved via the `SAVE_PARAM` instruction with the following syntax :

```
SAVE_PARAM %CH xy.0
```

### Restoring the saved adjustment parameters :

The `RESTORE_PARAM` instruction is used to restore the saved values of adjustment parameters in the processor memory and in the module.

Module adjustment parameters are restored via the `RESTORE_PARAM` instruction with the following syntax :

```
RESTORE_PARAM %CH xy.0
```

---

## 2.3-2 Adjustment parameters

### Filtering coefficient :

%MWxy.0.6	Filtering coefficient
-----------	-----------------------

The permitted values for the filtering coefficient are between 0 and 19 inclusive.

### “Manual” tare value :

%MWxy.0.7	manual tare value
-----------	-------------------

The permitted values for the “manual” tare value are between 0 and 65 535 inclusive : they may not exceed the maximum range.

### Cut-off points (thresholds) :

%MD xy.0.8	High flow cut-off point Q0
%MD xy.0.10	Low flow cut-off point Q1

The permitted values for thresholds are between 0 and the maximum range in high resolution format.

If no threshold check has been defined during configuration, no detection processing is executed. The default value of these thresholds is zero.

### Note :

- In weighing HF < LF < Maximum range,
- In downweighing LF < HF < Maximum range.

The module performs a threshold value consistency check. If this logic is not observed, the thresholds are refused.

### Output logic :

%MWxy.0.12	Output logic
%MWxy.0.12:X0	0: Weighing 1: Downweighing
%MWxy.0.12:X1	0: Q0 then Q1 1: Q0 and Q1 then Q1

---

**LF mask time :**

<b>%MWxy.0.13</b>	<b>LF mask time</b>
-------------------	---------------------

The permitted values are between 0 and 15 in steps of  $1/10^{\text{th}}$  second (0 = 0s, 1 = 0.1s, 2 = 0.2s, etc).

**Number of measurements used to calculate the flow rate :**

<b>%MWxy.0.14</b>	<b>Number of measurements used for flow rate</b>
-------------------	--

The permitted values are 2, 4, 8, 16, 32 or 64.

---

### 2.3-3 Adjustment procedures

The user may perform certain adjustment operations :

#### **Adjustment procedure for measurement filtering (%MWxy.0.6)**

To adjust measurement filtering, proceed in a step-by-step fashion until satisfactory measurement characteristics are obtained.

#### **Adjustment procedure for thresholds and outputs (%MDx0.8, %MDx0.10, %MWx0.12, %MWx0.13, %MWx0.14)**

To adjust thresholds, proceed as follows :

When batching finishes, the program is able to measure the in-flight error by calculating the differential between the weight of the product in the weigher and the theoretical batch setting for the product. Using this measurement, it can correct the in-flight error parameter or the low flow quantity parameter according to the formula selected, if necessary taking flow rates into account. The threshold(s) are then modified and sent back to the module.

#### **Note :**

The thresholds are calculated by the application, as a function of the batching quantity, the in-flight error and the low flow quantity, in accordance the following formulae :

Low flow cut-off point = batch setting - in-flight error

High flow cut-off point = Low flow cut-off point - low flow quantity

#### **Procedure for adjusting the number of measurements used for the flow rate (%Mwx0.14)**

To adjust the number of samples used to calculate the flow rate, proceed step-by-step until satisfactory measurement characteristics are obtained.

**2.3-4 Reading configuration parameters**

All parameters entered during **module** configuration can be accessed via the program in read-only mode.

**Maximum range :**

The maximum range configured for the measurement channel can be read by memory word %KW in the constant zone. It has the following syntax :

%KW xy.0.0

**Unit/Scale division :**

The unit and the scale division, configured for the measurement channel, can be read by memory word %KW in the constant zone. The scale division is always defined in the same unit as the measurement and has the following syntax :

%KW xy.0.1

with the unit coded on 3 bits of the low order byte

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Unit							
(bits 0 to 2 used)				Corresponding unit			
0				g			
1				kg			
2				t			
3				LB			
4				oz			
5				no unit			

and the scale division coded on 5 bits of the high order byte

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
Scale division							
(28 to 212)		Scale division value		(28 to 212)		Scale division value	
0		0.001		11		5	
1		0.002		12		10	
2		0.005		13		20	
3		0.01		14		50	
4		0.02		15		100	
5		0.05		16		200	
6		0.1		17		500	
7		0.2		18		1000	
8		0.5		19		2000	
9		1		20		5000	
10		2					



