

# PL7 Micro/Junior/Pro

## Micro PLC Applications

### Applications Base

TLX DS 37 PL7 xx eng







---

# Table of Contents



<b>About the book</b>	<b>13</b>
<b>Part I Shared application-specific functions</b>	<b>15</b>
Presentation	15
<b>Chapter 1 Common application specific functions: General</b>	<b>17</b>
Presentation	17
General Presentation	18
Configuration of an application-specific function.	20
Adjustment of an application-specific function	21
Debugging an application-specific function.	22
<b>Chapter 2 Objects associated with specific applications</b>	<b>23</b>
Presentation	23
2.1 Addressing of language objects associated with specific applications	25
Presentation	25
Addressing of built-in application-specific interfaces.	26
Addressing in-rack Input/Output module objects.	28
Addressing of language objects associated with AS-i bus	30
2.2 Implicit exchanges.	32
Implicit exchanges	32
2.3 Explicit exchanges.	34
Presentation	34
Explicit exchanges: General	35
READ_STS: Reading status words	37
WRITE_CMD: Writing command words	39
READ_PARAM: Reading adjustment parameters	40
WRITE_PARAM: Writing adjustment parameters	41
SAVE_PARAM: Saving adjustment parameters	42
RESTORE_PARAM: Restoring adjustment parameters	44
Exchange and report management.	45
2.4 Presymbolization.	48
Presentation	48
Presymbolized objects	49
Automatic symbolization of objects associated with a channel.	50



---

<b>Chapter 3</b>	<b>Application-specific instructions</b>	<b>51</b>
	Presentation	51
	Application-specific instructions	52
	Accessing a specific function, method or procedure type instruction	53
<b>Chapter 4</b>	<b>Appendices</b>	<b>55</b>
	Presentation	55
	Reminders concerning the configuration editor	56
	The PL7 toolbar	57
	The PL7 status bar	58
	How to declare a module in a PLC rack	59
	Confirming the configuration of a module	60
	Globally reconfiguring an application	61
	Application-specific fault processing (in-rack modules) by program	62
<b>Part II</b>	<b>Discrete application</b>	<b>63</b>
	At a Glance	63
<b>Chapter 5</b>	<b>General introduction to the discrete application -specific function</b>	<b>65</b>
	Introduction to the discrete application	65
<b>Chapter 6</b>	<b>Discrete application configuration</b>	<b>67</b>
	At a Glance	67
6.1	Configuring a discrete module: General	69
	At a Glance	69
	Description of the configuration screen for a discrete module	70
	How to access the configuration parameters of a discrete module	72
	Modifying the configuration parameters of a discrete module's: General	74
6.2	Discrete input/output channel parameters	76
	At a Glance	76
	Discrete inputs parameters	77
	Discrete output parameters	79
6.3	Configuration of discrete parameters	81
	Presentation	81
	How to modify the Task parameter of a discrete module	82
	How to modify the Monitoring the external supply parameter for a discrete module	83
	How to modify the Functions parameter of a discrete module	84
	How to modify the Type of inputs parameter of a discrete module	86
	How to modify the Network frequency parameter of a discrete module	87
	How to modify the Filtering time parameter for 24 DCV inputs of a discrete module	88
	How to modify the Fallback mode parameter of a discrete module	89
	How to modify the Reactivation of outputs parameter of a discrete module	90

---



	How to parameterize the RUN/STOP input of a discrete module . . . . .	91
	How to parameterize the save program and %MW input of a discrete module . . . . .	92
	How to parameterize the alarm output of a discrete module. . . . .	93
<b>Chapter 7</b>	<b>Debugging discrete modules . . . . .</b>	<b>95</b>
	At a Glance . . . . .	95
	Introduction to the debug function of a discrete module.. . . .	96
	Description of the debug screen for a discrete module. . . . .	97
	How to access the channels debug screen of a discrete module. . . . .	99
	How to access the Diagnostics function for a discrete module . . . . .	100
	Accessing the forcing/unforcing function. . . . .	101
	How to access the SET and RESET commands . . . . .	102
	How to access the reactivation of outputs command . . . . .	103
	Maintain outputs for a discrete module . . . . .	104
<b>Chapter 8</b>	<b>Bits and words associated with discrete specific applications . . . . .</b>	<b>105</b>
	Presentation . . . . .	105
8.1	Addressing of discrete I/O module objects . . . . .	107
	Addressing discrete input/output module objects . . . . .	107
8.2	Language objects associated with the discrete specific application . . . . .	108
	Presentation . . . . .	108
	Implicit exchange objects . . . . .	109
	Exchange management: Module %MWx.MOD.0:Xj or channel %MWx.i.0:Xj exchanges in progress . . . . .	110
	Exchange management: Module %MWx.MOD.1:Xj or channel %MWx.i.10:Xj report . . . . .	111
	Explicit exchange objects: General . . . . .	112
	Explicit exchange objects: Status Module %MWx.MOD.2:Xj . . . . .	113
	Explicit exchange object: Command Word %MWxMOD.3:Xj . . . . .	114
<b>Part III</b>	<b>AS-i Bus . . . . .</b>	<b>115</b>
	At a Glance . . . . .	115
<b>Chapter 9</b>	<b>General introduction to the AS-i Bus . . . . .</b>	<b>117</b>
	At a Glance . . . . .	117
	Introduction to the AS-i Bus . . . . .	118
	Architecture of the TSX SAZ 10 module . . . . .	120
	Structure of an AS-i slave . . . . .	122
	How to declare an AS-i communication module in the PLC rack . . . . .	123
	How to access the AS-i Bus configuration . . . . .	124
<b>Chapter 10</b>	<b>AS-i bus configuration . . . . .</b>	<b>125</b>
	At a Glance . . . . .	125
	Description of the configuration screen for an AS-i communication module . . . . .	126
	How to define a slave device on the AS-i bus . . . . .	128



	How to modify the software configuration of the AS-i Bus . . . . .	130
	How to access the description of an AS-i slave . . . . .	131
	How to define a new slave profile in the standard AS-I catalogue . . . . .	133
	How to modify AS-i slave general parameters: Automatic addressing . . . . .	135
	How to modify AS-i slave general parameters: Fallback mode . . . . .	136
<b>Chapter 11</b>	<b>Debugging the AS-i bus . . . . .</b>	<b>137</b>
	At a Glance . . . . .	137
	Introduction to the Debug function . . . . .	138
	Description of the debug screen for an AS-i module . . . . .	139
	How to access functionality in the module diagnostics and channel diagnostics for an AS-i device . . . . .	141
	Displaying the slaves' status . . . . .	143
	How to access adjustment of an AS-i device's parameters . . . . .	144
	How to access the AS-i channels' forcing/unforcing function . . . . .	145
	How to access the SET and RESET commands of the AS-i channels . . . . .	146
	Automatic replacement of a faulty AS-i slave . . . . .	147
	How to insert a slave device into an existing AS-i configuration.. . . .	148
<b>Chapter 12</b>	<b>Bits and words associated with the AS-i function . . . . .</b>	<b>149</b>
	At a Glance . . . . .	149
12.1	Addressing objects associated with the AS-i function . . . . .	151
	Addressing language objects associated with slave devices connected on the AS-i Bus. . . . .	151
12.2	Language objects associated with the AS-i function . . . . .	152
	At a Glance . . . . .	152
	Implicit exchange objects associated with the AS-i function . . . . .	153
	Managing exchanges: Module %MW4.MOD.0:Xj or channel %MW4.0.0:Xj exchanges in progress. . . . .	154
	Managing exchanges: Module %MW4.MOD.1:Xj or channel %MW4.0.1:Xj report . . . . .	155
	Explicit exchange objects: General . . . . .	156
	Explicit exchange objects: Channel status %MW4.0.2:Xj to %MW4.0.22:Xj . . .	157
	Explicit exchange object: Channel command %MW4.0.23:Xj . . . . .	159
	Explicit exchange objects: Adjustment parameters %MW4.0.24 to %MW4.0.39	160
	Explicit exchange objects: Status %MW4.MOD.2:Xj . . . . .	161
<b>Chapter 13</b>	<b>AS-i operating mode . . . . .</b>	<b>163</b>
	At a Glance . . . . .	163
	AS-i operating mode: General . . . . .	164
	AS-i protected mode . . . . .	166
	AS-i wiring test mode . . . . .	167
	AS-i offline operating mode . . . . .	168
<b>Chapter 14</b>	<b>AS-i performance . . . . .</b>	<b>169</b>
	AS-i bus performance . . . . .	169



---

<b>Part IV</b>	<b>Analog application . . . . .</b>	<b>171</b>
	At a Glance . . . . .	171
<b>Chapter 15</b>	<b>The analog application. . . . .</b>	<b>173</b>
	Introduction to the analog application on Micro. . . . .	173
<b>Chapter 16</b>	<b>Built-in analog interface. . . . .</b>	<b>175</b>
	Chapter overview . . . . .	175
16.1	Built-in analog interface. . . . .	177
	Introducing the built-in analog interface . . . . .	177
16.2	Input processing . . . . .	179
	At a Glance . . . . .	179
	Timing of measurements. . . . .	180
	Under/overshoot monitoring on inputs . . . . .	181
	Sensor link monitoring. . . . .	182
	Measurement filtering . . . . .	183
	Displaying measurements. . . . .	184
16.3	Output processing . . . . .	185
	Output characteristics . . . . .	185
<b>Chapter 17</b>	<b>Analog input modules TSX AEZ 801 / TSX AEZ 802 . . . . .</b>	<b>187</b>
	Chapter overview . . . . .	187
	Introduction to the TSX AEZ 801/TSX AEZ 802 modules. . . . .	188
	Timing of measurements. . . . .	189
	Range selection and input overflow monitoring. . . . .	190
	Sensor link checking on TSX AEZ 802 . . . . .	191
	What happens to the module in the event of an overload. . . . .	192
	Measurement filtering . . . . .	193
	Displaying measurements. . . . .	196
<b>Chapter 18</b>	<b>Analog input modules TSX AEZ 414. . . . .</b>	<b>197</b>
	Chapter overview . . . . .	197
	Introducing the TSX AEZ 414 module. . . . .	198
	Timing of measurements. . . . .	200
	Range selection and input overshoot monitoring . . . . .	201
	Sensor link monitoring. . . . .	204
	What happens to the module in the event of an overload. . . . .	205
	Measurement filtering . . . . .	206
	Displaying measurements. . . . .	207
<b>Chapter 19</b>	<b>Analog output module TSX ASZ 401 . . . . .</b>	<b>209</b>
	Chapter overview . . . . .	209
	Introducing the TSX ASZ 401 module . . . . .	210
	Writing and updating outputs. . . . .	211
	Fault handling . . . . .	212
<b>Chapter 20</b>	<b>Analog output module TSX ASZ 200 . . . . .</b>	<b>213</b>

---



---

	Chapter overview . . . . .	213
	Introducing the TSX ASZ 200 module . . . . .	214
	Writing and updating outputs . . . . .	215
	Fault handling . . . . .	216
	Fault Processing . . . . .	217
<b>Chapter 21</b>	<b>TSX AMZ 600 Analog Module . . . . .</b>	<b>219</b>
	Introduction to the Chapter . . . . .	219
	Introduction to the TSX AMZ 600 Module . . . . .	220
	Measuring Speed. . . . .	222
	Selection of Ranges and Monitoring of Input Overshoots . . . . .	223
	Monitoring of Sensor Link on TSX AMZ 600 . . . . .	224
	Module Behavior in Event of Overload . . . . .	225
	Filtering of Measurements . . . . .	226
	Display of Current Values . . . . .	229
<b>Chapter 22</b>	<b>Configuring the analog application . . . . .</b>	<b>231</b>
	At a Glance . . . . .	231
22.1	Reminder about the configuration editor . . . . .	233
	At a Glance . . . . .	233
	Accessing the configuration editor. . . . .	234
	Selection of Modules . . . . .	235
22.2	Accessing the analog application parameters . . . . .	237
	At a Glance . . . . .	237
	Accessing Parameterization of the Integrated Analog Interface . . . . .	238
	Accessing Parameterization of an Analog Module . . . . .	239
<b>Chapter 23</b>	<b>Configuration of analog channels . . . . .</b>	<b>241</b>
	At a Glance . . . . .	241
23.1	Channel configuration function - General . . . . .	243
	At a Glance . . . . .	243
	Default Configuration of Channels. . . . .	244
	Display of Channel Parameters . . . . .	246
23.2	Modifying an input channel's parameters . . . . .	250
	At a Glance . . . . .	250
	Modification of the Scanning Cycle . . . . .	251
	Modification of the Task Assigned to the Module Inputs . . . . .	252
	Modification of the Input Range . . . . .	253
	Modification of the Display Format. . . . .	254
	Modification of the Filtering Value . . . . .	256
23.3	Modifying an output channel's parameters . . . . .	257
	At a Glance . . . . .	257
	Modification of the Fallback Mode . . . . .	258
	Modifying the task to which the output is assigned . . . . .	259
	Modification of the Output Range (TSX ASZ 200 and TSX AMZ 600) . . . . .	260

---



<b>Chapter 24</b>	<b>Debugging function</b>	<b>261</b>
	At a Glance	261
	Introduction to the Debugging function	262
	Display of Channel Parameters	263
	Module Diagnostics Display	265
	Deletion of Module Channel Forcing	267
	Display of Detailed Channel Diagnostics	268
	Modification of the Filtering Value	269
	Channel Forcing / Delete Forcing	270
<b>Chapter 25</b>	<b>Bits and words associated with the analog application</b>	<b>273</b>
	At a Glance	273
	Implicit exchange objects associated with the analog application	274
	Explicit exchange objects associated with inputs/outputs	275
	Configuration objects associated with the analog application	277
<b>Part V</b>	<b>Operator Dialog functions</b>	<b>281</b>
	Introduction	281
<b>Chapter 26</b>	<b>General presentation of the Operator Dialog functions</b>	<b>283</b>
	General presentation	283
<b>Chapter 27</b>	<b>Built-in DOP functions</b>	<b>285</b>
	Introduction	285
27.1	Description of the parameters common to the different DOP functions	287
	Introduction	287
	General	288
	Parameters Zone: Terminal address	289
	Parameters field: Data to be sent	291
	Parameter field: Data to be received	293
	Parameters field: Report	294
	Message field	298
	Field zone	300
27.2	Description of the built-in DOP functions	302
	Introduction	302
	List of the built-in DOP functions	303
	SEND_MSG function	304
	GET_MSG function	306
	ASK_MSG function	309
	SEND_ALARM function	311
	DISPLAY_MSG function	314
	DISPLAY_GRP function	315
	DISPLAY_ALRM function	317
	ASK_VALUE function	320
	GET_VALUE function	321
	CONTROL_LEDS function	323



	ASSIGN_KEYS function . . . . .	326
	PANEL_CMD function . . . . .	329
	ADJUST function . . . . .	332
<b>Chapter 28</b>	<b>Appendices . . . . .</b>	<b>341</b>
	Introduction . . . . .	341
28.1	Precautions for DOP use . . . . .	343
	Precautions for DOP use . . . . .	343
28.2	Description of the built-in DOP functions "Data to send" parameter coding . . . . .	344
	Introduction . . . . .	344
	PLC status message display: SEND_MSG function . . . . .	345
	PLC checked status message entry: ASK_MSG and GET_MSG function . . . . .	348
	PLC alarm message display: SEND_ALARM function . . . . .	353
	Display of status, alarm or a group of messages contained in the CCX 17 memory: ASK_VALUE, DISPLAY_MSG, GET_VALUE, DISPLAY_ALARM and DISPLAY_GRP functions. . . . .	357
	Display of luminous column LEDs: CONTROL_LEDS function . . . . .	358
	Configuring command keys: ASSIGN_KEYS function. . . . .	359
	Generic send command: PANEL_CMD function . . . . .	361
<b>Part VI</b>	<b>Process control functions . . . . .</b>	<b>363</b>
	At a Glance . . . . .	363
<b>Chapter 29</b>	<b>General on the PID . . . . .</b>	<b>365</b>
	Introduction . . . . .	365
	General introduction. . . . .	366
	Principal of the regulation loop. . . . .	367
	Development methodology of a regulation application . . . . .	368
<b>Chapter 30</b>	<b>Description of the regulation functions. . . . .</b>	<b>369</b>
	Introduction . . . . .	369
	Programming a regulation function . . . . .	370
	PID Function . . . . .	371
	Programming the PID function. . . . .	373
	PWM Function . . . . .	378
	Programming the PWM function . . . . .	380
	SERVO Function . . . . .	382
	Programming the SERVO function . . . . .	386
	Performance of the functions in the operating mode . . . . .	389
<b>Chapter 31</b>	<b>Operator dialogue on CCX 17 . . . . .</b>	<b>391</b>
	Introduction . . . . .	391
	Dialog operator on the CCX 17 . . . . .	392
	Selecting a loop . . . . .	394
	Controlling a loop. . . . .	395
	Adjusting a loop . . . . .	396



---

	PID_MMI Function: programming . . . . .	397
	Performance of the PID_MMI function according to the PLC and CCX 17's operating modes . . . . .	401
<b>Chapter 32</b>	<b>Example of application. . . . .</b>	<b>403</b>
	Introduction . . . . .	403
	Description of the application example . . . . .	404
	Configuration of the example . . . . .	406
	Programming the example . . . . .	409
<b>Chapter 33</b>	<b>Appendices . . . . .</b>	<b>413</b>
	At a Glance . . . . .	413
	PID parameter adjustment method . . . . .	414
	Role and influence of PID parameters . . . . .	417
<b>Glossary</b>	<b>. . . . .</b>	<b>421</b>
<b>Index</b>	<b>. . . . .</b>	<b>423</b>

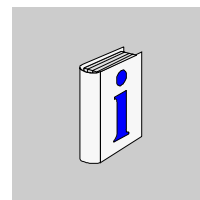






---

## About the book



---

### At a Glance

**Document Scope** This manual describes the software installation for the applications (excluding communication applications) on TSX 37 using PL7 software.

**Validity Note** The updating of this publication takes into account the functionalities of PL7 V4.2. However it enables the installation of previous versions of PL7.

**Related Documents**

Title of Documentation	Reference Number
Hardware installation manual	TSX DM 37 50 E

**User Comments** We welcome your comments about this document. You can reach us by e-mail at [TECHCOMM@modicon.com](mailto:TECHCOMM@modicon.com)

---

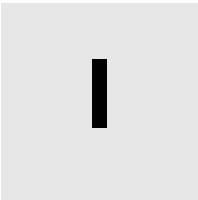






---

# Shared application-specific functions



---

## Presentation

**Subject of this part**

This part gives a general overview of the acknowledgment of specific applications by the PL7 software.