**Stability / Zero / Overload threshold / Use of outputs/ format :**

The extent of the range and of the stability time, the extent of the zero range and of the activity of zero tracking, of the overload threshold, the use of outputs and the weight formats, configured for the measurement channel, can be read by memory word %KW in the constant zone. It has the following syntax :

<b>%KW xy.0.2</b>
-------------------

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Time of stability				Extent of stability			

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
Format	Outputs	Zero track. active	Manual tare	Zero extent	Load cell supply	Overload	

Detail of bits 0 to 2 :

Extent of stability : ( bits 0 to 2 )	
Value read	equivalent (in 1/4 scale division)
0	2
1	3
2	4
3	6
4	8

Detail of bits 4 to 5 :

Stability time : ( bits 4 to 5 )	
Value read	equivalent (in seconds)
0	0.4
1	0.5
2	0.7
3	1

Overload : (bits 8 to 9)	
Values read	Type of overload selected
0	Maximum range + 9 scale divisions
1	Maximum range + 2% of maximum range
2	Maximum range + 5% of maximum range

Load cell power supply : (bit 10)	
bit 10 at 0	Continuous
bit 10 at 1	Switched

<b>Extent of zero : (bit 11)</b>	
bit 11 at 0	2% of maximum range
bit 11 at 1	5% of maximum range

<b>Predefined tare : (bit 12)</b>	
bit 12 at 0	No predefined tare
bit 12 at 1	Predefined tare

<b>Zero tracking activity : (bit 13)</b>	
bit 13 at 0	Zero tracking inactive
bit 13 at 1	Zero tracking active

<b>Use of outputs : (bit 14)</b>	
bit 14 at 0	Outputs not used
bit 14 at 1	Outputs used

<b>Format : (bit 15)</b>	
bit 15 at 0	Legal format (physical unit with fixed decimal)
bit 15 at 1	High resolution (1/100 <sup>th</sup> physical unit with fixed decimal)

---

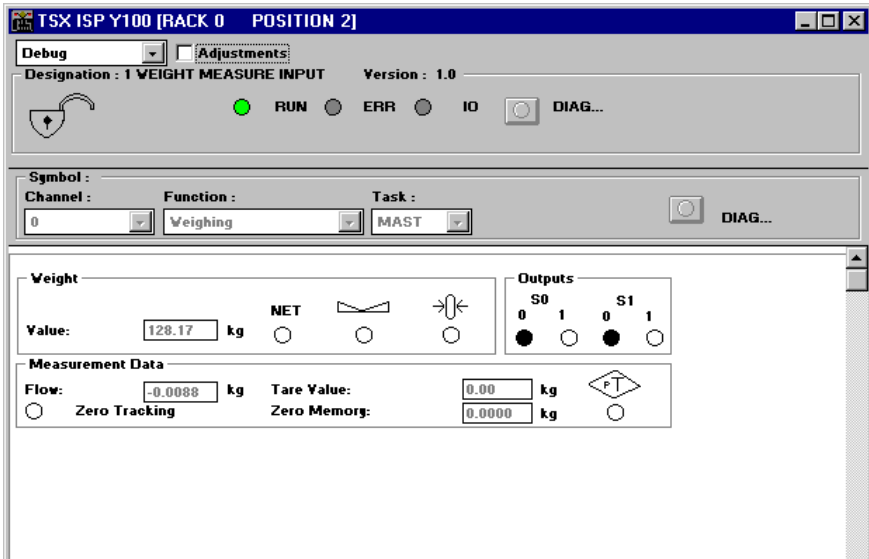
D



### 3.1 Weighing operation using PL7

Once the weighing application is operational, it may be supervised via the debug screen.

The Debug screen dynamically shows the principal weight measurement data.



#### Module information



or



Indicates whether or not the module is sealed (lock closed = sealed) (see section 6).








The first indicator light shows the module operating mode. The second signals an internal error and the third an external fault (see section 2.3, display of module status).

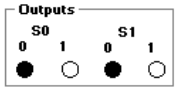


This indicator light turns red in the event of a fault associated with the weighing function. Access to the fault details is via the Diag button located beneath it.