**What's in this part?**

This Part contains the following Chapters:

Chapter	Chaptername	Page
1	Common application specific functions: General	17
2	Objects associated with specific applications	23
3	Application-specific instructions	51
4	Appendices	55







---

# Common application specific functions: General

1

---

## Presentation

**Subject of this chapter** This chapter presents the common application specific functions of the PL7 software.

**What's in this Chapter?** This Chapter contains the following Maps:

Topic	Page
General Presentation	18
Configuration of an application-specific function.	20
Adjustment of an application-specific function	21
Debugging an application-specific function	22

---



## General Presentation

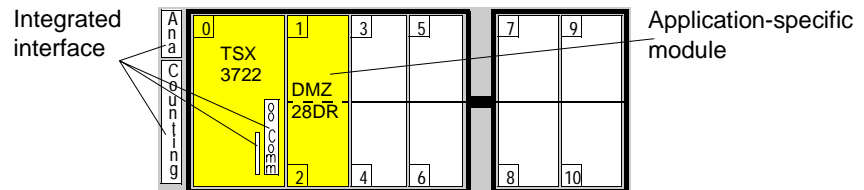
### Introduction

The PL7 software allows for the installation of application-specific functions in software form (DISCRETE, analog, process control, etc.).

The application-specific functions are automated functions which interface with the control part (the PLC program) and the operational part (sensors, actuators and man/machine interfaces).

An application-specific function on TSX Micro is presented, depending on the application, in the form of:

- a module (e.g.: counting application),
- a built-in interface (e.g.: communication port).





## Software installation principle

The table below gives a short description of the general principle for the installation of an application-specific function. This principle will be covered again later in this manual in more specific detail for each application.

Mode	Phase	Description
Local	Configuration	Configuration of the module or built-in interface
Local or online	Symbolization	Symbolization of variables associated with the application-specific function.
	Programming	Programming of the function to be carried out using: <ul style="list-style-type: none"> <li>● bit and word objects associated with the module,</li> <li>● application-specific instructions.</li> </ul>
	Transfer	Transferring the application to the PLC.
	Documentation	Printing of the different information about the application.
Online	Debug	Debugging the application (I/O control, fault identification).

**Note:** The order shown above is given as an indication only; the PL7 software allows the editors to be used interactively in the order required (however, the data or program editor cannot be used if the I/O modules have not been configured first).

## Software installation resources

An application-specific function is installed using:

- standard PL7 tools:
  - pull-down menus,
  - status bars, tool bars,
  - editors,
  - ...
- application-specific screens,
  - configuration screens,
  - adjustment screens,
  - debug screens,
- language objects giving program-based access to inputs and outputs of the module or built-in interface.
- instructions relative to the application-specific function, where applicable.

**Note:** The different screens as well as the objects associated with an application-specific module are accessible via the software as soon as the module is declared in the configuration, without it even being necessary to write a program line.



## Configuration of an application-specific function.

### Introduction

The **Configuration** function makes it possible to define the operating characteristics of the module or the application-specific interface.

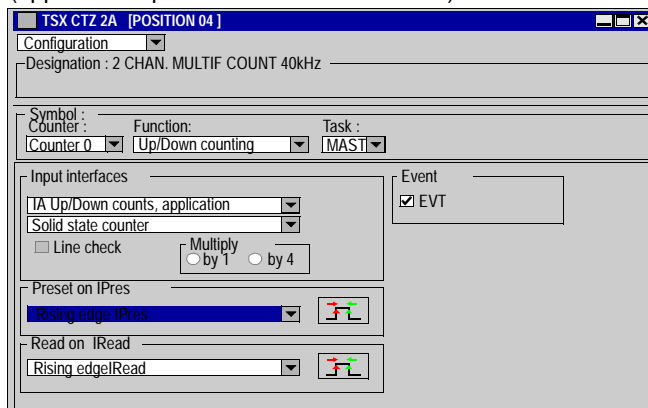
This function is carried out from the PL7 configuration editor:

- in local mode,
- in online mode when the application is stored in non-write protected RAM (limited to certain parameters).

**Note:** The configuration parameters are not modifiable by program.

### Illustration

The screen below is an example of a configuration screen of an application-specific module (application-specific module TSY CTZ 2A).



Here, the operating characteristics are:

- the selection of a function associated with a channel: up counting, down counting or up/down counting,
- selection of the task which updates the inputs/outputs of the module,
- ...

### Confirmation

The characteristics defined in the configuration screen must be subject to a global confirmation of the application. This may be done:

- in local mode, so that the modifications are taken into account,
- in online mode in order to:
  - update the configuration parameters in the PLC,
  - reconfigure the channel of the module with its new parameters (the adjustment parameters return to their initial value).



## Adjustment of an application-specific function

### Introduction

The **Adjustment** function allows the operating parameters of the application-specific module or interface to be displayed and modified, when they are modifiable.

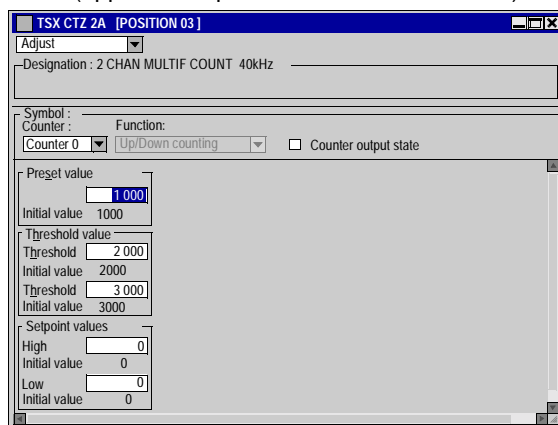
This function is carried out from the PL7 adjustment editor:

- in local mode, in order to define the initial parameters (value of the parameters during setup or on a cold restart),
- in online mode, in order to define the current parameters (values lost upon cold restart if they have not been saved in advance).

**Note:** The adjustment parameters are modifiable by program.

### Illustration

The screen below is an example of an adjustment screen of an application-specific module (application-specific module TSY CTZ 2A).



Here, the operating parameters are:

- the threshold values,
- the setpoint values,
- the counter output states.

### Confirmation

The characteristics defined in the adjustment screen must, depending on the mode, be subject to:

- local: a global confirmation of the application,
- online: confirmation of the modifications in order to update the current parameters in the PLC and on the module channel.



## Debugging an application-specific function

### Introduction

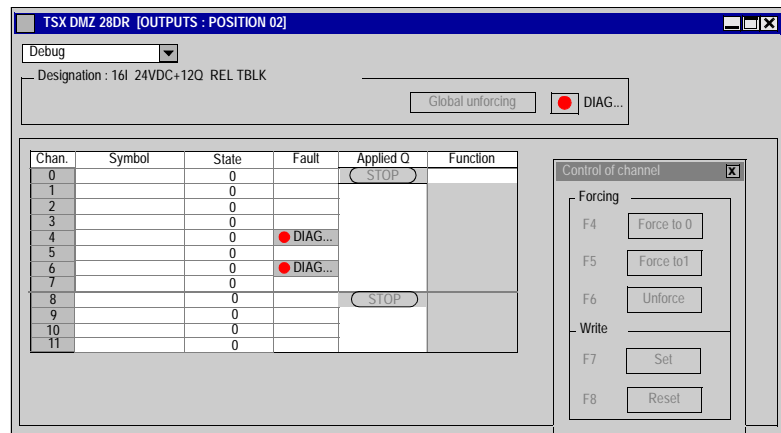
The **Debug** function of the module or the built-in application-specific interface provides the tools to help with debugging of the application-specific function such as:

- display of module channel status,
- display of possible faults,
- control of language objects,
- and, in the event of a fault, access to the module or channel diagnostics,
- ...

This function is carried out in **online mode**, with the PLC in STOP or in RUN, from the PL7 debug editor.

### Illustration

The screen below is an example of a debug screen of an application-specific module (application-specific module TSX DMZ 28DR).



Here the debug tools are:

- forcing of the output channels to 0 or 1,
- access to module and channel diagnostics.



---

# Objects associated with specific applications



---

## Presentation

### Subject of this chapter

This chapter presents the addressing and the exchange modes for language objects associated with PL7 specific applications.

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
2.1	Addressing of language objects associated with specific applications	25
2.2	Implicit exchanges	32
2.3	Explicit exchanges	34
2.4	Presymbolization	48







---

# 2.1                    Addressing of language objects associated with specific applications

---

## Presentation

**Subject of this section**                    This section presents the addressing of language objects associated with application-specific modules.

**What's in this Section?**                    This Section contains the following Maps:

Topic	Page
Addressing of built-in application-specific interfaces	26
Addressing in-rack Input/Output module objects	28
Addressing of language objects associated with AS-i bus	30



## Addressing of built-in application-specific interfaces

---

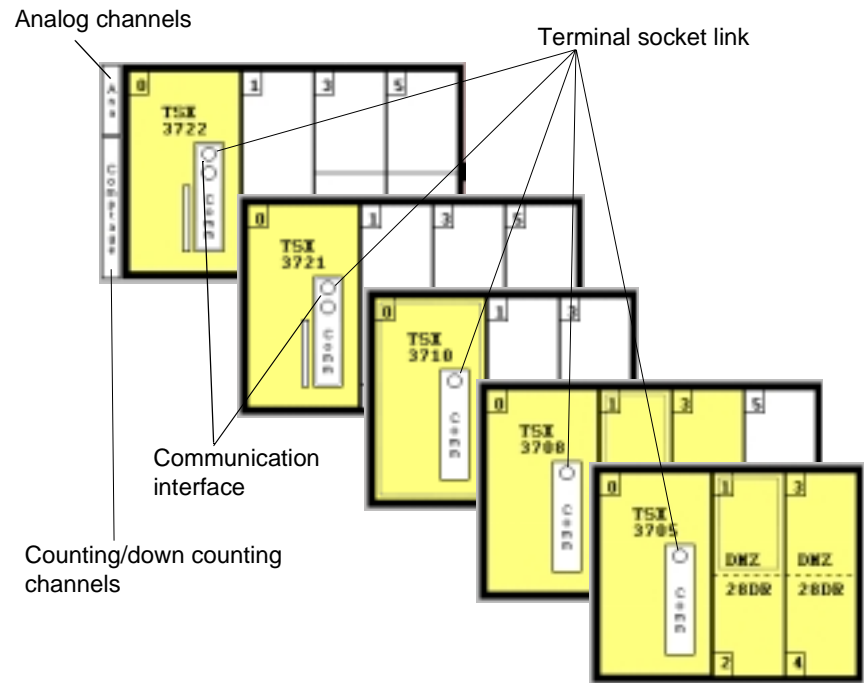
### At a Glance

The TSX Micro range provides up to 4 built-in application-specific interfaces, according to PLC type:

- terminal socket link,
- communication interface,
- analog channels,
- counting/down counting channels.

### Illustration

The illustration below locates each in-built interface for different PLC types.





**Channel number** The table below provides the addressing of the different built-in application-specific channels.

PLC	Terminal socket	Communication interface	analog input channels	Analog output channel	Counting/down counting channels
TSX 37-05/08/10	Channel 0	-	-	-	-
TSX 37-21		Channel 1	-	-	-
TSX 37-22			Channels 2 to 9	Channel 10	Channels 11 and 12

**Example** **%IW0.2** contains the channel 2 analog entry process value.



## Addressing in-rack Input/Output module objects

### At a Glance

Addressing of main bit and word objects of input/output modules is done geographically. That means that it depends:

- on the physical position of the module in the rack,
- on the module channel number.

### Illustration

Addressing is defined in the following way:

%	I, Q, M, K	X, W, D, F	Y	.	i	.	r
Symbol	Object type	Format	Position		Channel no.		Position

### Syntax

The table below describes the different elements of addressing.

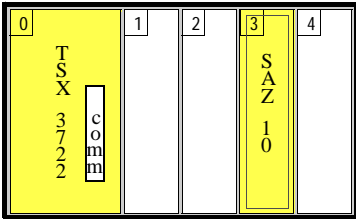
Family	Element	Values	Description
Symbol	%	-	-
Object type	I	-	Picture of the physical input of the module,
	Q	-	Picture of the physical output of the module, This information is exchanged automatically at each cycle of the task to which it is attached.
	M	-	Internal variable. This reading or writing information is exchanged as the application requires.
	K	-	Internal constant. This configuration information is available as read only.
Format (size)	X	-	Boolean. For Boolean objects this element can be omitted.
	W	16 bits	Single length.
	D	32 bits	Double length.
	F	32 bits	Floating point. The floating format used is the IEEE Std 754-1985 standard (equivalent to IEC 559).
Module position	y	0 to 4 0 to 6 0 to 8 0 to 10	Position number in the rack. TSX 37-05 TSX 37-08 TSX 37-10 TSX 37-21/22
Channel no.	i	0 to 31 or MOD	MOD: channel reserved for managing the module and the parameters common to all the channels.



Family	Element	Values	Description
Position	r	0 to 15 or ERR	Position of the bit in the word. ERR: indicates a module or channel fault.

### Examples

The table below presents some examples of object addressing.

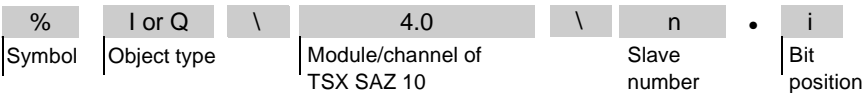
Object	Description	Illustration
%IW0.8	Process value of the channel 8 analog input of the built-in application-specific interface.	
%I1.5	Channel 5 input status of TSX DMZ 64 DTK module situated in position 1 in the rack.	
%Q1.2	Channel 2 output status of TSX DMZ 64 DTK module situated in position 1 in the rack.	
%IW7.1	Channel 1 analog input process value of TSX AEZ 801 module situated in position 7 in the rack.	
%I1.MOD.ERR	Information of a fault on TSX DMZ 64 DTK module situated in position 1 in the rack.	



## Addressing of language objects associated with AS-i bus

**At a Glance** Addressing of the main bit and word objects linked to the AS-i bus is geographical. That means that it is dependent on the number (address) of the slave device on the AS-I bus.

**Illustration** Addressing is defined in the following way:



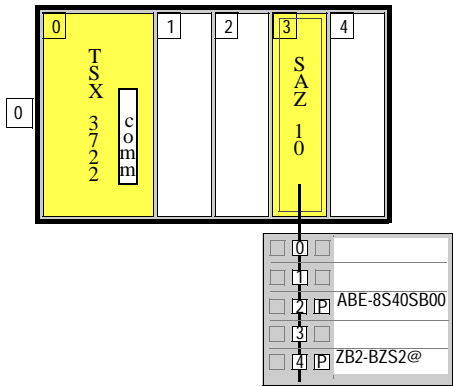
**Syntax** The table below describes the different elements of addressing.

Family	Element	Values	Description
Symbol	%	-	-
Object type	I Q	- -	Picture of the physical input of the module, Picture of the physical output of the module, This information is exchanged automatically at each cycle of the task to which it is attached.
Module position	4	-	Position number in the rack. It is vital that the TSX SAZ10 module is positioned in slot 4.
Channel no.	0	-	The TSX SAZ 10 module only has one channel.
Slave number	n	0 to 31	Physical slave address.
Position	i	0 to 3	Position of the input or output bit image.



**Example**            The table below presents some examples of object addressing.

Object	Description
%I\4.0\2.1	Input 1 of slave 2, with the module TSX SAZ 10 being positioned in slot 4 on rack 0.
%Q\4.0\6.3	Output 3 of slave 6, with the module TSX SAZ 10 being positioned in slot 4 on rack 0.





## 2.2 Implicit exchanges

---

### Implicit exchanges

---

#### Presentation

A built-in specific application interface where the addition of a module automatically enhances the application of language objects making it possible to program this interface or module.

These objects correspond to the images of the I/Os of the module or built-in specific application module.

The %I bits and the %IW words, images of the module's input values are updated automatically in the PLC processor at the start of the task, whether the task is in RUN or STOP mode.

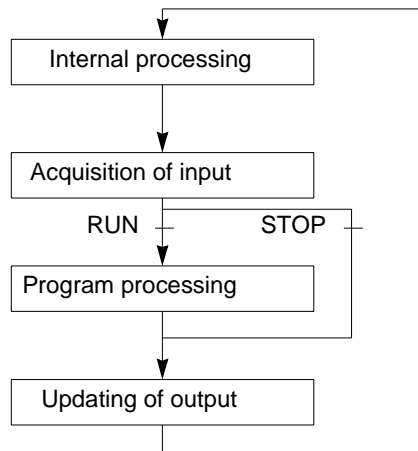
The %Q bits and %QW words, images of the module's output values are updated automatically in the module by the processor at the end of the task, with the task in RUN mode.

**Note:** When the task in STOP mode, depending on the configuration chosen:

- the outputs are in the fallback position (fallback mode),
- the outputs are maintained at their last value (maintain mode).

#### Illustration

The graph illustrates the operational cycle relating to a PLC task (cyclical execution).





**Examples**

The table below shows several examples of implicit exchanges relating to a given application.

Object	Specific application	Description
%I13.1	Discrete	Gives the status of channel 1 of the module located in position 3.
%IW4.2	Analog	Gives the analog value of channel 2 of the module located in position 4.
%IW3.2:X4	Counting	Gives the status of the capture input of the module located in position 3.
%Q6.5	Discrete	Gives the status of channel 5 of the module located in position 6.
%I6.5.ERR	-	Indicates, when the bit is set to 1, that channel 5 of the module located in position 6 is faulty.
%I7.MOD.ERR	-	Indicates, when the bit is set to 1, that the module located in position 7 is faulty.



## 2.3                   Explicit exchanges

---

### Presentation

<b>Subject of this section</b>	This section presents the principle of explicit exchanges as well as the different instructions which allow them to be carried out.																		
<b>What's in this Section?</b>	<div>This Section contains the following Maps:</div> <table><tr><th>Topic</th><th>Page</th></tr><tr><td>Explicit exchanges: General</td><td>35</td></tr><tr><td>READ_STS: Reading status words</td><td>37</td></tr><tr><td>WRITE_CMD: Writing command words</td><td>39</td></tr><tr><td>READ_PARAM: Reading adjustment parameters</td><td>40</td></tr><tr><td>WRITE_PARAM: Writing adjustment parameters</td><td>41</td></tr><tr><td>SAVE_PARAM: Saving adjustment parameters</td><td>42</td></tr><tr><td>RESTORE_PARAM: Restoring adjustment parameters</td><td>44</td></tr><tr><td>Exchange and report management</td><td>45</td></tr></table>	Topic	Page	Explicit exchanges: General	35	READ_STS: Reading status words	37	WRITE_CMD: Writing command words	39	READ_PARAM: Reading adjustment parameters	40	WRITE_PARAM: Writing adjustment parameters	41	SAVE_PARAM: Saving adjustment parameters	42	RESTORE_PARAM: Restoring adjustment parameters	44	Exchange and report management	45
Topic	Page																		
Explicit exchanges: General	35																		
READ_STS: Reading status words	37																		
WRITE_CMD: Writing command words	39																		
READ_PARAM: Reading adjustment parameters	40																		
WRITE_PARAM: Writing adjustment parameters	41																		
SAVE_PARAM: Saving adjustment parameters	42																		
RESTORE_PARAM: Restoring adjustment parameters	44																		
Exchange and report management	45																		



## Explicit exchanges: General

### Introduction

Explicit exchanges are exchanges carried out on request of the program user using instructions:

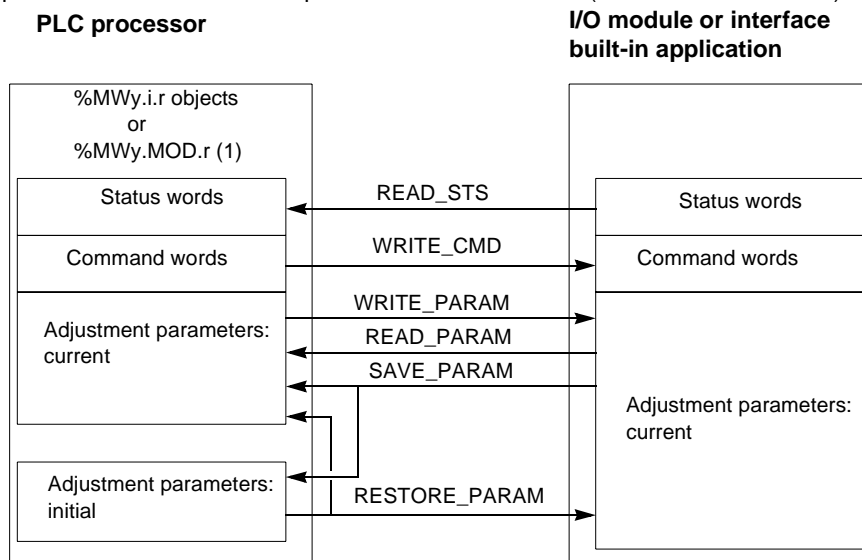
- READ\_STS (reading status words),
- WRITE\_CMD (writing command words),
- WRITE\_PARAM (writing adjustment parameters),
- READ\_PARAM (reading adjustment parameters),
- SAVE\_PARAM (saving adjustment parameters),
- RESTORE\_PARAM (restoring adjustment parameters).