## Weight data

<b>Value:</b> <input type="text"/> g	Gives the current value of the weight measured.
<b>NET</b> 	If the indicator light is on, shows that the weight shown is Net weight (= Gross Weight - Tare).
 	If the indicator light is on, shows that the measurement is stable, ie. that it is within a range around a point for a time which conforms to the values defined during configuration.
 	If the indicator light is on, shows that the measurement performed is within the zero range (defined on the configuration screen).

## Output data

	Shows the current state of discrete outputs Q0 and Q1.
---	--

## Measurement data

<b>Flow:</b> <input type="text"/> -0.0088 kg	Indicates the current flow rate calculated by the weight unit per measurement period (20 ms).
 <b>Zero Tracking</b>	If the indicator light is on, shows that the zero tracking option is in use (configuration screen).
<b>Tare Value:</b> <input type="text"/> 0.00 kg	Indicates the current tare value.
<b>Zero Memory:</b> <input type="text"/> 0.0000 kg	Corresponds to the accumulation of offsets following zero reset commands since the last calibration.
 	Indicates that the tare is predefined.

All measurement data is also accessible in the form of PLC variables and may be displayed in animation tables.

Data displayed	Object address
Protected module Uncalibrated module	%MWxy.0.2:X8 %MWxy.0.2:X9
Weight Net weight indicator Stability indicator Zero range indicator	%IDxy.0.0 %IWxy.0.4:X8 %IWxy.0.4:X9 %IWxy.0.4:X10
State of discrete output Q0 State of discrete output Q1	%IWxy.0.4:X0 %IWxy.0.4:X1
Flow rate Tare value Offset memory Zero tracking indicator Predefined tare indicator	%IDxy.0.2 %IDxy.0.5 %IDxy.0.7 %IWxy.0.4:X11 %IWxy.0.4:X12

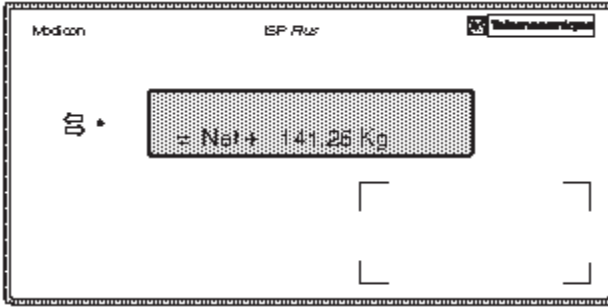
D

---

## 3.2 The display report

---

The data sent from the module to the display unit are metrological indicators (see installation documentation for TSX XBT H100).



This report indicates :

- whether the measurement is stable, by the sign =,
- whether the measurement concerns a Net weight (Net) or a Gross weight (if unspecified),
- whether the measurement is positive (+) or negative (-) or whether it is around zero (small 0).
- signed numerical data for the current weight,
- the name or symbol of the measurement unit of weight,
- (g for gram, kg for kilogram, lb for pound, oz for ounce and t for tonne).

**Note:**

The serial link is tested when the weighing module is powered up. The XBT must be connected to the TX ISP Y100 when the PLC is powered up.

All valid measurements are transmitted to the display every 100 ms in physical units with a fixed decimal point.

If the channel fault is set, the measurement is replaced by the following line of characters : '\_\_\_\_'. In the event of overload, it displays '>>>>' and in the event of underload, it displays '<<<<'.

The display unit continuously monitors data reception. If no data is received (because of disconnection, non-transmission by the module, etc) the 'Time Out' error is displayed. When the TSX XBT H100 is powered up it runs a test on its resources. All data received is checked during operation. In the event of problems, the "checksum" error is displayed.

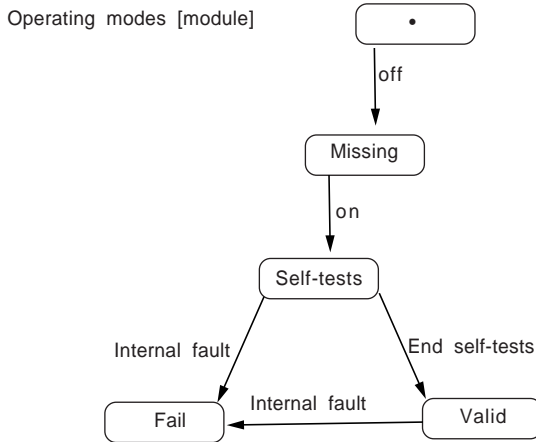
**Note:**

A space is reserved on the stamped identification plate of the TSX XBT H100 to satisfy the legal requirements for metrology.