These exchanges apply to a group of %MW objects that are of the same type (status, command or parameter) on the same channel.

**Note:** These objects are not essential when programming a application-specific function, but they carry extra information (eg: terminal block fault, module absent...) and the additional orders for advanced programming of the application specific functions (for more information on the explicit exchange Objects which are relative to a task, you can refer to the corresponding chapter).

### General principles on the use of the explicit instructions

The diagram below presents the different kinds of explicit exchanges that are possible between the PLC's processor and the module (or the built-in interface).



(1) Only with the instructions READ\_STS and WRITE\_CMD.



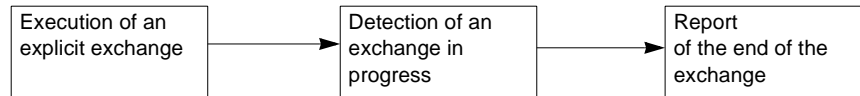
## Managing exchanges

During an explicit exchange, it may be interesting to monitor the progress that this makes, in order, for example, not to take into consideration the data read until the exchange has been carried out successfully.

For this, two kinds of information are available:

- the detection of an exchange which is in progress,
- the end of exchange report.

The diagram below describes the principal of the management of an exchange



---

## Logical channel %CHy.i

The %CHy.i channel is a general syntax for updating all of the objects which are of the same type and linked to this channel or to this group of channels, via the explicit instructions.

**Example:** `READ_STS%CH2 . 3` (reading the status words of channel 3 on module 2).

**Note:** In the case of a group of channels, the address used has to be the address of the first channel in the group of channels which is managed by the module.

**Example:** In the case of 32 discrete inputs (4 groups of 8 input channels, 0 to 7, 8 to 15, ...) situated in slot 3. In order to read channel 12's status words (fifth channel of the second group of channels), the syntax to use is: `READ_STS%CH3 . 8`.

---



## READ\_STS: Reading status words

### Introduction

Status words contain information about the functioning state of the module or the channel.

The READ\_STS instruction allows reading in the module (or in the built-in interface) of these kind of words. **This reading updates the %MW status words**.

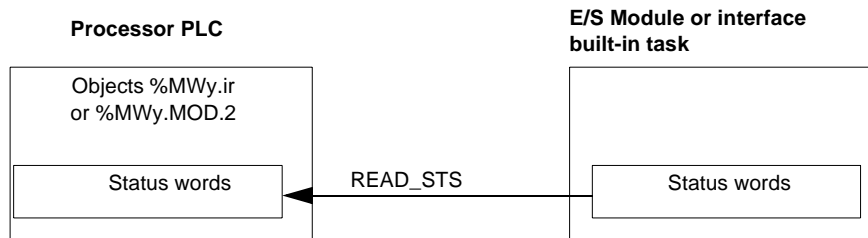
These words can be used to carry out diagnostics by program.

**Note:** The status words are specific to each task. However, 2 words are used by all of the modules in the TSX Micro range:

- %MWy.MOD.2 (module level fault),
- %MWy.i.2 (channel i level fault).

### Illustration

Writing instruction



### Syntax

The READ\_STS instruction is defined in the following way :

READ\_STS%CHy.i

The table below describes the different elements that are part of the instruction.

Element	Description
READ_STS	Name of the instruction
%CH	Channel-type object
y	Module position.
i	Channel or MOD number.



## Examples

The example below presents some examples of explicit exchanges using the READ\_STS instruction.

Object	Description
READ_STS%CH2.1	Reading the status words of channel 1, which is on the module situated in position 2.
READ_STS%CH1.MOD	Reading the status words of the module situated in position 1.

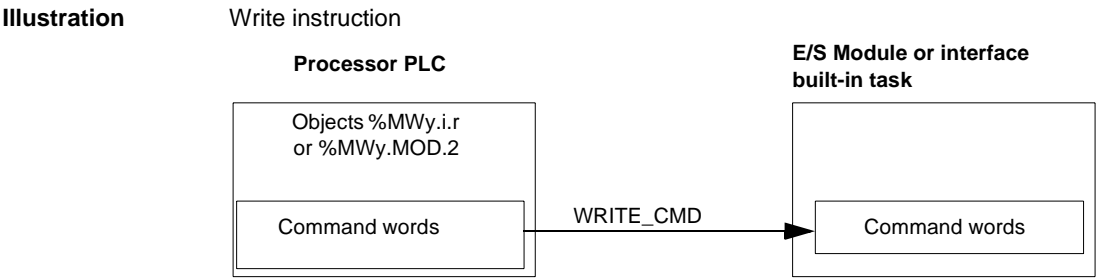
---



# WRITE\_CMD: Writing command words

**Introduction**      The command words act on the module or the channel (e.g.: reactivation of discrete outputs).  
**The WRITE\_CMD instruction enables the %MW command words to be written**  
to the module (or the built-in interface).

**Note:** The command words are specific to each application.



**Syntax**      The WRITE\_CMD instruction is defined as follows:  
WRITE\_CMD%CHy.i

The table below describes the different elements which constitute the instruction.

Element	Description
WRITE_CMD	Name of instruction.
%CH	Channel-type object.
y	Module position.
i	Channel number.

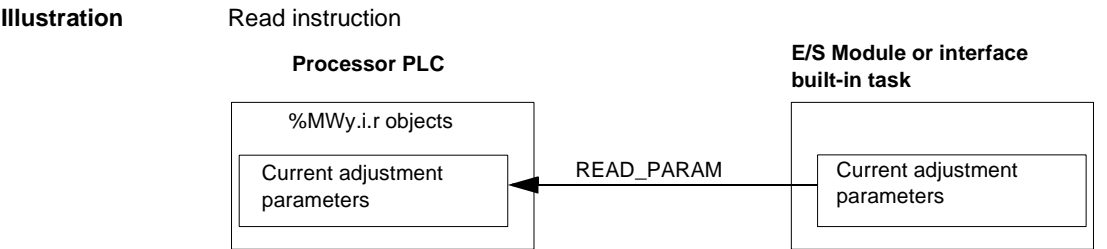
**Example**      The table below gives an example of an explicit exchange using the WRITE\_CMD instruction.

Object	Description
WRITE_CMD%CH3.0	Writing the command information of channel 0 of the module located in position 3.



# READ\_PARAM: Reading adjustment parameters

**Introduction** The READ\_PARAM instruction enables the adjustment parameters of the module (or built-in interface) to be read.  
**Reading updates the status words associated with the adjustment parameters %MWy.i.r .**



**Syntax** The READ\_PARAM instruction is defined as follows:  
READ\_PARAM%CHy . i

The table below describes the different elements which constitute the instruction.

Element	Description
READ_PARAM	Name of instruction.
%CH	Channel-type object.
y	Module position.
i	Channel number.

**Example** The table below gives an example of an explicit exchange using the READ\_PARAM instruction.

Object	Description
READ_PARAM%CH3.1	Reading the adjustment parameters of channel 1 of the module located in position 3.

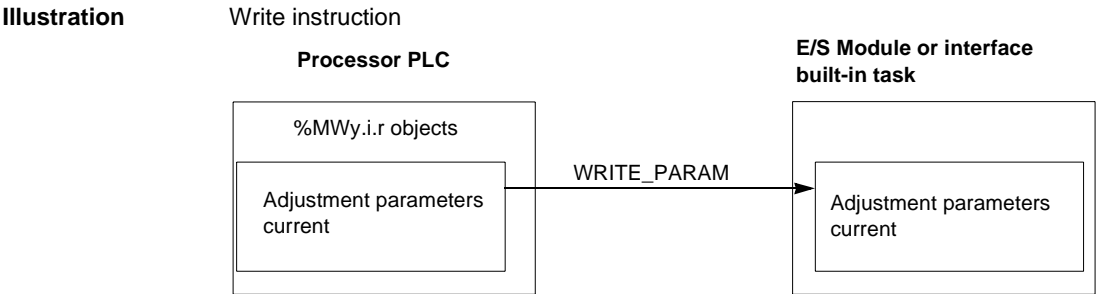


# WRITE\_PARAM: Writing adjustment parameters

**Introduction**

The WRITE\_PARAM instruction enables the adjustment parameters to be written to the module (or the built-in interface).

**This instruction enables the adjustment values defined in the configuration contained in the %MWy.i.r words to be modified by the program.**



**Syntax**

The WRITE\_PARAM instruction is defined as follows:  
`WRITE_PARAM%CHy.i`

The table below describes the different elements which constitute the instruction.

Element	Description
WRITE_PARAM	Name of instruction.
%CH	Channel-type object.
y	Module position.
i	Channel number.

**Example**

The table below gives an example of an explicit exchange using the WRITE\_PARAM instruction.

Object	Description
WRITE_PARAM%CH3.1	Writing the adjust parameters of channel 1 of the module located in position 3.



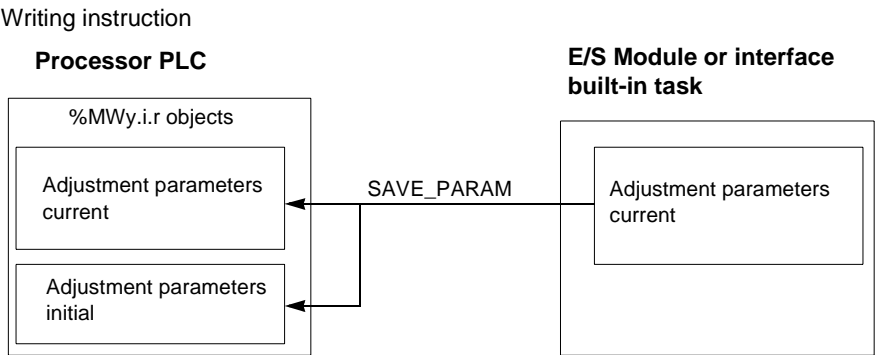
## SAVE\_PARAM: Saving adjustment parameters

### Introduction

When modifying the module's (or built-in interfaces') adjustment parameters, the SAVE\_PARAM instruction allows you to save these new parameters and substitute them for the initial parameters.  
These parameters replace the initial defined values with the help of the configuration editor (or of the last save ).  
The SAVE\_PARAM instruction is the equivalent to the order **Services** → **Save parameters**

**Note:** During a cold start, the current parameters (not saved) are replaced by the initial parameters.

### Illustration



### Syntax

The SAVE\_PARAM instruction is defined in the following way:  
SAVE\_PARAM%CHy.i

The table below describes the different elements that are part of the instruction.

Element	Description
SAVE_PARAM	Name of the instruction.
%CH	Channel-type object.
y	Module position.
i	Channel number.



**Example**

The table below presents an example of explicit exchange using the instruction SAVE\_PARAM.

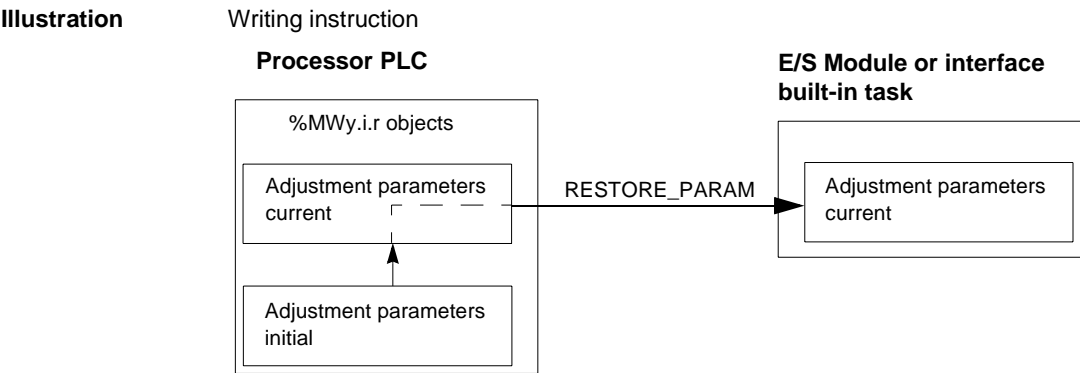
Object	Description
SAVE_PARAM%CH5.2	Reading then saving the adjustment parameters of channel 2, which is part of the module situated in position 5.

---



## RESTORE\_PARAM: Restoring adjustment parameters

**Introduction** The RESTORE\_PARAM instruction allows you to restore the initial adjustment parameters (written during configuration or during the last save). The RESTORE\_PARAM instruction is the equivalent to the order **Services** → **Save parameters**



**Syntax** The RESTORE\_PARAM instruction is defined in the following way:  
RESTORE\_PARAM%CHy.i

The table below describes the different elements that are part of the instruction.

Element	Description
RESTORE_PARAM	Name of the instruction
%CH	Channel-type object.
y	Module position.
i	Channel number.

**Example** The table below presents an example of explicit exchange using the instruction RESTORE\_PARAM.

Object	Description
RESTORE_PARAM%CH1.0	Writing the adjustment parameters of channel 0, which is part of the module situated in position 1.



# Exchange and report management

## Introduction

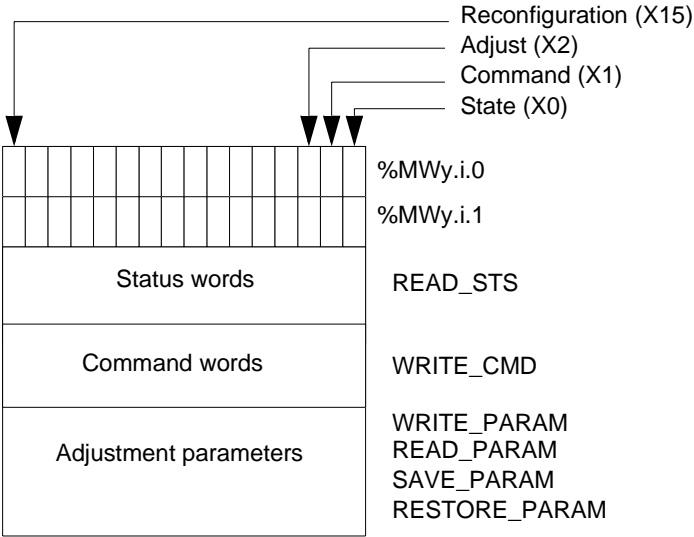
When data is exchanged between the PLC memory and the module, several task cycles may be required for the exchange to be acknowledged by the interface card. 2 words are used to manage the exchanges:

- %MWy.i: Exchange in progress,
- %MWy.i.1: Report

**Note:** These words are described in detail in each part on specific applications.

## Illustration

The illustration below shows the various bits which are significant for exchange management.





**Description of significant bits**

Each of the bits of the words %MWy.i.0 and %MWy.i.1 is associated with a type of parameter:

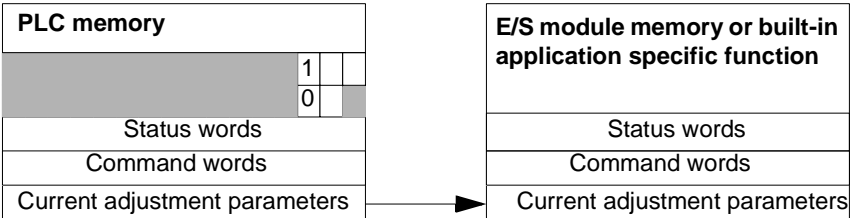
- position 0 bits are associated with status parameters:
  - the bit %MWy.i.0:X0 indicates whether a status word read request is in progress,
- position 1 bits are associated with command parameters:
  - the bit %MWy.i.0:X1 indicates whether the control parameters have been sent to channel i of the module,
  - the bit %MWy.i.1:X1 specifies whether the control parameters have been accepted by channel i of the module,
- position 2 bits are associated with adjustment parameters:
  - the bit %MWy.i.0:X2 indicates whether the adjustment parameters have been exchanged with channel i of the module (by WRITE\_PARAM, READ\_PARAM, SAVE\_PARAM, RESTORE\_PARAM),
  - the bit %MWy.i.1:X2 specifies whether the adjustment parameters have been accepted by the module. If the exchange has been made successfully the bit is set to 0,
- the position 15 bits indicate that channel i of the module has been reconfigured from the terminal (modification of the configuration parameters + cold start of the channel).

<p><b>Note:</b> The exchange and report words also exist at module level (%MWy.MOD and %MWy.MOD.1).</p>
---



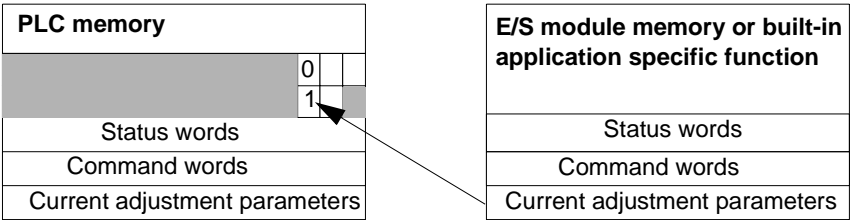
**Example**

Phase 1: Transmission of data using the WRITE\_PARAM instruction



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

Phase 2: Analysis of data by the I/O module and report



When data is exchanged between the PLC memory and the module, acknowledgment by the interface card is managed by the %MWy.i.1:X2 bit: Report (0 = exchange successful, 1=exchange unsuccessful).

**Note:** There are no adjustment parameters at module level.



## 2.4 Presymbolization

---

### Presentation

---

**Purpose of this section** This section presents the presymbolization function for the objects of a specific application.

---

**What's in this Section?** This Section contains the following Maps:

Topic	Page
Presymbolized objects	49
Automatic symbolization of objects associated with a channel	50

---



## Presymbolized objects

### Role

Certain application specific functions (examples: counting, analog, ...) support an automatic symbolization of the objects which are linked to them.

If you give the generic symbol of the module's %CHy.i channel, all of the symbols of the objects linked to this channel can then be automatically generated on request.

### Syntax

These objects are symbolized with the following syntax:

#### **PREFIX\_USER\_SUFFIX\_MANUFACTURER**

The elements have the following meaning and characteristics:

Element	Maximum number of characters	Description
<b>PREFIX_USER</b>	12	generic symbol given to the channel by the user
<b>SUFFIX_MANUFACTURER</b>	20	part of the symbol which corresponds to the bit object or word of the channel given by the system

**Note:** As well as the symbol, a manufacturer's comment is automatically generated, this comment recalls succinctly the object's role.

### Example

This example shows a counting module situated in slot 3 of the automatic tray.

If the generic symbol (prefix-user) given to channel 0 is `COMPT_PIECES`, the following symbols are automatically generated.

Address	Type	Symbol	Comment
%CH3.0	CH		
%ID3.0	DWORD	COMPT_PIECES_CUR-MEAS	Counter current value
%ID3.0.4	DWORD	COMPT_PIECES_CAPT	Counter captured value
%I3.0	EBOOL	COMPT_PIECES_ENAB_ACTIV	Counter enable active
%I3.0.1	EBOOL	COMPT_PIECES_PRES_DONE	Preset done



## Automatic symbolization of objects associated with a channel

### Introduction

The manufacturer's presymbolization (manufacturer's suffix) assigned to language objects is specific to each application. The detailed list of these suffixes is contained in the documentation relating to the specific application concerned.

### Conditions required

Automatic symbolization implies that:

- the module has been declared in the PL7 configuration in advance.
- the application-specific module accepts this function. The specific applications concerned are:
  - analog,
  - counting,
  - communication.

### Procedure

The table below shows the procedure for the automatic symbolization of objects associated with a channel.

Step	Action
1	Access the variable editor.
2	Access to I/O type variables. <b>Note:</b> The channels whose objects can be symbolized have a letter <b>P</b> on the button to the left of the <b>%CH</b> address.
3	Double-click on the P button for the channel to be symbolized.
4	Enter the user prefix. <b>Note:</b> If a symbol is already defined for the channel, the prefix proposed is the retrieved symbol truncated to 12 characters.
5	Confirm with the <b>Presymbolize</b> button.

### De-activating automatic symbolization.

Canceling automatic symbolization, for a given logical channel, makes it possible to delete all or part of an object's symbols.

Two options are proposed:

If the option chosen is ...	then ...
Delete all presymbols	No prefix is chosen: all symbols are deleted (including those which have been modified directly using the editor).
Delete prefixed presymbols	Only objects with a prefix identical to that entered are deleted.



---

# Application-specific instructions



---

## Presentation

### Subject of this chapter

This chapter presents the application-specific instructions.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Application-specific instructions	52
Accessing a specific function, method or procedure type instruction	53



## Application-specific instructions

---

### Introduction

Application-specific instructions (of function, method or procedure type) are programming instructions specific to an application-specific function which supplement the basic and advanced instructions.

They are defined in the documentation of each specific application.

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

**Note:** The Function type instructions require a supplementary memory application (only when they are actually used in the program). This memory requirement is to be considered for each function however much it is used, and this should be done in accordance with the memory capacity of the PLC in question.

### Example of a specific instruction

The (procedure type) `DISPLAY_MSG` function which is dedicated to operator dialog (CCX 17) consoles is used to display a status message contained in the CCX 17 memory.

```
DISPLAY_MSG (ADR#0.0.4,%KW200,%MW10:4)
```

Each parameter has a particular meaning:

- `ADR#0.0.4` is the console address,
- `%KW200` contains the data to be sent,
- ...



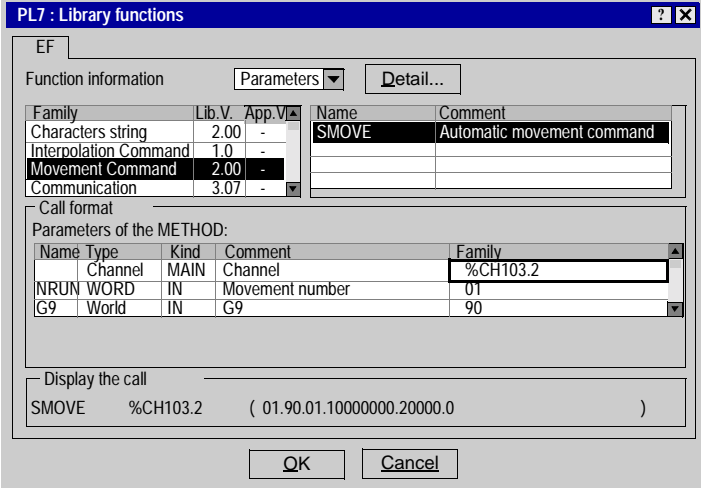
# Accessing a specific function, method or procedure type instruction

## Presentation

Application-specific function entry can be accessed by:

- directly entering the instruction and its parameters in an operate block,
- by the entry help function accessible through the program editors (LD, IL, ST).

## Calling up a function

Step	Action
1	Access the required editor.
2	<p>Depending on the editor used, choose one of the following methods to open the function library.</p> <ul style="list-style-type: none"> <li>• Press <b>Shift + F8</b> (LD, IL,ST editors).</li> <li>• Click on the ((LD editor) icon.</li> <li>• Select the command <b>Utilities</b> → <b>Enter call for a</b> ((IL, ST editors) function.</li> </ul> <p><b>Note:</b> The function library appears.</p> 
3	Select the specific application in the <b>Family</b> field.
4	Select the instruction in the <b>Name</b> field.
5	Many of the instructions have a customized entry help screen. Access this screen by clicking on the <b>Details</b> button.



Step	Action
6	Enter each instruction parameter (each instruction is developed in the relevant application-specific documentation) <ul style="list-style-type: none"><li>● in the customized screen</li><li>or</li><li>● in the <b>Entry field</b> in the <b>Library Functions</b> screen. To do this, the <b>Parameter</b> item must be selected in the <b>Function Information</b> field.</li></ul>
7	Click <b>Ok</b> to confirm.

---



---

Presentation

Subject of this chapter

This chapter introduces several elementary notions that are useful for the installation of application-specific functions.  
Some of these notions are taken from the PL7 Installation and Start-up Guide.

What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Reminders concerning the configuration editor	56
The PL7 toolbar	57
The PL7 status bar	58
How to declare a module in a PLC rack	59
Confirming the configuration of a module.	60
Globally reconfiguring an application.	61
Application-specific fault processing (in-rack modules) by program	62



## Reminders concerning the configuration editor

### Presentation

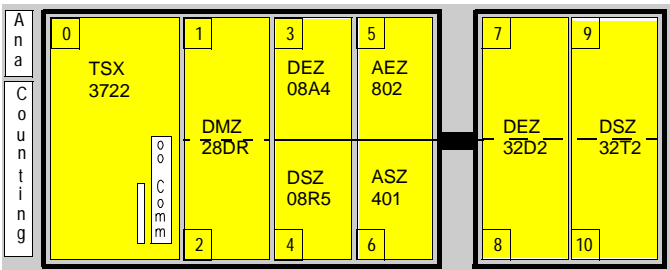
The configuration editor enables the various constitutive elements of the PLC to be declared and configured in a graphical and intuitive manner:

- rack,
- power supply,
- processor,
- application-specific modules.

In online mode the configuration editor also performs debug, adjustment and diagnostics functions.

### Illustration

The following screen provides an example of a hardware configuration.



### Accessing the editor

The following table shows the different ways of accessing the configuration editor.

From:	Action
the menu bar	Select <b>Tools</b> → <b>Configuration</b> .
the application browser	Double-click on <b>Hardware configuration</b> or select it using the arrow keys and confirm with <b>Enter</b> .

The screenshot shows the 'Application Browser' window with a tree view. The 'Configuration' folder is expanded, and 'Hardware configuration' is selected. Other items in the tree include 'Software configuration', 'Program', 'MAST Task', 'Events', 'DFB types', 'Variables', and 'Animation tables'.



## The PL7 toolbar

### Presentation

The software's basic functions can be accessed quickly via the toolbar, using the mouse.

Access to the different functions is dynamic and varies according to the context.

### Illustration

The PL7 toolbar is displayed as follows:



### Elements and functions

This table gives the function of each element in the toolbar:

Element	Function	Element	Function
	New application		Local mode
	Open an application		Online mode
	Save the application		PLC changes to RUN
	Print all or part of the application		PLC changes to STOP
	Undo last modifications		Start / Stop the animation
	Confirm modifications		Organize windows so that they overlap
	Go to		Tile windows horizontally
	Application browser		Tile windows vertically
	Cross references		Help
	Function library		What's this?
	PLC <-> terminal transfer		

**Note:** All these functions can also be accessed via the menu.



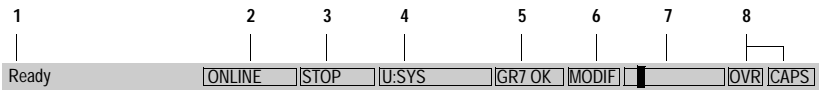
## The PL7 status bar

### At a Glance

The status bar, situated at the bottom of the screen, shows a range of information associated with operational aspects of the software.

### Illustration

The PL7 status bar appears thus:



### Elements and functions

This table describes the different zones that make up the status bar:

Number	Zone	Function
1	Information	supplies information concerning menu commands, toolbar icons and the different editors when these are selected.
2	Operating mode	indicates the current operating mode (offline, online).
3	PLC state	indicates the PLC state (Run, Stop, faulty, etc.).
4	Network address	gives the network address of the PLC.
5	Grafcet mode	indicates whether Grafcet mode is used in the application.
6	Modification in progress	indicates that the current application has not been saved or is different from the back-up.
7	Animation indicator	indicates that the PLC is in online mode.
8	Keyboard functions	indicates the status of the <b>Insert</b> and <b>All Caps</b> keyboard functions.

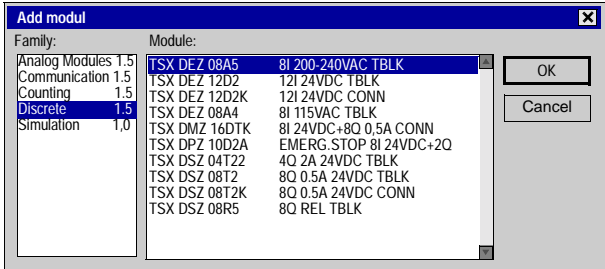
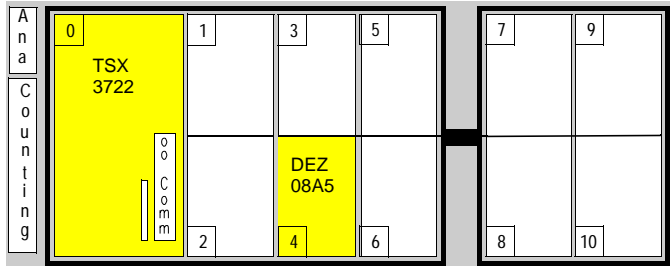


## How to declare a module in a PLC rack

### Procedure

This operation enables the user to make a software declaration of a module in a PLC rack.

The example below concerns a Discrete module; the same procedure is used whatever the type of in-rack module.

Step	Action
1	Access the application's hardware configuration screen.
2	<p>Double-click on the slot in which the module is to be configured.</p> <p><b>Result:</b> the <b>Add module</b> screen appears.</p> 
3	Select the specific application (e.g.: <b>Discrete</b> ) in the <b>Family</b> field.
4	Select the module reference in the <b>Module</b> field.
5	<p>Click on <b>Ok</b> to confirm the selection.</p> <p><b>Result:</b> the module is declared in its slot; the slot is shown in gray and contains the module reference.</p> 



## Confirming the configuration of a module.

---

### Introduction

When a module is declared, and when the configuration or adjustment parameters are modified, this module's configuration must be confirmed.

---

### Illustration


The table below shows where the confirmation of a module's configuration is positioned within the various hardware installation phases of an application.

Step	Description
1	Declaration of a module:
2	Configuration of module channels.
3	Adjustment
4	<b>Confirmation of the configuration of the module.</b>
5	Declaration and/or parametering of new modules
6	Global confirmation of the application

---

### Procedure

The table below shows the procedure for confirming the configuration of a module.

Step	Action
1	Select <b>Edit</b> → <b>Confirm</b> or use the icon  located in the toolbar.

---



## Globally reconfiguring an application.

### Introduction

An application needs to be globally reconfigured in order for the modifications confirmed for each module to be taken into account. This is generally carried out in local mode,

**Note:** Global reconfiguration of an application in online mode causes the module to stop.


### Illustration

The table below shows where the global reconfiguration of an application is positioned within the various hardware installation phases of an application.

Step	Description
1	Declaration of a module:
2	Configuration of module channels
3	Adjustment
4	Confirmation of the configuration of the module
5	Declaration and/or parametering of new modules
6	<b>Global confirmation of the application</b>

### Procedure

The table below shows the procedure for globally reconfiguring a module.

Step	Action
1	Select <b>Edit</b> → <b>Confirm</b> ... or use the icon  located in the toolbar.
2	Confirm the reconfiguration:



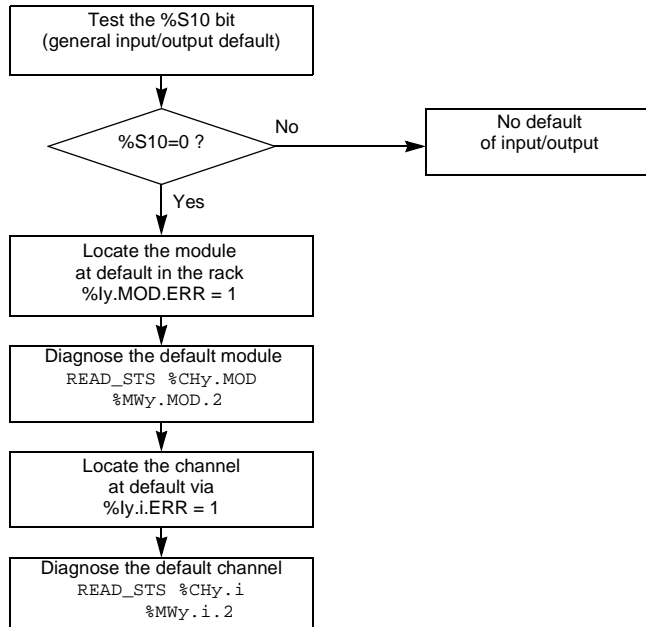
## Application-specific fault processing (in-rack modules) by program

### Presentation

Application-specific fault processing can be performed using the debugging and diagnostics screens.  
It may however be advantageous to use a program to perform this processing.

### Algorithm

The following algorithm is an application example of the detection and management application-specific (in rack) module faults by a program.





---

# Discrete application



---

## At a Glance

### Subject of this Part

This part presents the Discrete application-specific function on TSX37 PLCs and describes its installation using PL7 Micro, Junior and Pro software.

### What's in this part?

This Part contains the following Chapters:

Chapter	Chaptername	Page
5	General introduction to the discrete application -specific function	65
6	Discrete application configuration	67
7	Debugging discrete modules	95
8	Bits and words associated with discrete specific applications	105







---

# General introduction to the discrete application -specific function

## 5

---

### Introduction to the discrete application

#### Introduction

The application-specific discrete function on the TSX Micro applies to rack-mounted discrete input/output modules.

In order to install the discrete application, the physical context (processor and module type, and the position of the modules in the rack) must be defined and the necessary software installation carried out.

This second aspect will be carried out using the different PL7 editors:

- either in local mode,
- or in online mode. In this case, modification is limited to certain parameters.

**Note:** The application program can be created before the physical configuration to prevent it from being linked to a certain type of hardware. However, program execution will depend upon the definition of the physical configuration

---



**Installation principles**

The table below shows the different installation phases of the discrete application-specific function.

Mode	Phase	Description
Local	Declaration of a module	Selection of the module and its geographical position
	Configuration of module channels	Entry of configuration parameters.
	Confirmation of configuration parameters (See Configuration of discrete parameters, p. 81)	Confirmation of module level.
	Global confirmation of the application (See Confirming the configuration of a module., p. 60)	Confirmation of application level.
Local or online	<b>Programming</b>	Programming of functions to be performed by the application using: <ul style="list-style-type: none"> <li>● bit and word objects associated with the module,</li> <li>● specific application instructions.</li> </ul>
	<b>Transfer</b>	Transferring the application to the PLC.
	<b>Documentation</b>	Printing the different information relating to the application.
Online	<b>Debugging</b>	Debugging of the application using: <ul style="list-style-type: none"> <li>● debugging help screens enabling the user to control inputs/ outputs,</li> <li>● diagnostics screens enabling the user to identify faults.</li> </ul>

**Note:** The order shown above is given as an indication only. The PL7 software allows the editors to be used interactively in the order required (however, the data editor cannot be used if the input/output modules have not been configured first).



---

# Discrete application configuration



---

## At a Glance

### Subject of this Chapter

This chapter describes the Configuration aspect of the installation of the discrete application.

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
6.1	Configuring a discrete module: General	69
6.2	Discrete input/output channel parameters	76
6.3	Configuration of discrete parameters	81







# 6.1

# Configuring a discrete module: General

## At a Glance

**Subject of this Section** This section describes the basic operations required for configuring a Discrete module.

**What's in this Section?** This Section contains the following Maps:

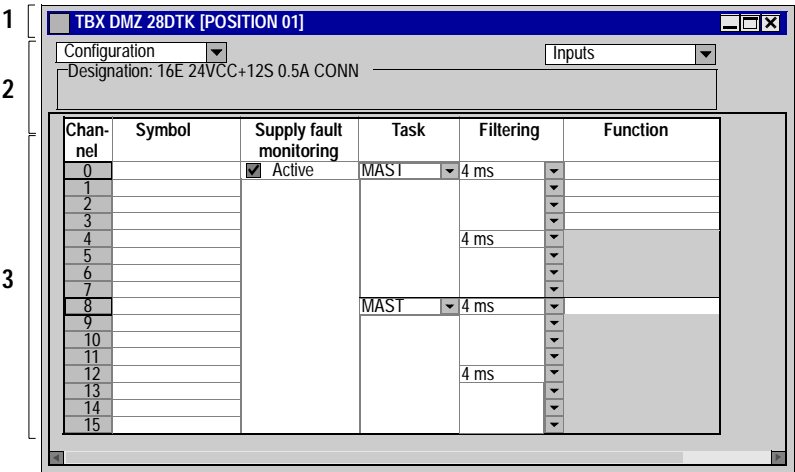
Topic	Page
Description of the configuration screen for a discrete module	70
How to access the configuration parameters of a discrete module.	72
Modifying the configuration parameters of a discrete module's: General	74



## Description of the configuration screen for a discrete module

**At a Glance** The configuration screen of the module selected in the rack displays the parameters associated with the Discreteinput or output channels.

**Illustration** This screen gives access to parameter viewing and modification in local mode, and Debugging in online mode.