### 3.3 Operating mode

When the module is powered up it runs self-tests (REPRO, RAM, display unit link, etc). If a fault is detected following these tests, the module goes to fallback mode and the outputs go to 0.

The same occurs during normal operation if an internal module malfunction (fault on RAM, watchdog, etc) is detected : the outputs go to 0 and dashes (—) appear on the screen.



On power outage, the machine parameters (Tare, Zero offset, etc) are saved. The operating parameters (Thresholds, Number of measurements used to calculate the flow rate, etc) are however lost.

---

D

## 4.1 Adjustment protection procedure

### Sealing :

All weighing instruments used for commercial transactions must be approved. The parameters associated with measurement must therefore be protected.

It must not be possible to introduce, via the interface of an instrument, instructions or data which can be used to :

- falsify the weighing results displayed,
- change an adjustment factor.

### Note:

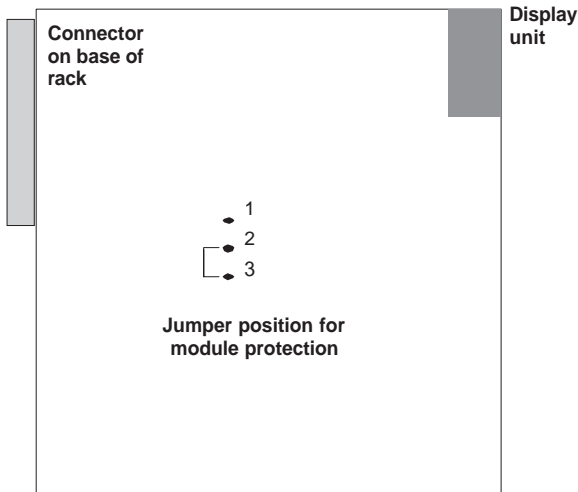
Protection by sealing is intended to guarantee measurement conformity, the accessible parameters therefore only affect module data use by the control system.

### Protection procedure

Once calibration and adjustment are complete, the module may be protected.

To do this, remove the module from the rack (the rack may remain powered up), then remove the casing.

The module protective jumper is located behind the module display unit. To activate the protection system, place the jumper across 2-3 as shown in the diagram.



Replace the module in its casing, then replace it in the PLC.

### Note:

To open the casing, use a TORX type screwdriver

## Effect of protection system on configuration parameters

There are two distinct types of data. Data maybe protected (once the module is sealed, this type of data can only be accessed in read-only mode) and open-access data (Read and Write modes).

The table below identifies the characteristics of this data depending on whether the protection system is operational.

Functions	unsealed	sealed
Task	Modifiable	Modifiable
Unite	Modifiable	Not modifiable
Maximum Range (MR)	Modifiable	Not modifiable
Scale division	Modifiable	Not modifiable
Overload threshold	Modifiable	Not modifiable
Filtering/ Coefficient	Modifiable	Not modifiable
Flow rate/ Calculation on n measurements	Modifiable	Modifiable
Tare/ Predefined	Modifiable	Modifiable
Data format	Modifiable	Not modifiable
Stability/ Extent of range	Modifiable	Not modifiable
Stability/ Time	Modifiable	Not modifiable
Zero/ Zero tracking	Modifiable	Not modifiable
Zero/ Extent of range	Modifiable	Not modifiable
Threshold check/ Active	Modifiable	Modifiable
Threshold check/ Direction	Modifiable	Modifiable
Threshold check/ Active outputs	Modifiable	Modifiable
Threshold check/ Cut-off points	Modifiable	Modifiable
Threshold check/ TF mask time	Modifiable	Modifiable
Load cell power supply	Modifiable	Not modifiable

Data word %IWxy.0.4:X4 (at 1) indicates whether or not the measurement is protected.

### Consequences of a protection system

- A sealed module which receives a different configuration from that memorized (before the power is switched off prior to moving the jumper) is refused.
- In this case, the module will appear to be missing in the PLC diagnostics, but transmits a weight to the display unit.

#### Comment :

Using the documentation enables a paper record of the configuration to be kept.

- The filtering of a sealed module cannot be modified.
- A sealed module will not accept a new request for calibration.



---

## 4.2 Metrology legal requirements and regulations

---

### 4.2-1 CE type approval

The assembly consisting of the weighing hopper + load cells + module may be considered to be a non-automatic weighing instrument. To qualify for this description, and to be authorized for use in commercial transactions, it has received CE type approval.

If it is only for use in internal processes, the display must have an identification plate showing :

Manufacturer's trademark	Max =
Type of instrument	sd =
Serial number	
'Not legal for trade'	

If it is for use in regulated operations (eg. commercial transactions), the display must have a stamped identification plate showing :

Manufacturer's trademark	Max =
Type of instrument	Min =
Serial number	sd =
SDM No. 97.06	

In addition, it must be initially checked when it leaves the factory and regularly checked on site by an approved organization. These checks must in general be performed every year, and are the responsibility of the owner of the instrument.

---

### 4.2-2 Approval for national model

#### Measurement and automatic operation equipment for weigher dosers and circuit weighers with batch totalizer

This non-automatic weighing instrument may be complemented with application-specific programs, the 'Weigher Doser' or 'circuit weigher with batch totalizer'. To qualify for this description, weighing instruments must have national approvals for use as measurement and control apparatus for the automatic operation of weigher dosers and circuit weigher with batch totalizer.

Manufacturers of weigher dosers or circuit weigher with batch totalizer can then very simply obtain approval for automatic weighing instruments.

Machine manufacturers have also to provide the identification plate and, if necessary, present the machine for initial checking.

---

### Approval for model with continuous totalizer

When associated with a weighing table, it is approved as a continuous totalizer.

When not for use in commercial transactions, the nameplate shows :

- Trademark	Qmax =
- type	dt =
- Serial no.	
'Not legal for trade'	

When the intended use involves commercial transactions, the identification plate shows :

- Trademark	Qmax =
- type	dt =
- Serial no	
Products weighed :	
- Max =	L =
- v =	d =

It must also be checked. The first phase of initial checking is performed in the factory on the finished instrument, not connected to its transporter, using a movement simulator : other phases are performed on the finished instrument.

---

#### 4.2-3 Class of apparatus

For normal precision, the apparatus covers a range of resolution from 100 to 1000 scale divisions inclusive.

For average precision, the apparatus covers a range from the minimum (500 scale divisions) up to 6000 scale divisions. These instruments may or may not be authorized for commercial transactions. If not, the legend 'NOT LEGAL FOR TRADE' must appear on the front panel of the apparatus.

## 5.1 Example of tare operation

**Important** : this section highlights the control of a weighing process by stressing the essential operations to be executed.

The operation involves a conversion to a NET weight (taring).

Bit %M101 is used to perform this action. When set, it causes the gross weight currently being measured to be taken into account as the weighing tare, then the display unit to switch to NET mode.

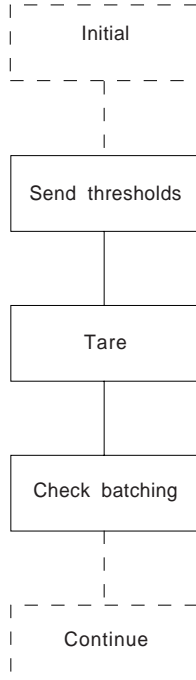
```
! (* Sealed weighing module, slot6 *)
! (* Awaiting tare conditions *)
      IF %M100 THEN
          IF NOT %MW6.0:X1 AND NOT %MW6.0.1:X1
          THEN
              SET %M101;
              RESET %M100;
          ELSE
              RETURN;
          END_IF;
      END_IF
!
(* Tare *)
      IF %M101 THEN
          (* send tare order *)
          IF NOT %MW6.0:X1 AND NOT %MW6.0.1:X1 AND NOT %M102 THEN
              %MW6.0.3:=0;
              SET %MW6.0.3:X4;
              WRITE_CMD %CH6.0;
              SET %M102;
          END_IF;
          (* tare completed and OK *)
          IF NOT %MW6.0:X1 AND NOT %MW6.0.1:X1 THEN
              %MW6.0.3:=0;
              RESET %M101;
              RESET %M102;
              SET %M103;
          ELSE
              (* tare refused => error *)
              IF NOT %MW6.0:X1 AND %MW6.0.1:X1 THEN
                  SET %M200;
                  %MW6.0.3:=0;
                  RESET %M101;
                  RESET %M102;
                  RESET %MW6.0.1:X1;
                  SET %M100;
              END_IF;
          END_IF;
      END_IF;
```