**Description** The table below shows the different elements in the configuration screen and their functions.

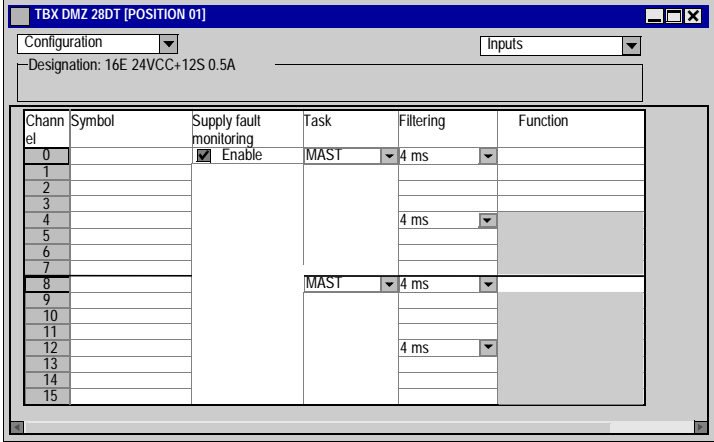
Number	Element	Function
1	Title bar	Indicates the reference of the selected module and its physical position in the rack.
2	Module zone	<p>Enables the user to select:</p> <ul style="list-style-type: none"> <li>the installation phase: <ul style="list-style-type: none"> <li><b>Configuration</b>,</li> <li><b>Debugging</b> (diagnostics), only accessible in online mode.</li> </ul> </li> <li>the type of channels (inputs or outputs), if the module in question has both inputs and outputs.</li> </ul> <p>Displays the designation of the selected module.</p> <p>According to the type of module, gives access to the configuration parameters which apply to all the channels of the same type of the module in question, such as:</p> <ul style="list-style-type: none"> <li>the type of inputs (positive or negative Logic),</li> <li>the network frequency,</li> <li>...</li> </ul> <p>Displaying this zone is optional. The selection is made using the command <b>View → Module zone</b>.</p>
4	Channel zone	<p>Enables the different channels to be parameterized. The <b>Symbol</b> column displays the symbol associated with the channel when this has been defined by the user (via the variables editor).</p> <p>Note: For discrete modules with more than 16 channels of the same type, use the scrollbar to access the different groups of channels.</p>



## How to access the configuration parameters of a discrete module.

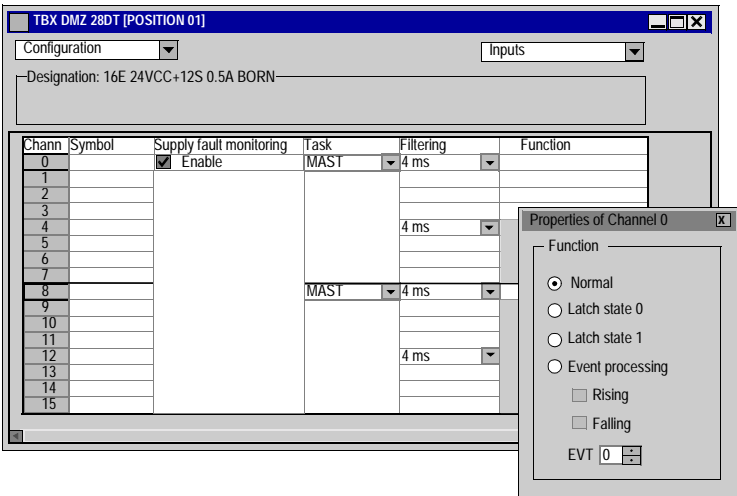
**Procedure**

This operation enables the user to access the configuration parameters for the channels of a discrete module.

Step	Action
	Access the module's <b>Hardware configuration</b> screen.
	<p>Double-click on the module to be configured or select the module and execute the command <b>Service</b> → <b>Open module</b>.</p> <p>For a standard format module which occupies two positions, double clicking on the high position will display the input channels parametering screen, and double clicking on the low position will display the output channels parametering screen.</p> <p><b>Result:</b> the configuration screen for the selected module appears.</p> 



Some modules have a dialog box giving access to additional parameters. To access this dialog box:

Step	Action																																																																																																						
1	<p>Perform one of the following:</p> <ul style="list-style-type: none"><li>● right click on the row in the table corresponding to the channel to be parametered and then select the <b>Properties</b> command from the pull-down menu,</li><li>● or double left click on the row in the table corresponding to the channel to be parameterized,</li><li>● or select the <b>Function</b> cell of the channel to be parameterized and then confirm using <b>Enter</b>.</li></ul> <p><b>Result:</b> The dialog box appears, superimposed on the configuration screen.</p>  <p>The screenshot displays the 'TBX DMZ 28DT [POSITION 01]' configuration window. It features a 'Configuration' dropdown, an 'Inputs' dropdown, and a 'Designation: 16E 24VCC+12S 0.5A BORN' label. Below these is a table with columns: Chann, Symbol, Supply fault monitoring, Task, Filtering, and Function. The table lists channels 0 through 15. Channel 0 is selected, and the 'Properties of Channel 0' dialog box is open. This dialog box shows the 'Function' settings for Channel 0, including radio buttons for 'Normal', 'Latch state 0', and 'Latch state 1', a checkbox for 'Event processing', and checkboxes for 'Rising' and 'Falling' edge detection. The 'EVT' field is set to 0.</p> <table><tr><th>Chann</th><th>Symbol</th><th>Supply fault monitoring</th><th>Task</th><th>Filtering</th><th>Function</th></tr><tr><td>0</td><td></td><td><input checked="" type="checkbox"/> Enable</td><td>MAST</td><td>4 ms</td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>4</td><td></td><td></td><td></td><td>4 ms</td><td></td></tr><tr><td>5</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>6</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>7</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>8</td><td></td><td></td><td>MAST</td><td>4 ms</td><td></td></tr><tr><td>9</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>10</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>11</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>12</td><td></td><td></td><td></td><td>4 ms</td><td></td></tr><tr><td>13</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>14</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>15</td><td></td><td></td><td></td><td></td><td></td></tr></table> <p>Properties of Channel 0</p> <p>Function</p> <p><input checked="" type="radio"/> Normal</p> <p><input type="radio"/> Latch state 0</p> <p><input type="radio"/> Latch state 1</p> <p><input type="radio"/> Event processing</p> <p><input type="checkbox"/> Rising</p> <p><input type="checkbox"/> Falling</p> <p>EVT 0</p>	Chann	Symbol	Supply fault monitoring	Task	Filtering	Function	0		<input checked="" type="checkbox"/> Enable	MAST	4 ms		1						2						3						4				4 ms		5						6						7						8			MAST	4 ms		9						10						11						12				4 ms		13						14						15					
Chann	Symbol	Supply fault monitoring	Task	Filtering	Function																																																																																																		
0		<input checked="" type="checkbox"/> Enable	MAST	4 ms																																																																																																			
1																																																																																																							
2																																																																																																							
3																																																																																																							
4				4 ms																																																																																																			
5																																																																																																							
6																																																																																																							
7																																																																																																							
8			MAST	4 ms																																																																																																			
9																																																																																																							
10																																																																																																							
11																																																																																																							
12				4 ms																																																																																																			
13																																																																																																							
14																																																																																																							
15																																																																																																							



## Modifying the configuration parameters of a discrete module's: General

---

### Introduction

The configuration editor provides a set of functions which enable the user to input or modify the parameters of modules such as:

- contextual menus,
  - single or multiple selection of channels,
  - cutting/pasting of parameters (using the contextual menus).
- 

### Accessing the contextual menus

These can be accessed by right-clicking with the mouse, and give quick access to the main commands.

If the element to be selected is...	Then the available functions are...
the cell	Copy parameters
	Paste parameters
the module zone (except in tables)	Undo modifications
	Confirm
	Animate

---

### Selecting a channel or cell

The table below shows the procedure for selecting a module channel cell or channel.

Step	Action
1	Left-click on the required cell or channel number.

---

### Selecting a group of consecutive channels

The table below shows the procedure for selecting a group of consecutive channels of a module.

Step	Action
1	Select the first channel.
2	Press <b>Shift</b> and click on the last channel.

---



**Selecting a group of non-consecutive channels**

The table below shows the procedure for selecting a group of non-consecutive channels of a module.

Step	Action
1	Select the first channel.
2	Press <b>Ctrl</b> and click on each of the channels in turn.

**Selecting a group of consecutive cells**

The table below shows the procedure for selecting a group of consecutive cells of a module.

Step	Action
1	Select the first cell.
2	Move the mouse up or down whilst holding down the mouse button, then release the button when the last cell has been reached.



## 6.2 Discrete input/output channel parameters

### At a Glance

**Subject of this section** This section introduces the different input and output channel parameters by discrete module type.

**What's in this Section?** This Section contains the following Maps:

Topic	Page
Discrete inputs parameters	77
Discrete output parameters	79



## Discrete inputs parameters

### At a Glance

Discrete input modules contain parameters per channel, or per group of consecutive channels.

### Parameters

The table below shows the parameters available for each discrete input module.

Module reference	No. of inputs	Associated task (1)	Function (2)	Filtering (group of 4 channels)	Type of logic	Network	Supply fault monitoring
TSX DEZ 08A4	8	<b>Mast</b> / Fast / None	-	<b>20 ms</b>	-	<b>50 Hz</b> / 60 Hz	<b>Active</b> / Inactive
TSX DEZ 08A5	8	<b>Mast</b> / Fast / None	-	<b>20 ms</b>	-	<b>50 Hz</b> / 60 Hz	<b>Active</b> / Inactive
TSX DEZ 12D2	12	<b>Mast</b> / Fast / None	-	<b>4 ms</b> or (4)	<b>positive</b> / negative	-	<b>Active</b> / Inactive
TSX DEZ 12D2K	12	<b>Mast</b> / Fast / None	-	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
TSX DEZ 32D2	32	<b>Mast</b> / Fast / None	<b>Normal</b> or (3)	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
TSX DPZ 10D2A	8 (inputs)	<b>Mast</b> / Fast / None	-	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
TSX DMZ 16DTK	8 (inputs)	<b>Mast</b> / Fast / None	-	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
TSX DMZ 28DT	16 (inputs)	<b>Mast</b> / Fast / None	<b>Normal</b> or (3)	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
TSX DMZ 28DTK	16 (inputs)	<b>Mast</b> / Fast / None	<b>Normal</b> or (3)	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
TSX DMZ 28DR	16 (inputs)	<b>Mast</b> / Fast / None	<b>Normal</b> or (3)	<b>4 ms</b> or (4)	<b>positive</b> / negative	-	<b>Active</b> / Inactive
TSX DMZ 28AR	16 (inputs)	<b>Mast</b> / Fast / None	<b>Normal</b> or (3)	<b>20 ms</b>	-	<b>50 Hz</b> / 60 Hz	<b>Active</b> / Inactive

#### Key:

(1)	Group of 8 consecutive channels, except for the last group if this has less than eight channels.
(2)	Only available on channels 0, 1, 2, 3 and 8 of the module situated in slot 1.
(3)	Latching 0 or 1, event processing on rising edge (RE), or on falling edge (FE).
(4)	0.1 to 7.5 ms



Module reference	No. of inputs	Associated task (1)	Function (2)	Filtering (group of 4 channels)	Type of logic	Network	Supply fault monitoring
TSX DMZ 64DTK	32 (inputs)	<b>Mast</b> / Fast / None	<b>Normal</b> or (3)	<b>4 ms</b> or (4)	-	-	<b>Active</b> / Inactive
<b>Key:</b>							
(1)	<b>Group of 8 consecutive channels, except for the last group if this has less than eight channels.</b>						
(2)	<b>Only available on channels 0, 1, 2, 3 and 8 of the module situated in slot 1.</b>						
(3)	<b>Latching 0 or 1, event processing on rising edge (RE), or on falling edge (FE).</b>						
(4)	<b>0.1 to 7.5 ms</b>						

**Note:**

- Parameters in bold correspond to the parameters configured by default.
- When a group of channels is not assigned to a task **None**, the corresponding channels are not exchanged by the system.



## Discrete output parameters

### At a Glance

The discrete output modules contain parameters per channel or per group of consecutive channels.

### Parameters

The table below shows the parameters available for each discrete output module.

Module reference	No. of outputs	Associated task (1)	Reactivate	Fallback mode	Supply fault monitoring
TSX DSZ 04T22	4	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DSZ 08R5	8	<b>Mast</b> / Fast / None	-	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DSZ 08T2	8	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DSZ 08T2K	8	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DSZ 32R5	32	<b>Mast</b> / Fast / None	-	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DSZ 32T2	32	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DMZ 16DTK	8 (outputs)	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DMZ 28AR	12 (outputs)	<b>Mast</b> / Fast / None	-	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DMZ 28DR	12 (outputs)	<b>Mast</b> / Fast / None	-	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DMZ 28DT	12 (outputs)	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DMZ 28DTK	12 (outputs)	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DMZ 64DTK	32 (outputs)	<b>Mast</b> / Fast / None	<b>Programmed</b> / Automatic	<b>Fallback</b> to 0/ Maintain state	<b>Active</b> / Inactive
TSX DPZ 10D2A	2 (outputs)	<b>Mast</b> / Fast / None	-	-	<b>Active</b> / Inactive
<b>Key:</b>					
<b>(1)</b>	<b>Group of 8 consecutive channels, except for the last group if this has less than eight channels.</b>				



**Note:**

- Parameters in bold correspond to the parameters configured by default.
- When a group of channels is not assigned to a task **None**, the corresponding channels are not exchanged by the input/output system.



---

## 6.3 Configuration of discrete parameters

---

### Presentation

#### Subject of this section

This section presents the installation of different discrete I/O channel configuration parameters.

#### What's in this Section?

This Section contains the following Maps:

Topic	Page
How to modify the Task parameter of a discrete module	82
How to modify the Monitoring the external supply parameter for a discrete module	83
How to modify the Functions parameter of a discrete module.	84
How to modify the Type of inputs parameter of a discrete module	86
How to modify the Network frequency parameter of a discrete module	87
How to modify the Filtering time parameter for 24 DCV inputs of a discrete module	88
How to modify the Fallback mode parameter of a discrete module	89
How to modify the Reactivation of outputs parameter of a discrete module.	90
How to parameterize the RUN/STOP input of a discrete module	91
How to parameterize the save program and %MW input of a discrete module	92
How to parameterize the alarm output of a discrete module	93



## How to modify the Task parameter of a discrete module

### At a Glance

This parameter defines the processor task in which inputs are acquired and outputs are updated.

The task is defined for 8 consecutive channels.

**Note:** If the module consists of a number of inputs which is not a multiple of 8, the last group will be made up of a number of channels between 1 and 7.

The possible choices are as follows:

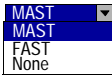
- the **MAST** task
- the **FAST** task
- **None** if the group of channels is not used.

It is advisable to deconfigure the groups of channels not used in the application. As a result, even if no connector is connected, the module will not indicate a fault.

**Note:** This parameter can only be modified in local mode.

### Procedure

The table below shows the procedure for defining the type of task assigned to the channels of a module.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>In the pull-down menu located in the <b>Task</b> column, click on the button for the required group of channels.</p> <p><b>Result:</b> a pull-down list appears.</p>  <p><b>Note:</b> For discrete modules with more than 16 channels, use the scrollbar to access the different groups of channels.</p>
3	<p>Select the required task.</p> <p><b>Note:</b> Selecting <b>None</b> causes the group of channels in question and the following groups to be deconfigured, after the modification is confirmed. Reconfiguration (<b>MAST</b> or <b>FAST</b> selection) of a group of deconfigured channels will, after the modification is confirmed, cause the preceding non-configured groups to be reconfigured.</p>
4	Confirm the deconfiguration or reconfiguration if necessary.



---

## How to modify the Monitoring the external supply parameter for a discrete module

---

### At a Glance

This parameter defines the status (activated or deactivated) of external supply fault monitoring.  
It works on groups of 16 consecutive channels.  
By default checking is active (box checked).

---

### Procedure

The table below shows the procedure for activating or deactivating the Monitoring external supply fault function.

Step	Action
1	Access the hardware configuration screen of the required module.
2	For the required group of channels, click on the check box situated in the <b>Supply fault monitoring</b> column. Note: For discrete modules with more than 16 channels, use the scrollbar to access the different groups of channels.

---



## How to modify the Functions parameter of a discrete module.

---

### At a Glance

This parameter enables the user to assign to the first four input channels (%I1.0 to %I1.3) of a discrete module located in position 1, a specific function, such as:

- normal (no event associated with the channel),
- channel by channel latching,

This function is used to take into account a short duration pulse (less than one PLC cycle time) in order to process it in the next cycle.

Acknowledgement of the pulse occurs when the input state changes:

- Latch state 1: change from state 0 to state 1,
- Latch state 0: change from state 1 to state 0.

**Note:** The switch to latching can be accessed in either local or online mode.

- channel by channel event processing,

This function is used to acknowledge events so that they are immediately processed (uninterrupted processing).

The possible choices are as follows:

- event triggered on a rising edge (RE) of the pulse (state 0 -> state 1),
- event triggered on a falling edge (FE) of the pulse (state 1 -> state 0),

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

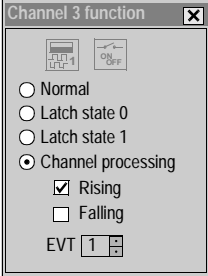
- 1 to 8 on the **TSX 37-10**PLC,
- 0 to 15 on the **TSX 37-21/22**PLC.

**Note:** The switching to event processing can be accessed in either local or online mode.



# **Procedure**

The table below shows the procedure for accessing the various functions.

Step	Action
1	Access the hardware configuration screen of the module situated in position 1.
2	Select the required channel.
3	<p>In the <b>Function</b> column, double click on one of the first four cells of the channel to be parameterized.</p> <p><b>Result:</b> the properties dialog box appears.</p> 
4	Select the required function.
5	If necessary, enter the event number <b>Evti</b> .
6	Repeat the operation for every channel to be configured (from step 2).



## How to modify the Type of inputs parameter of a discrete module

### At a Glance

This parameter defines the type of inputs of the modules **TSX DEZ 12D2** and **TSX DMZ 28DR**.  
It applies to all the module's inputs.

The possible choices are as follows:

- positive logic inputs: Sink (common to + supply sensors),
- negative logic inputs: Source (common to - supply sensors).

**Note:** This parameter can only be modified in local mode.

### Procedure

The table below shows the procedure for defining the type of inputs assigned to the module.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>Click on the <b>Positive logic</b> / <b>Negative logic</b> check box, located in the <b>Type of input</b> field.</p> <p><b>Result:</b> the type of input chosen will be assigned to the module in this way.</p> <div data-bbox="495 846 1094 998"></div> <p><b>Note:</b> Modifying the type of input implies:</p> <ul style="list-style-type: none"><li>• the corresponding modification of the position of the switch situated on the module,</li><li>• different input connections depending on whether the inputs are at <b>Positive logic</b> (sink) or at <b>Negative logic</b> (source).</li></ul>



# How to modify the Network frequency parameter of a discrete module

## At a Glance

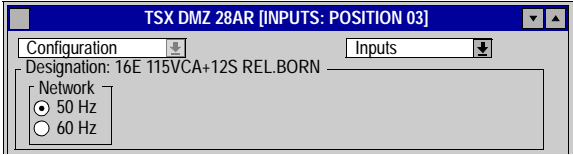
This parameter is used to adapt alternating inputs to the network frequency (50 or 60 Hz) of the **TSX DEZ 08A4**, **TSX DEZ 08A5** and **TSX DMZ 28AR** modules (inputs).

It applies to all the module's inputs.