%MW6.0:X1	Exchange in progress
%MW6.0.1:X1	Exchange report
%MW6.0.3:X4	Tareorder
%MW6.0.3	Command order (tare, calibration, etc)

## 5.2 Example of batching

The following example uses a weighing module in slot 2 of the PLC. It shows a batching application divided into steps as in the flowchart below.

D



The program data used for the module:

%IW2.0.4:X0	Image of output Q0
%IW2.0.4:X1	Image of output Q1
%IW2.0.4:X5	Processing in progress indicator
%CH2.0	Data structure to send command
%MD2.0.8	High flow cut-off point Q0
%MD2.0.10	Low flow cut-off point Q1
%MW2.0:X1	Exchange in progress
%MW2.0:X2	Transmission in progress
%MW2.0.1:X1	Command accepted
%MW2.0.1:X2	Command accepted
%MW2.0.2:X2	Saturation of analog measurement system
%MW2.0.2:X7	Application fault
%MW2.0.3	Command order

**Main Program**

```

(* /////////// Send thresholds /////////// *)
%L100:
  IF NOT %M99 THEN
    JUMP %L120;
  END_IF;

(*Load and send thresholds *)
IF RE %M99 THEN
  %MD2.0.8:=%MD230;(* High flow cut-off point Q0*)
  %MD2.0.10:=%MD232; (* Low flow cut-off point Q1*)
  WRITE_PARAM %CH2.0;JUMP %L120;
END_IF;

(*Transmission in progress*)
IF %MW2.0:X2 THEN
  JUMP %L120;
END_IF;

(*command accepted*)
IF NOT %MW2.0.1:X2 THEN
  RESET %M99;
END_IF;

(*END INIT SCAN*)
%L120:

(* /////////// TARE PHASE (%MW100 =4) /////////// *)
%L260:
  IF %MW100<>4 THEN
    JUMP %L300;
  END_IF;

(*Tare request *)
IF %M72 THEN
  RESET %M72;
  %MW270.2:=4;
END_IF;

(*Command management *)
SR8; (* %MW270 contains information on tare command type 4 *)

(*Awaiting tare return*)
IF %MW270=-1 AND %MW271=-1 THEN
  %MW100:=5;
  SET %M72;
  JUMP %L800;
END_IF;

```

```
(* //////////////// BATCHING PHASE          (%MW100 = 5) //////////////// *)
%L300:
  IF %MW100<>5 THEN
    JUMP %L340;
  END_IF;

(*Validate thresholds *)
  IF %M72 THEN
    RESET %M72;
    %MW270:2:=8;
  END_IF;

(*Command management*)
  SR8;(* %MW270 contains information on Validate thresholds command type 8 *)

(*Awaiting command return*)
  IF %MW270>=0 OR %MW271>=0 THEN
    JUMP %L800;
  END_IF;

(*Check outputs to continue*)
  IF NOT %IW2.0.4:X0 AND NOT %IW2.0.4:X1 THEN
    %MW100:=6;
    SET %M72;
    JUMP %L800;
  END_IF;

(*PHASE 6   continue   *)
%L340:
  IF %MW100<>6 THEN
    JUMP %L380;
  END_IF;
%L800:
```

#### SUBROUTINE SR8 :

```
(* Send request for module*)
  IF %MW270>=0 THEN (* %MW270 contains information on order to be executed *)
    %M0:16:=0;
    SET %M0[%MW270];
    %MW2.0.3:=%M0:16;
    %MW271:=%MW270;
    %MW270:=-1;
    WRITE_CMD %CH2.0;
    RET;
  END_IF;

(*Command in progress ? *)
  IF %MW2.0:X1 OR %IW2.0.4:X5 THEN
    RET;
  END_IF;

(*command accepted ? *)
  IF NOT %MW2.0.1:X1 AND NOT %MW2.0.2:X7 THEN
    %MW270:2:=-1;
  ELSE
    %MW270:=%MW271;
  END_IF;
```