**Note:** The network frequency can only be modified in local mode.

## Procedure

The table below shows the procedure for defining the **Network frequency** assigned to the module.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>Click on the <b>50Hz / 60Hz</b> check box located in the module zone of the <b>Network</b>field.</p> <p><b>Result:</b> the network frequency chosen will therefore be assigned to the module.</p> 



## How to modify the Filtering time parameter for 24 DCV inputs of a discrete module

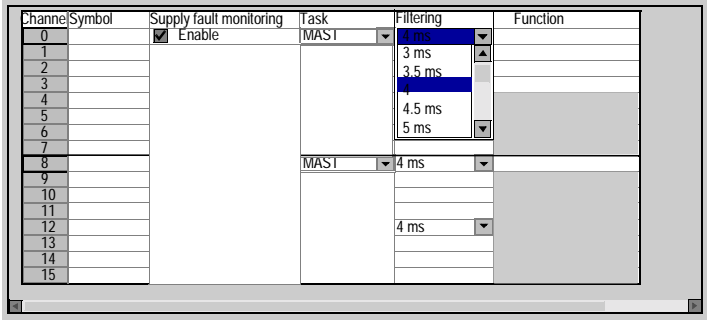
### At a Glance

This parameter defines the filtering period for the selected channel.  
The filtering time is defined for a group of 4 consecutive channels and may vary between 0.1 and 7.5 ms in increments of 0.5.

**Note:** The filtering time can only be modified in local mode.

### Procedure

The table below shows the procedure for defining the Filtering time parameter.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>In the pull-down menu located in the <b>Filtering</b> column, click on the button for the required group of channels.</p> <p><b>Result:</b> a pull-down list appears.</p> 
3	Select the required filtering period.



# How to modify the Fallback mode parameter of a discrete module

## At a Glance

This parameter defines the fallback mode adopted by outputs when the PLC changes to Stop, in the event of a processor or rack fault.

The possible modes are as follows:

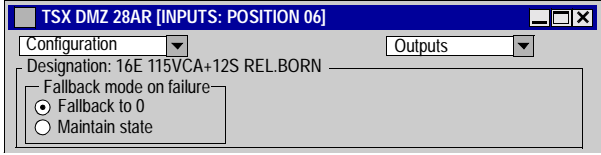
- **Fallback to 0:** the outputs are set at state 0,
- **Maintain state:** the outputs remain in the state in which they were before the change to Stop.

This parameter applies:

- to all the outputs of a module for half format modules or mixed module outputs,
- to the even or odd position for a 32 output channel module.

## Procedure

The table below shows the procedure for defining the Fallback mode assigned to a group of channels.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>Click on the <b>Fallback to 0 / Maintain state</b> check box located in the module zone of the <b>Fallback mode</b> field.</p> <p><b>Result:</b> the fallback mode chosen will then be assigned to the module.</p> 



## How to modify the Reactivation of outputs parameter of a discrete module.

### At a Glance

This parameter defines the reactivation mode of static outputs.

The possible modes are as follows:

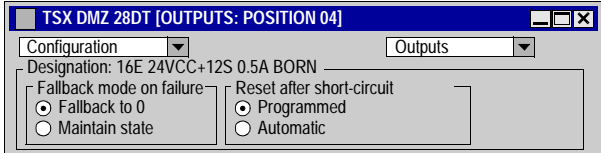
- **Programmed:** reactivation is executed by a PLC application command. In order to avoid closely spaced, repetitive reactivation, the module automatically allows a period of 10 seconds between two reactivations,
- **Automatic:** reactivation is carried out automatically every 10 seconds until the fault disappears.

This parameter applies:

- to all the outputs of a module for half format modules or mixed module outputs,
- to the even or odd position for a 32 output channel module.

### Procedure

The table below shows the procedure for defining the reactivation of outputs.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>Click on the <b>Programmed / Automatic</b> check box located in the module zone of the <b>Reactivation after short-circuit</b> field.</p> <p><b>Result:</b> the reactivation of outputs chosen will then be assigned to the module.</p> 



# How to parameterize the RUN/STOP input of a discrete module

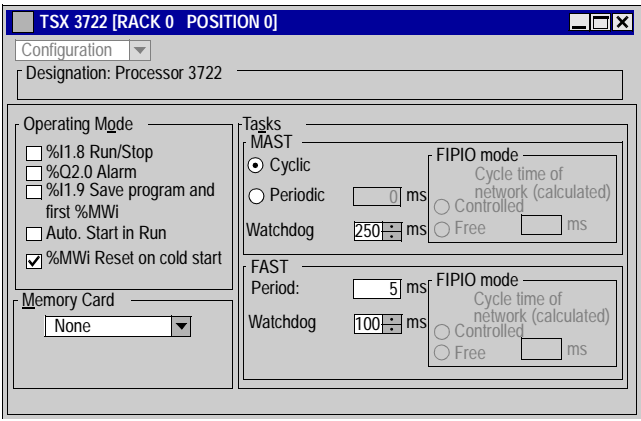
## At a Glance

The discrete input (%I1.8) can be parameterized to control the starting (Run) or stopping (Stop) of the execution of the application program.

**Note:** The change to Stop via the physical input %I1.8 takes priority over a change to Run command from a terminal or network.

## Procedure

The table below shows the procedure for parameterizing the Run/Stop input.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>Double click on the 0 position.</p> <p><b>Result:</b> The processor configuration screen appears.</p> 
3	Check the %I1.8 Run/Stop box in the <b>Operating mode</b> field.



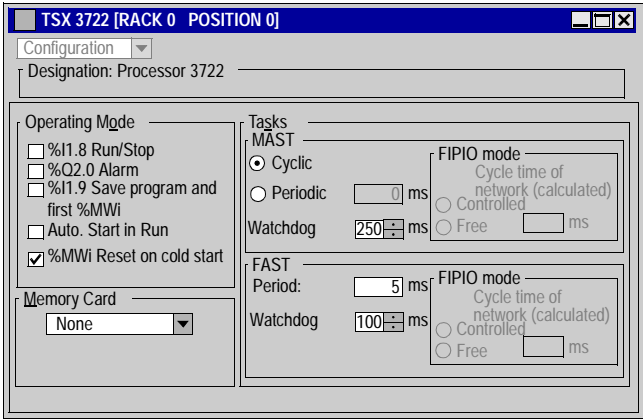
## How to parameterize the save program and %MW input of a discrete module

### At a Glance

The discrete input (%I1.9) can be assigned to the save program and %MWi internal words function.  
Thus, when input %I1.9 is rising edge, this starts the RAM -> FLASH EPROM transfer of the program application and the %Mwi.

### Procedure

The table below shows the procedure for configuring the %I1.9 input as a save program and %MWi input.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<div>Double click on the 0 position. <b>Result:</b> The processor configuration screen appears.</div> <div></div>
3	Check the <b>%I1.9 Save program and first %MWi</b> box in the <b>Operating mode</b> field.



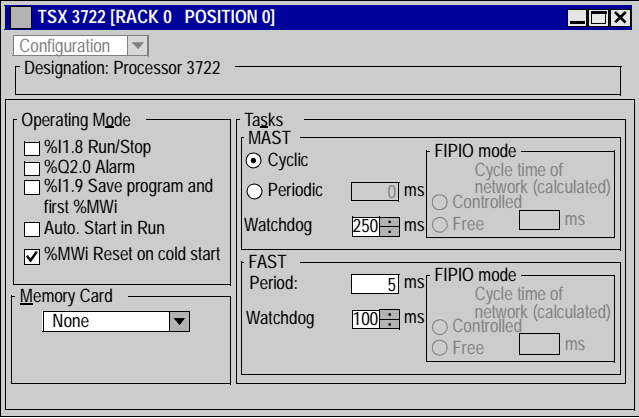
# How to parameterize the alarm output of a discrete module

## At a Glance

The discrete output (%Q2.0) can be assigned to the Alarm function. Then, as soon as a blocking fault appears, output %Q2.0, normally at state 1, switches to state 0.

## Procedure

The table below shows the procedure for assigning the Alarm function to the output.

Step	Action
1	Access the hardware configuration screen of the required module.
2	<p>Double click on the 0 position.</p> <p><b>Result:</b> The processor configuration screen appears.</p> 
3	Check the <b>%Q2.0 Alarm</b> box in the <b>Operating mode</b> field.







---

# Debugging discrete modules

# 7

---

## At a Glance

### Subject of this chapter

This chapter describes the Debug aspect of the installation of the discrete application.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Introduction to the debug function of a discrete module.	96
Description of the debug screen for a discrete module.	97
How to access the channels debug screen of a discrete module.	99
How to access the Diagnostics function for a discrete module	100
Accessing the forcing/unforcing function	101
How to access the SET and RESET commands	102
How to access the reactivation of outputs command	103
Maintain outputs for a discrete module	104



## Introduction to the debug function of a discrete module.

---

### Introduction

The Debug function enables the user to view the parameters of each channel (symbol, channel status, etc.) for each of the application's discrete input/output modules, and to access diagnostics and adjustment for the selected channel in the event of a fault being present.

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

<b>Note:</b> This function can only be accessed in online mode.
---

---



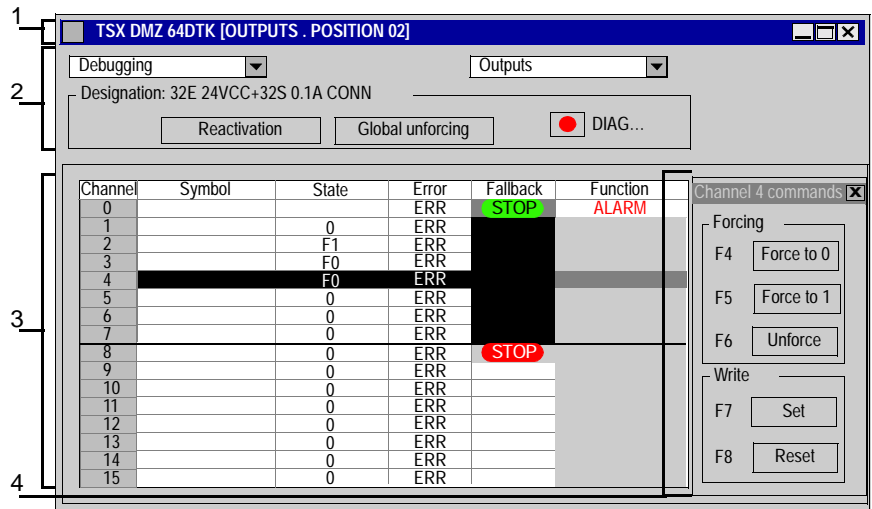
## Description of the debug screen for a discrete module.

### At a Glance

The debug screen displays the value and state of each of the selected module's channels in real time. It is also used to access the channel commands (forcing of the input/output value, Set/Reset an output, etc.).

### Illustration

The debug screen looks like this:





**Description** The table below shows the different elements in the debug screen and their functions.

Number	Element	Function
1	Title bar	Indicates the reference of the selected module and its physical position in the PLC.
2	Module zone	<p>Enables the user to select:</p> <ul style="list-style-type: none"> <li>parameter type: <ul style="list-style-type: none"> <li><b>Configuration</b>,</li> <li><b>Debugging</b> (diagnostics), can only be accessed in online mode.</li> </ul> </li> <li>channel type (inputs or outputs) displayed, if the module in question contains both inputs and outputs or the position (even or odd) for 32 channel modules.</li> </ul> <p>Displays the designation of the selected module.</p> <p>Provides direct access to:</p> <ul style="list-style-type: none"> <li>the <b>Global unforcing</b> function for channels,</li> <li>the module diagnostics, if the module is faulty (shown by the indicator integrated in the diagnostics access button, <b>DIAG</b>, which turns red).</li> <li>the <b>Reactivation</b> of outputs function ( for static output modules).</li> </ul>
3	Channel zone	<p>Displays, in real time, the value and status of each of the channels of the module. The <b>Symbol</b> column displays the symbol associated with the channel when this has been defined by the user (via the variables editor).</p> <p><b>Note:</b> This zone also enables the user to display specific functions associated with the inputs %I1.0..%I1.3 and the output %Q2.0 of the discrete module.</p>
4	Command zone	<p>Gives access to the commands for a channel</p> <p><b>Note:</b> When a "counting application" function is associated with one of the first four inputs of the discrete module located in slot 1, an icon allows access to the adjustment and debug screens of this function (see).</p>



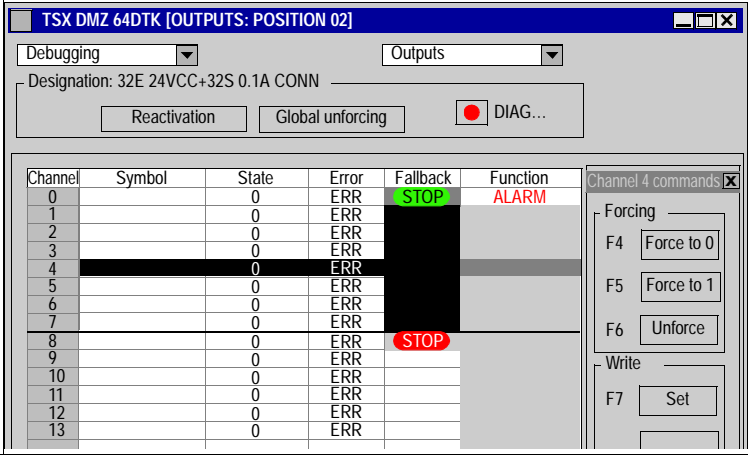
## How to access the channels debug screen of a discrete module.

### At a Glance

This operation enables the user to access the debug screen for the channels of a discrete module.  
The debug screen can only be accessed in online mode.

### Procedure

The table below shows the procedure for accessing the **Debug** function.

Step	Action
1	Access the hardware configuration screen.
2	<p>Double click on the module to be configured or select the module then execute the command <b>Service</b> → <b>Open module</b>.</p> <p><b>Result:</b> The configuration screen for the selected module appears.</p> 



## How to access the Diagnostics function for a discrete module

### At a Glance

The module Diagnostics function displays the current faults, if there are any, classified according to their category:

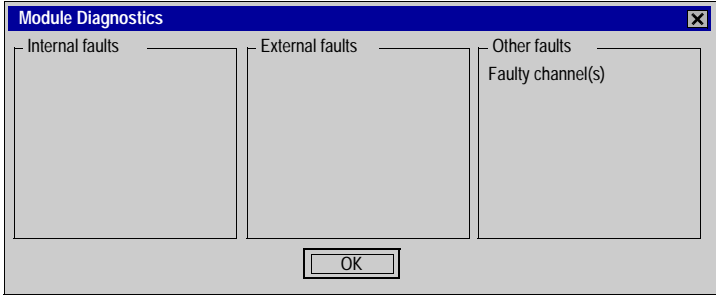
- internal faults (module failure),
- external faults (tripped output, sensor or actuator supply fault),
- other faults (module missing or switched off).

The DIAG indicator turns red to indicate a faulty module

- in the configuration editor at rack level:
  - the module position indicator,
- in the configuration editor at module level:
  - the **Diag** indicator.

### Procedure

The table below shows the procedure for accessing the module Diagnostics screen.

Step	Action
1	Access the module's debug screen.
2	<p>Click on the <b>Diag</b> button in the module zone.</p> <p><b>Result:</b> The list of module faults appears.</p>  <p><b>Note:</b> If a configuration fault occurs, in the event of a major failure or missing module, the module diagnostics screen cannot be accessed. The following message then appears on the screen: Module is missing or different from that configured in this position .</p>



# Accessing the forcing/unforcing function

## Presentation

This function is used to modify the status of all or part of a module's channels. The status of a forced output is fixed and can only be modified by the application after unforcing.

**Note:** However, when a fault causes the outputs to fall back, the status of these channels takes the value set when the fallback mode parameter was configured.

The available commands are as follows:

- for one or more channels:
  - forcing to 1,
  - forcing to 0,
  - unforcing (when the selected channel(s) is (are) forced),
- for all channels of a module (when at least one channel is forced):
  - global unforcing of channels.

## Procedure

The table below shows the procedure for forcing or unforcing all or part of the channels of a module.  
See Modifying the configuration parameters of a discrete module's: General, p. 74 for multiple selection.

Step	Action for one channel	Action for all channels
1	Access the module's debug screen.	
2	In the <b>Status</b> column, double-click on the cell of the required channel (1).	Click on the <b>Global unforcing</b> button in the module zone.
3	Select the required function.	-
<b>Key:</b>		
(1)	The <b>Channel commands</b> screen can also be accessed by double-clicking on the required channel and then left-clicking on the <b>Command</b> button.	



## How to access the SET and RESET commands

---

### At a Glance

These commands are used to modify the state of a module's outputs to 0 (RESET) or 1 (SET).

**Note:** The state of the output assigned by one of these commands is temporary and can be modified at any time by the application when the PLC is in RUN.

### Procedure

The table below shows the procedure for assigning the value 0 or 1 to all or part of the channels of a module.

See (Modifying the configuration parameters of a discrete module's: General, p. 74) for multiple selection.

Step	Action for one channel
1	Access the module's debug screen.
2	In the <b>Status</b> column, double click on the cell of the required channel (1).
3	Select the required function.
<b>Key:</b>	
(1)	The <b>Channel commands</b> screen can also be accessed by double-clicking on the required channel and then left-clicking on the <b>Command</b> button.



# How to access the reactivation of outputs command

---

**At a Glance**      If a fault has caused the outputs to trip, this command is used to reactivate all the outputs (configured in programmed reactivation) if no fault persists at their terminals.

---

**Procedure**      The table below shows the procedure for reactivating tripped outputs.

Step	Action
1	Access the debug screen of the module in question.
2	Click on the <b>Reactivate</b> button, located in the module zone.

---



## Maintain outputs for a discrete module

---

### At a Glance

This check (red Stop indicator on) informs the user that, for a given output channel group, these outputs are not being correctly maintained by the PLC (fallback state). The possible causes are as follows:

- processor fault,
  - rack fault.
-



---

# Bits and words associated with discrete specific applications



---

## Presentation

### Subject of this chapter

This chapter presents the different word and bit objects associated with discrete specific applications as well as their addressing mode.

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
8.1	Addressing of discrete I/O module objects	107
8.2	Language objects associated with the discrete specific application	108







## 8.1 Addressing of discrete I/O module objects

### Addressing discrete input/output module objects

#### At a Glance

Addressing of bit and word objects for input/output modules is defined in the part entitled Shared applications (See Addressing of language objects associated with specific applications, p. 25).

This page describes the specific features linked to the discrete input/output modules.

#### Illustration

Reminder of the principles of addressing:

%	I, Q, M, K	X, W or D	x	.	i	.	r
Symbol	Object type	Format	Position		Channel no.		Position

#### Specific values

The table below shows the values specific to discrete input/output module objects.

Element	Values	Comment
i	0 to 63 or MOD	MOD: channel reserved for management of the module and the parameters shared by all channels.
r	0 to 3 or ERR	ERR: indicates a module or channel fault.



## 8.2 Language objects associated with the discrete specific application

### Presentation

**Subject of this section** This section presents the different language objects associated with the discrete specific application.