## 6.1 Technical characteristics

<b>Weighing module</b>	
Number of modules per PLC (*)	The ISP Y100 module can be installed in any available slot in a Premium PLC (TSX, PMX or PCX) configuration, provided the following maximum limits are respected : <ul style="list-style-type: none"> <li>• 8 "application-specific" channels with a TSX / TPMX P57 102 or TPCX 57 1012 processor.</li> <li>• 24 "application-specific" channels with a TSX / TPMX P57 2•2 processor.</li> <li>• 32 "application-specific" channels with a TSX / TPMX P57 302 or P 57 352 processor.</li> <li>• 48 "application-specific" channels with a TSX / TPMX P57 4•2 processor.</li> </ul>
Number of weighers per module	1
Module power consumption	At 5V : 330 mA maximum At 24V : 130 mA maximum
<b>Analog measurement system</b>	
Electrical range	0 to 25 mV
Minimum measurable range	4.5 mV
Maximum measurable range	25 mV
Converter resolution	20 bits (1 048 576 pts)
Limitation on use	50 000 pts
Conversion speed	50 measurements/second
Zero drift	< 0.2 mV/°C
Gain drift	< 10 ppm/°C
Non linearity	< 20 ppm(PE)
Rejection in serial mode 50 Hz	> 120 dB
Maximum length of measurement cable (1 to 8 load cells)	100m for a 0.4mm <sup>2</sup> cable 200m for a 0.6mm <sup>2</sup> cable
<b>Cell powered by module</b>	
Supply voltage	10 VDC
Load impedance	> 43 Ω (8 x 350 Ω cells)

(\*) The ISP Y100 module is considered to be a module with 2 "application-specific" channels.

<b>Discrete outputs</b>	
Number of channels	2
Nominal supply voltage	24 V
Isolation voltage	1500V rms
Maximum current	500 mA
Protection	- against short-circuits - against inversions, via reverse diode Place a fuse on the + 24 V of the preactuators
<b>Serial link</b>	
Type	RS485 not isolated
Transmission speed	9600 bauds
Format	1 start bit, 8 data bits and 1 stop bit
Maximum permitted distance	100 meters maximum

## 6.2 Standards

<b>Standards for TSX ISP Y100 module</b>	
NFEN45501	Yes
IEC1131-2	Yes
<b>Temperature</b>	
Operating	from 0°C to 55°C
Storage	from 40°C to 70°C
<b>Standards for TSX XBT H100 display unit</b>	
NFEN45501	Yes
IP65	Degree of protection conforms to IEC529 and NFC20-010
<b>Temperature</b>	
Operating	from 0°C to 55°C
Storage	from 40°C to 70°C

## 6.3 Approvals

EC approval : Class III, 6,000 divisions, registered as 5 DM 9706.



---

## 6.4 Recommendations for installing an analog measurement system

---

### Load distribution

The quality of the measurement provided by the module may be considerably reduced if the appropriate precautions for mounting and installing load cells are not observed. The following recommendations, while not intended to replace genuine expertise, will make you aware of certain precautions which need to be taken.

In an analog measurement system, the load cells can tolerate the following weights :

- the maximum weight to be measured (or maximum range),
- the weight of the load receiver and its structures (or meteorological tare).

This total weight is divided between 1, 2, 3, 4, 6, or even 8 cells. The design of the mechanical structures, the shape of the load receptacle and the load distribution on or in the receptacle all may lead to an uneven distribution of the total weight between the cells (except of course in the case of a single load cell).

The load cells must therefore be sized to be capable of tolerating the total weight (maximum range + tare) which they will have to support (see load cell selection guide).

### Restrictions on the load receptacle

As the deflection of a load cell is very small (a few tenths of a millimeter), any restriction on the load receptacle or friction on the static framework will produce an error in the weight measurement and make it impossible to adjust the module correctly.

### Mechanical mounting of the load cells

Load cells which are subject to traction or compression must be used vertically, observing the direction in which they act (traction or compression).

The maximum permitted tolerance for vertical mounting depends on the mounting and the required precision.

### Protecting the load cells against electrical interference

It is recommended that each cell is equipped with a grounding strip to act as an electrical shunt and to protect the load cells against any currents which may be circulating in the metal framework (currents from the ground or solder point, electrostatic discharges, etc).