**What's in this Section?** This Section contains the following Maps:

Topic	Page
Implicit exchange objects	109
Exchange management: Module %MWx.MOD.0:Xj or channel %MWx.i.0:Xj exchanges in progress	110
Exchange management: Module %MWx.MOD.1:Xj or channel %MWx.i10:Xj report	111
Explicit exchange objects: General	112
Explicit exchange objects: Status Module %MWx.MOD.2:Xj	113
Explicit exchange object: Command Word %MWxMOD.3:Xj	114



---

## Implicit exchange objects

---

### At a Glance

Exchanges for these objects are carried out automatically on each cycle of the task in which the module channels are configured.

---

### Bit objects

The table below shows the different implicit exchange bit objects.

Address	Function	Meaning when the bit is at state 1
%Ix.i	Input channel bit	Indicates for input channel i that the output of the sensor controlling the input is activated.
%Qx.i	Output channel bit	Indicates that output channel i is activated.
%Ix.i.ERR	Channel error bit	Indicates that input channel i is faulty.
%Ix.MOD.ERR	Module error bit	Indicates that the module is faulty.

---



## Exchange management: Module %MWx.MOD.0:Xj or channel %MWx.i.0:Xj exchanges in progress

---

### At a Glance

These word-type objects provide information on the module or channel i exchanges in progress.  
They are updated automatically by the system.

---

### Description

The table below explains the meanings of the different bits in the word **%MWx.MOD.0**.

Address	Meaning for Xj = 1
%MWx.MOD.0:X0	Exchange of status words in progress on at least one channel of the module.
%MWx.MOD.0:X1	Exchange of command words in progress on at least one channel of the module.

---

### Description

The table below explains the meanings of the different bits in the word **%MWx.i.0**.

Address	Meaning for Xj = 1
%MWx.i.0:X0	Exchange of status words in progress on channel i.
%MWx.i.0:X1	Exchange of command words in progress on channel i.

---

### Example

The example below shows one possible way in which this type of word can be used

```
(* Update request for channel 0 status words *)
(* for the module located at slot 3*)
(* if no exchange in progress on this channel *)
IF NOT %MW3.0:X0 THEN READ_STS %CH3.0;
END_IF;
```

**Note:** When the duration of the explicit exchange is shorter than the PLC task cycle time, the %MWx.i:X0 bit never switches to 1.

---



## Exchange management: Module %MWx.MOD.1:Xj or channel %MWx.i10:Xj report

**At a Glance** These word-type objects provide information on the exchange reports for the module or channel i. They are updated automatically by the system.

**Description** The table below explains the meanings of the different bits in the word **%MWx.MOD.1**.

Address	Meaning for Xj = 1
%MWx.i.1:X0	Status parameter exchange error on at least one channel of the module.
%MWx.i.1:X1	Command parameter exchange error on at least one channel of the module.

**Description** The table below explains the meanings of the different bits in the word **%MWx.i.1**.

Address	Meaning for Xj = 1
%MWx.i.1:X0	Status parameter exchange error on channel i.
%MWx.i.1:X1	Command parameter exchange error on channel i.

**Example** The example below shows one possible way in which this type of word can be used

```
( * Status error detected on module * )
( * located at slot 3 * )
IF NOT %MW3.MOD.0:X0 THEN READ_STS %CH3.MOD;
END_IF;
IF %MW3.MOD.1:X0 THEN SET %M100;
END_IF;
```



## Explicit exchange objects: General

---

### At a Glance

Explicit exchange objects carry information (e.g. terminal block fault, module missing, etc.) and additional commands for advanced programming of application specific functions.

**Note:** The configuration constants %KWx.i.r, which are not documented in this manual, can only be accessed in read mode and correspond to the configuration parameters entered using the Configuration editor.

Explicit exchange objects are exchanged at the request of the user program using the following instructions:

- READ\_STS (read status words),
  - WRITE\_CMD (write command words),
  - WRITE\_PARAM (write adjustment parameters),
  - READ\_PARAM (read adjustment parameters),
  - SAVE\_PARAM (save adjustment parameters),
  - RESTORE\_PARAM (restore adjustment parameters).
-



---

## Explicit exchange objects: Status Module %MWx.MOD.2:Xj

---

### At a Glance

This word-type object provides information on the status of the module.

---

### Description

The table below explains the meanings of the different bits in the word:  
**%MWx.MOD.2.**

Address	Bit position	Meaning for Xj = 1
%MWx.MOD.2:X0	0	Internal error Module inoperative.
%MWx.MOD.2:X1	1	Reserved.
%MWx.MOD.2:X2	2	Reserved.
%MWx.MOD.2:X3	3	Reserved.
%MWx.MOD.2:X4	4	Reserved.
%MWx.MOD.2:X5	5	Reserved.
%MWx.MOD.2:X6	6	Module Missing or Switched Off
%MWx.MOD.2:X7	7	Reserved.
%MWx.MOD.2:X8	8	Tripped Output.
%MWx.MOD.2:X9	9	Sensor or Actuator Supply Fault.
%MWx.MOD.2:X10 to X15	10 to 15	Reserved.

---



## Explicit exchange object: Command Word %MWxMOD.3:Xj

---

### At a Glance

This word-type object is a command word. It is written by the application or from the Debug screen for the reactivation command.

---

### Description

The table below explains the meanings of the different bits in the word:  
**%MWx.MOD.3.**

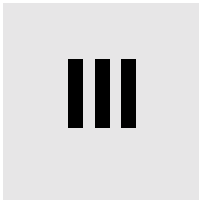
Address	Bit position	Meaning for Xj = 1
%MWx.MOD.3:X0	0	Reactivation of tripped outputs.
%MWx.MOD.3:X1 to 3	1 to 3	Reserved.
%MWx.MOD.3:X4	4	Inhibiting the monitoring of the external supply fault for the channels 0 to 15.
%MWx.MOD.3:X5	5	Confirming the monitoring of the external supply fault for the channels 0 to 15.
%MWx.MOD.3:X6	6	Inhibiting the monitoring of the external supply fault for the channels 16 to 31.
%MWx.MOD.3:X7	7	Confirming the monitoring of the external supply fault for the channels 16 to 31.
%MWx.MOD.3:X8 to 11	8 to 11	Reserved.
%MWx.MOD.3:X12	12	Inhibiting the monitoring of the external supply fault for the channels 0 to 15.
%MWx.MOD.3:X13	13	Confirming the monitoring of the external supply fault for the channels 0 to 15.
%MWx.MOD.3:X14	14	Inhibiting the monitoring of the external supply fault for the channels 16 to 31.
%MWx.MOD.3:X15	15	Confirming the monitoring of the external supply fault for the channels 16 to 31.

---



---

# AS-i Bus



---

## At a Glance

**Aim of this part** This part introduces the AS-i bus on PLC TSX37 and describes its implementation using the software PL7 Micro, Junior and Pro.

**What's in this part?** This Part contains the following Chapters:

Chapter	Chaptername	Page
9	General introduction to the AS-i Bus	117
10	AS-i bus configuration	125
11	Debugging the AS-i bus	137
12	Bits and words associated with the AS-i function	149
13	AS-i operating mode	163
14	AS-i performance	169







---

# General introduction to the AS-i Bus

## 9

---

### At a Glance

#### Aim of this Chapter

This chapter introduces the AS-i bus on PLC TSX37 and describes how to access the various application specific editors.

#### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Introduction to the AS-i Bus	118
Architecture of the TSX SAZ 10 module	120
Structure of an AS-i slave	122
How to declare an AS-i communication module in the PLC rack	123
How to access the AS-i Bus configuration	124



## Introduction to the AS-i Bus

---

### At a Glance

The AS-i (Actuator Sensor-Interface) Bus is used for the interconnection, on a single cable, of sensors/actuators at the lowest level of automation. These sensors/actuators will be defined in the documentation as **slave devices**.

The implementation of the AS-i application requires you to define the physical context into which the application will be integrated (PLC rack, modules, AS-i slave devices connected on the bus) and then to complete the software implementation.

This second aspect will be carried out from the different PL7 editors:

- either in local mode,
  - or in connected mode; in this case modification is limited to certain parameters.
-



**Implementation principles**

The table below shows the different implementation phases for the AS-i bus.

Mode	Phase	Description
Local	Declaration of module	Installation in the rack slot.
	Declaration of slave devices	Selection for each device: <ul style="list-style-type: none"> <li>• of its slot number on the bus,</li> <li>• of the slave type.</li> </ul>
	Configuration of the module channel (See AS-i bus configuration, p. 125)	Input of the configuration parameters.
	Confirmation of the configuration parameters (See Confirming the configuration of a module., p. 60)	Confirmation on the module level.
	Global confirmation of the application (See Globally reconfiguring an application., p. 61)	Confirmation at application level.
Local or connected	<b>Symbolization</b>	Symbolization of the variables associated with the slave devices.
	<b>Programming</b>	Programming of the functions carried out via the AS-i bus.
	<b>Transfer</b>	Transfer of the application onto the PLC.
	<b>Documentation</b>	Printing of the different information relating to the application.
Connected	<b>Debugging</b>	Debugging the application using: <ul style="list-style-type: none"> <li>• debug help screens allowing you to view the slave connections, their parameters, etc.</li> <li>• diagnostic screens allowing you to identify errors.</li> </ul>

**Note:** The order defined above is given for information purposes. The PL7 software allows you to use the editors interactively in the required order. (However, you cannot use the data editor or the program editor without having first configured the module and the slave devices.)



## Architecture of the TSX SAZ 10 module

---

### At a Glance

The **TSX SAZ 10** module operates in master/slave mode. The master controls only the exchanges on the bus.

The AS-i standard defines several levels of service offered by the master:

- Profile M0 - Minimum Master: the master only offers the configuration of the slaves connected on the bus on power up and only input/output exchanges.
- Profile M1 - Full Master: this profile covers all the functionality defined by the AS-i standard,
- Profile M2 - Reduced Master: this profile matches the functionality of profile M0 with the option of parameterizing the slaves.

**Note:** The TSX SAZ 10 module matches the M2 profile with the additional option of reading diagnostic information from the slaves.

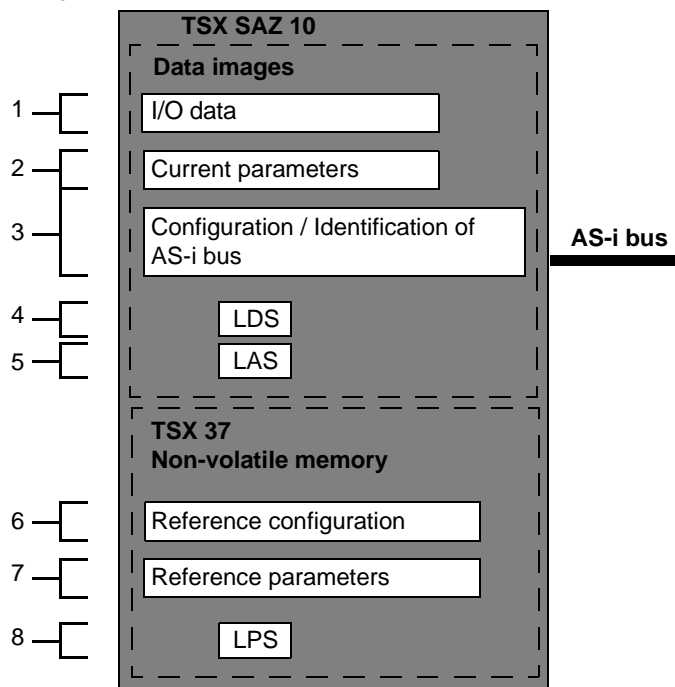
The module includes data fields that allow you to manage the lists of slaves and the images of input / output data. This information is stored in volatile memory.

---



## Illustration of the architecture

The figure below shows the architecture of the **TSX SAZ 10** module.



## Description of the constituent elements

The table below shows the different elements making up the architecture of the **TSX SAZ 10** module.

Number	Element	Description
1	I/O data	Images of the 124 inputs and 124 outputs of the AS-i Bus.
2	Current parameters	Image of the parameters of all the slaves.
3	Configuration/ Identification	This field contains all the I/O codes and the identification codes for all the slaves detected.
4	LDS	List of all slaves detected on the bus.
5	LAS	List of slaves activated on the bus.
6	Reference configuration	Module reference configuration.
7	Reference parameters	Parameters saved via PL7 or resulting from a backup.
8	LPS	List of slaves provided on the bus and configured via PL7.



## Structure of an AS-i slave

### At a Glance

The AS-i bus is used to interconnect 31 slave devices, each one having:

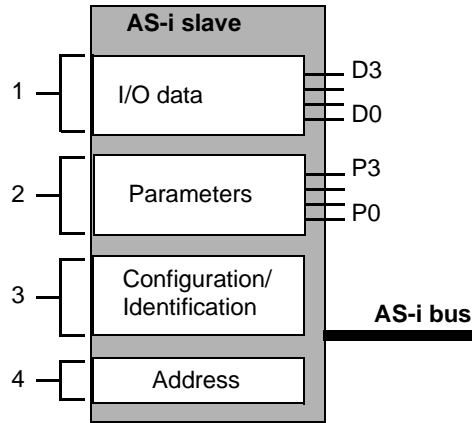
- 4 input bits,
- 4 output bits,
- 4 parametering bits,

Thus the AS-i bus can manage a maximum number of 248 I/O.

Each slave has its own address and a profile (defines variables exchange).

### Structure illustration

The figure below shows the structure of an AS-i slave.



### Description of constituent elements

The table below shows the different elements that make up the structure of an AS-i slave.

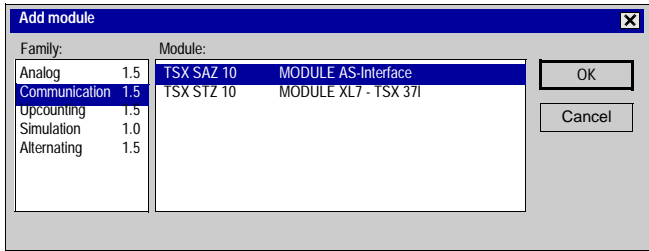
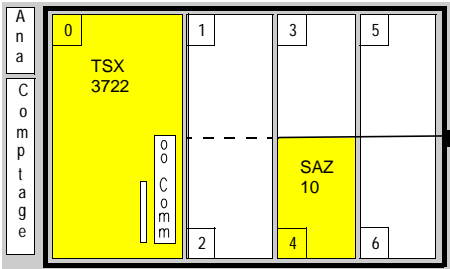
Number	Element	Description
1	Input/output data	Input data is stored by the slave and made available for the AS-i master. Output data is updated by the master module.
2	Parameters	The parameters are used to control and switch internal operating modes to the sensor or the actuator.
3	Configuration/ Identification	This field contains: <ul style="list-style-type: none"> <li>• the code which corresponds to I/O configuration,</li> <li>• the slave identification (ID) code.</li> </ul>
4	Address	Physical address of slave.
<b>Note:</b> The operating parameters, address, configuration and identification data are saved in a non-volatile memory.		



## How to declare an AS-i communication module in the PLC rack

### Procedure

This operation is used to declare an AS-i communication module in the PLC TSX 37 rack using software.

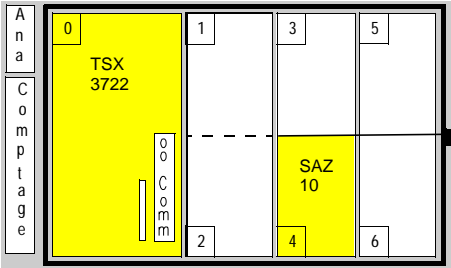
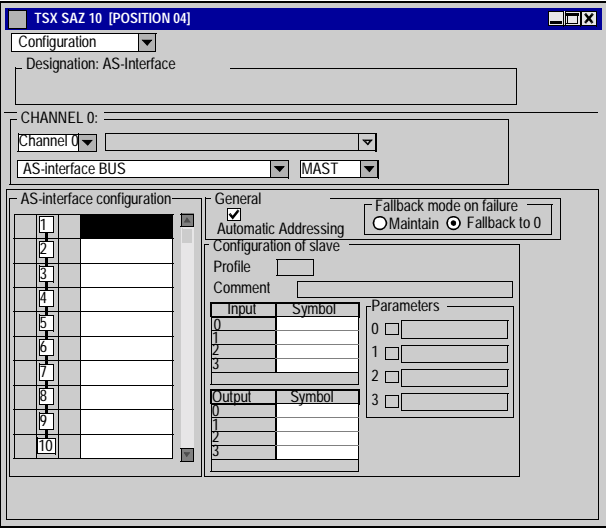
Step	Action
1	<p>Double click on slot 4 of the base rack.</p> <p><b>Result:</b> The dialog box <b>Add module</b> appears:</p> 
2	Select <b>Communication</b> in the <b>Family</b> field.
3	Select the reference of the module in the <b>Module</b> field.
4	<p>Confirm the selection with <b>OK</b>.</p> <p><b>Result:</b> The module is declared in its slot; the latter is grayed out and contains the module reference.</p>  <p><b>Note:</b> Only one <b>TSX SAZ 10</b> communication module can be installed in a TSX 37 configuration.</p>



## How to access the AS-i Bus configuration

**Procedure**

This operation is used to access the configuration of the AS-i Bus communication module.

Step	Action
1	<p>Access the <b>Hardware configuration</b> screen</p> <p><b>Result:</b> The rack hardware configuration screen appears.</p> 
2	<p>Double click on the position of the communication module or select the module and execute the command <b>Service</b> → <b>Open module</b>.</p> <p><b>Result:</b> The hardware configuration screen for the module selected appears.</p> 



---

## At a Glance

### Subject of this Chapter

This Chapter describes the Configuration aspect for installing the AS-i bus.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Description of the configuration screen for an AS-i communication module	126
How to define a slave device on the AS-i bus	128
How to modify the software configuration of the AS-i Bus	130
How to access the description of an AS-i slave	131
How to define a new slave profile in the standard AS-I catalogue	133
How to modify AS-i slave general parameters: Automatic addressing	135
How to modify AS-i slave general parameters: Fallback mode	136



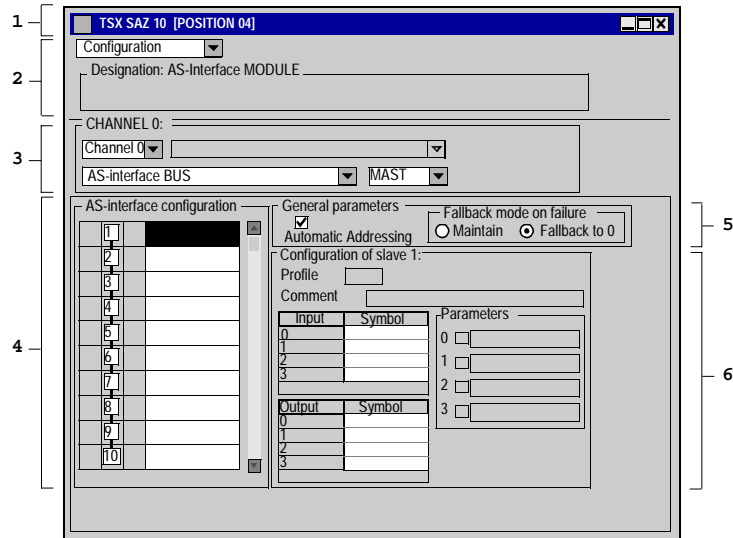
## Description of the configuration screen for an AS-i communication module

### At a Glance

The configuration screen of the AS-i module gives access to the parameters associated with the module and the slave devices.