This strip must be long enough to avoid any mechanical restriction : it must be positioned in the immediate vicinity of the load cells, between the static framework and the load receptacle.

---

### **Splashing and corrosive products**

The weighing cells are of dust and damp proof construction, however, it is recommended that they are protected against splashing, corrosive products and direct sunlight.

**D**

### **Preventive maintenance of the installation and accessories**

The weighing module does not require any special maintenance. The weighing cells must however be cleaned periodically if they are used in hostile environments.

It is recommended that the load receptacle is checked and serviced periodically to ensure that it is in good mechanical order.

- Clean the receptacle and its structures as deposits of products or other materials may lead to significant variations in the tare.
- Check that the load cells are vertical.
- Check the condition of the cells and actuators depending on their length of service.
- Etc

#### **Note:**

Statistics show that 90% of malfunctions detected on weighing/batching installations can be attributed to the installation itself (defective limit switches, mechanical faults, etc) and not to the electronic control unit.

**A**

Accessing measurements	1/16
Active	1/10
Active outputs phase 1	1/10
Adapting the process	2/21
Adjusting measurement filtering	2/26
Adjusting outputs	2/26
Adjusting the number of measurements used for	
the Adjusting thresholds	2/26
Adjustments	2/21
Analog measurement system	6/1
Approval for model with continuous totalizer	4/4
Approval for national model	4/3
Approvals	6/2

**B**

Batching	5/2
----------	-----

**C**

Calibration	2/7
Calibration in progress	2/11
CE type approval	4/3
Channel fault	1/19
Channel status	1/21
Class of apparatus	4/4
Configuration	1/3
Configuration parameters	1/4
Constants	1/14
Customizing the measurement	2/21
Cut-off points	1/11, 2/24

**D**

Data display	1/2
Data exchanges with the processor	1/2
Data format	1/6
Default configuration	1/12
Direction	1/10
Disable thresholds	2/20
Discrete outputs	6/2
Display	3/4
Display of manual tare	2/18
Downweighing	1/10

**E**

Enable thresholds	2/19
Explicit exchange objects	1/14
Extent of the range	1/5, 1/6

**F**

Filtering	1/7, 2/4
Filtering coefficient	2/24
Flow rate	1/9, 1/17, 2/4
Forced calibration	2/10
Format	2/28

**G**

Gross weight	1/21
--------------	------

**I**

Implicit exchange objects	1/14
Instability indication	1/18

**L**

Load cell power supply	1/7, 6/1
Load distribution	6/3

**M**

Maintenance	6/4
Manual tare	2/18
Manual tare value	2/24
Mask time	1/11, 2/25
Maximum range	1/4, 2/27
Measurement data	2/3, 3/2
Measurement processing	1/1
Measurement status word	1/17
Measurement verification	1/1
Module fault	1/19
Module information	3/1
Module status	1/20
Mounting the load cells	6/3

**N**

Non-automatic weighing instrument	4/3
Number of measurements used to calculate the flow	2/25

**O**

Offset memory	1/18
Operating mode	3/5
Output data	3/2
Output logic	2/24
Output management	1/2
Outputs	2/3
Overload threshold	1/5, 2/28

**P**

Parameter adjustment	2/4
Power-up	3/5
Protecting the load cells	6/3
Protection	4/1, 4/2

**R**

READ_PARAM	2/23
RESTORE_PARAM	2/23
Restrictions on the load receptacle	6/3
Return to gross weight	2/17

**S**

Save calibration	2/11
SAVE_PARAM	2/23
Scale division	1/4, 2/27
Sealing	4/1
Serial link	6/2
Stability	1/6, 2/28
Standard load	2/9, 2/11
Standards	6/2

**T**

Tare	1/9, 2/4
Tare operation	2/13, 5/1
Tare value	1/18
Temperature	6/2
Threshold check	1/10, 2/4, 2/20
Time	1/6

**U**

Unit	1/4, 2/27
Use of outputs	2/28

**V**

Value of the standard load	2/11
----------------------------	------

**W**

Weighing	1/10
Weighing module	1/1, 6/1
Weight	2/3
Weight data	3/2
WRITE_CMD	2/6
WRITE_PARAM	2/22

**Z**

Zero	1/5, 2/28
Zero load	2/11
Zero load calibration	2/9
Zero reset	2/15
Zero tracking	1/5