### Illustration

This screen allows display and modification of parameters in local mode, and also debugging in connected mode.





**Description**      The table below shows the different elements of the configuration screen and their functions.

Number	Element	Function
1	Title bar	Indicates the reference of the selected module and its physical position in the rack.
2	Command zone	<p>Allows the selection of the parameter type:</p> <ul style="list-style-type: none"> <li>● <b>Configuration</b>,</li> <li>● <b>Debugging</b> (diagnostic), only accessible in connected mode.</li> </ul> <p>Displays the designation of the selected module.</p> <p>The display of this zone is optional. The selection is made by using the command <b>View → Module zone</b>.</p>
3	Channel zone	<p>Allows selection of the task in which the information coming from the AS-i communication channel will be scanned:</p> <ul style="list-style-type: none"> <li>● <b>MAST</b> task,</li> <li>● <b>FAST</b> task.</li> </ul> <p>The display of this zone is optional. The selection is made by using the command <b>View → Channel zone</b>.</p>
4	AS-i configuration zone	<p>Allows:</p> <ul style="list-style-type: none"> <li>● viewing of the slave devices connected to the bus,</li> <li>● definition of a new configuration (adding, modifying or deleting slave devices).</li> </ul>
5	General parameters zone	<p>Allows the viewing and selection of general parameters applied to all the slaves on the bus.</p> <ul style="list-style-type: none"> <li>● Automatic addressing,</li> <li>● Fallback mode.</li> </ul>
6	Slave configuration zone	<p>Allows viewing of the data associated with the selected slave, such as:</p> <ul style="list-style-type: none"> <li>● its profile,</li> <li>● a comment,</li> <li>● its AS-i symbol (the symbols are defined using the variables editor),</li> <li>● its parameters.</li> </ul>



## How to define a slave device on the AS-i bus

### At a Glance

The PL7 software offers a catalogue, which regroups all of the AS-i slaves that are available. This catalogue is structured in families (e.g.: Inductive detectors). The list of slave device families contains two distinct elements:

- Non-specialized products,
- Private family.

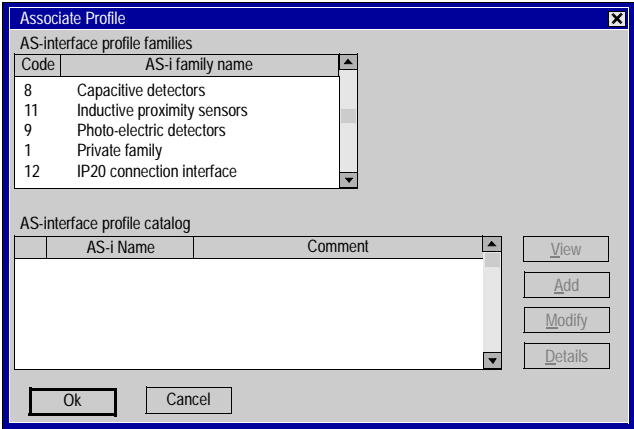
Selection of a non-specialized product allows the selection of an AS-i profile amongst 240 possibilities.

Selecting **Private family** gives the user the possibility of managing a specific AS-i device catalogue file via its programming terminal.

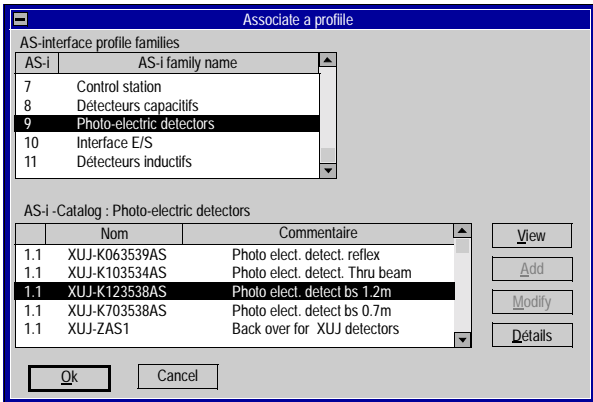
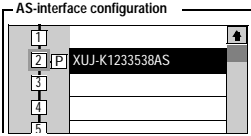
**Note:** An application using the AS-i products from the Private Family catalogue is always linked to the usage of this same private family catalogue.

### Procedure

This operation supports the declaration of a slave device on the AS-i bus.

Step	Action
1	Access the AS-i module's hardware configuration screen.
2	<p>In the field <b>AS-interface configuration</b>, double click in the cell corresponding to the new slave's (1 to 31) host slot number or select the said cell and then carry out the <b>Edit</b> → <b>Add a slave</b> command.</p> <p><b>Result:</b> The screen <b>Associate a profile</b> appears.</p> 



Step	Action
3	<p>In the <b>Family Name</b> field select the required family.  <b>Result:</b> The <b>Profile catalogue</b> associated with the selected family appears.</p> 
4	In the <b>Profile catalogue</b> select the required device.
5	<p>Confirm the selection with <b>OK</b>.  <b>Result:</b> The slave device is defined in its slot, the reference of the connected device appears opposite the number of the slave.</p> 
6	To connect other slave devices to the AS-i bus, repeat the procedure from step 2.



## How to modify the software configuration of the AS-i Bus

---

### At a Glance

The PL7 software offers, from the configuration screen of the AS-i module, a set of functions which allow you to easily modify, in local mode, the software configuration of the AS-i Bus.

---

### Procedure for deleting a slave

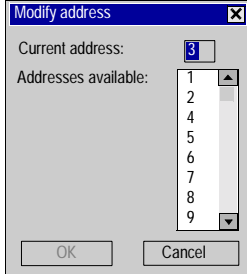
This operation is used to delete a slave declared on the AS-i Bus.

Step	Action
1	Select the slave to be deleted.
2	Select the command <b>Edit</b> → <b>Delete AS-i slave</b> .

---

### Procedure for moving a slave

This operation is used to move a slave declared on the AS-i Bus.

Step	Action
1	Select the slave to be moved.
2	Select the command <b>Edit</b> → <b>Modify AS-i slave address</b> .  <b>Result:</b> The following dialog box appears. 
3	Select the new address.
4	Confirm using <b>OK</b> .

---



---

## How to access the description of an AS-i slave

---

### At a Glance

The PL7 software allows access to all the information relating to an AS-i device such as:

- the definition of a profile,
  - the details of a profile.
- 

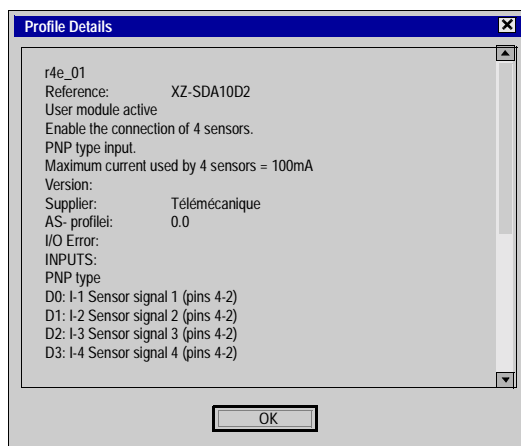
### Definition of a profile

A profile is defined by:

- its name,
  - a comment (optional),
  - identifiers (IO, ID),
  - a number of inputs and/or outputs,
  - operation parameters.
- 

### Details of a profile

The **Detail** function allows access, for a given slave, to all the information presented in the catalog file.





**Procedure for  
accessing  
information on a  
profile**

The following table shows the procedure for displaying the characteristics of a slave device.

Step	Action
1	Access the AS-i module's hardware configuration screen.
2	Double click on the required slave. <b>Result::</b> The window <b>Associate a profile</b> displays and highlights the device in question
3	Click on the button: <ul style="list-style-type: none"><li>● <b>View</b> to access definition information,</li><li>● <b>Detail</b> to access all the information.</li></ul>

---



## How to define a new slave profile in the standard AS-I catalogue

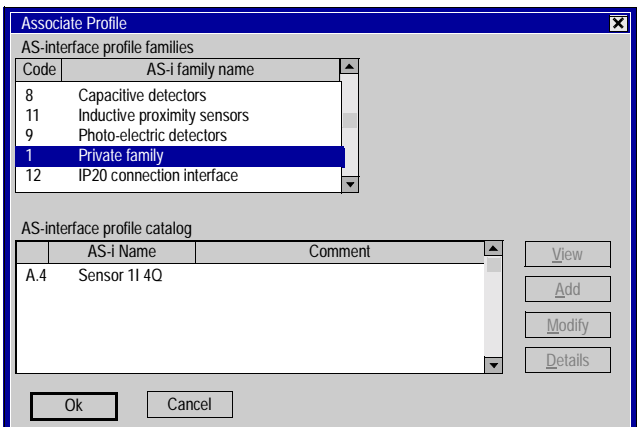
### At a Glance

The PL7 software offers the possibility to define a slave profile, which was not provided, in the standard catalogue.  
The new profile, defined in this way, is added to the catalogue in **Private family**.  
This profile can then be used as a standard catalogue profile.

**Note:** A profile cannot be deleted, only their name and comment can be changed.

### Procedure

The following table presents the procedure for defining a slave profile, which is not provided in the standard catalogue.

Step	Action
1	Access the AS-i module's hardware configuration screen.
2	Double click in a slave's host cell (1 to 31). <b>Result:</b> The screen <b>Associate a profile</b> appears.
3	Select <b>Private family</b> in the field <b>Family name</b> . <b>Result:</b> The <b>Profile catalog</b> linked to the selected family appears.
	 <p>The screenshot shows the 'Associate Profile' dialog box. It has two main sections. The top section, 'AS-interface profile families', contains a table with columns 'Code' and 'AS-i family name'. The rows are: 8 Capacitive detectors, 11 Inductive proximity sensors, 9 Photo-electric detectors, 1 Private family (highlighted), and 12 IP20 connection interface. The bottom section, 'AS-interface profile catalog', contains a table with columns 'AS-i Name' and 'Comment'. It has one row: 'A.4 Sensor 1I 4Q'. To the right of this table are four buttons: 'View', 'Add', 'Modify', and 'Details'. At the bottom of the dialog are 'Ok' and 'Cancel' buttons.</p>
4	Click on the button <b>Add</b> .
5	Enter: <ul style="list-style-type: none"> <li>the name of the new profile,</li> <li>a comment (optional).</li> </ul>



Step	Action
6	Select: <ul style="list-style-type: none"><li>● the <b>IO</b> code (corresponds to the input/output configuration),</li><li>● the <b>ID</b> code (identifier).</li></ul>
7	For each parameter define: <ul style="list-style-type: none"><li>● the system's acknowledgement (box checked),</li><li>● a label (optional).</li></ul>
8	Confirm the introduction of a new profile using <b>Confirm</b> .

---



## How to modify AS-i slave general parameters: Automatic addressing

### At a Glance

Each slave on the AS-i bus must be assigned (via configuration) a unique physical address. This must be the same as the one declared in PL7.

PL7 software offers an automatic slave addressing utility so that an AS-i console does not have to be used.

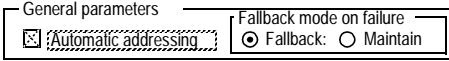
The automatic addressing utility, which can be accessed via PL7, is used for:

- replacing a faulty slave (See Automatic replacement of a faulty AS-i slave, p. 147),
- inserting a new slave (See How to insert a slave device into an existing AS-i configuration., p. 148).

**Note:** A new configuration with automatic addressing will not be accepted if one or more slaves with a 0 address are on the bus. In this case, the Configuration refused by module message appears.

### Procedure

The table below shows the procedure for setting the **Automatic addressing** parameter.

Step	Action
1	Access the AS-i communication module's configuration screen.
2	<p>Click on the <b>Automatic addressing</b> check box found in the <b>General parameters</b> zone.</p> <p><b>Result:</b> The <b>Automatic addressing</b> utility will be activated (box checked) or disabled (box not checked).</p>  <p><b>Note:</b> By default, the <b>Automatic addressing</b> parameter has been selected in the configuration screen.</p>



## How to modify AS-i slave general parameters: Fallback mode

### At a Glance

This parameter sets the fallback mode which slave outputs take on while switching to STOP mode or if there is a PLC fault.


Possible modes are:

- **Fallback to 0:** AS-i slave outputs which are present on the bus are set to 0 (%Q objects are not modified),
- **Maintain state:** AS-i slave outputs remain in the state they were before STOP mode was engaged.

**Note:** The fallback mode for slaves which do not have a watchdog (AS-i bus monitoring function) is not guaranteed in case of an AS-i bus fault, or AS-i power supply failure. For slaves with a watchdog, the fallback position is preset within the device.

### Procedure

The table below shows the procedure for setting the **Fallback mode** assigned to slave device outputs.

Step	Action
1	Access the AS-i communication module's configuration screen.
2	<p>Click on the <b>Fallback to 0 / Maintain state</b> check-box found in the <b>General parameters</b> zone, in the <b>Fallback mode on fault</b> field.</p> <p><b>Result:</b> The fallback mode selected will therefore be assigned to the slave device.</p> 



---

# Debugging the AS-i bus

# 11

---

## At a Glance

### Subject of this Chapter

This Chapter describes the Debug aspect of the AS-i bus.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Introduction to the Debug function	138
Description of the debug screen for an AS-i module	139
How to access functionality in the module diagnostics and channel diagnostics for an AS-i device	141
Displaying the slaves' status	143
How to access adjustment of an AS-i device's parameters	144
How to access the AS-i channels' forcing/unforcing function	145
How to access the SET and RESET commands of the AS-i channels	146
Automatic replacement of a faulty AS-i slave	147
How to insert a slave device into an existing AS-i configuration.	148



## Introduction to the Debug function

---

### At a Glance

The **Debug** function is used for each AS-i communication module present in the application:

- to display the slave state (connection, parameters etc.),
- to access the adjustment function for the selected channel (channel forcing etc.).

The function also accesses module diagnostics in the event of a fault.

<b>Note:</b> This function is only available in online mode.
--

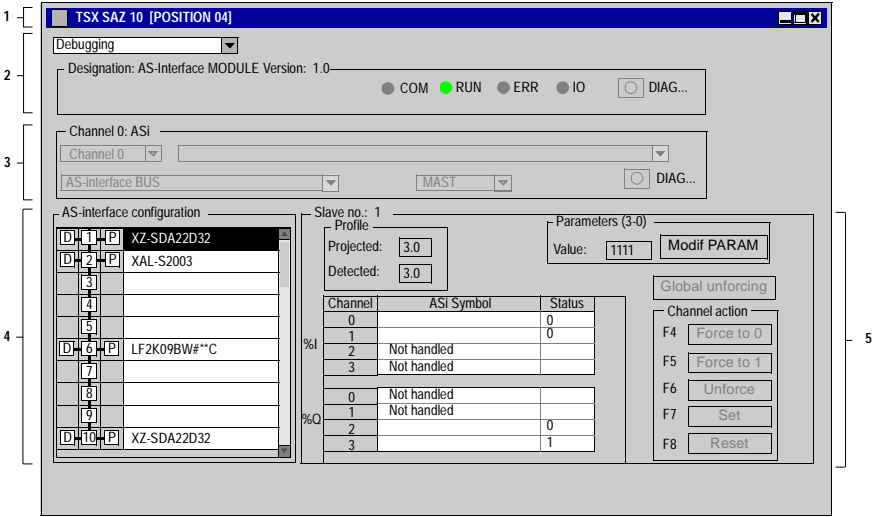
---



## Description of the debug screen for an AS-i module

**At a Glance**      The debug screen dynamically displays the status of the AS-i module and the devices connected on the bus.  
It is also used to access slave parameter adjustment and channel commands (forcing the value of an input or output, Set/Reset of an output, etc.).

**Illustration**      The debug screen looks like this:





**Description**            The table below shows the different elements of the debug screen and their functions.

Number	Element	Function
1	Title bar	Indicates the reference of the module selected and its physical position in the PLC.
2	Module zone	Allows the selection of the parameter type: <ul style="list-style-type: none"><li>● <b>Configuration</b>,</li><li>● <b>Debugging</b> (diagnostic), only accessible in connected mode.</li></ul> Displays the status of the module lights <b>COM</b> , <b>RUN</b> , <b>ERR</b> , <b>I/O</b> . Provides direct access to the module diagnostics when a module is faulty (signaled by the light integrated in the diagnostics access button <b>DIAG</b> , which turns red).
3	Channel zone	Provides direct access to the channel diagnostics when a channel is faulty (signaled by the light integrated in the diagnostics access button <b>DIAG</b> , which turns red).
4	AS-i configuration zone	Displays the slave devices connected to the bus.
5	Slave zone	Displays the status of the slave channels and gives access to the debug functions.

---



## How to access functionality in the module diagnostics and channel diagnostics for an AS-i device

---

### At a Glance

The functions of the module or channel diagnostics display the current errors in which are classed according to their category:

- internal errors (internal software errors, communication error with the processor, configuration, parametering or command error),
- external errors (slave device failed, AS-i supply switched off, terminal error, difference between physical configuration and PL7 configuration),
- other errors (module absent or switched off).

A faulty module becomes apparent when certain lights change to red, such as:

- in the configuration editor at rack level:
  - the module position light,
- in the configuration editor at module level:
  - the lights **RUN**, **ERR** and **I/O**,
  - the light **DIAG**.

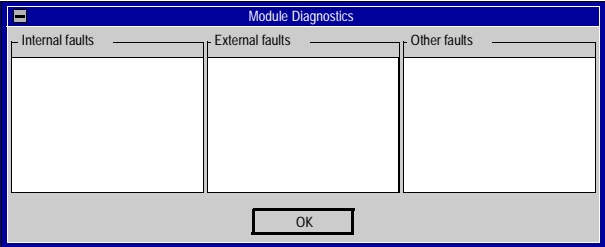
A faulty channel becomes apparent when certain lights change to red, such as:

- in the configuration editor at rack level:
    - the module position light,
  - in the configuration editor at channel level:
    - the light **DIAG**.
-



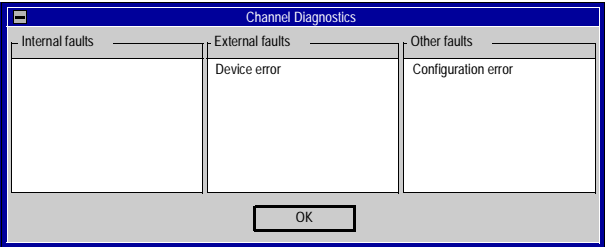
Procedure for accessing the module diagnostics

The following table presents the procedure to access the screen **Module diagnostics**.

Step	Action
1	Access the AS-i module's hardware configuration screen.
2	Click on the button <b>DIAG</b> situated in the module zone. <b>Result:</b> The list of module errors appears. <div></div>

Procedure for accessing the channel diagnostics

The following table presents the procedure for accessing the screen **Channel diagnostics**.

Step	Action
1	Access the AS-i module's hardware configuration screen.
2	Click on the button <b>DIAG</b> situated in the channel zone. <b>Result:</b> The list of channel errors appears. <div></div>



## Displaying the slaves' status

### At a Glance

The lower part of the communication module's debugging screen is reserved for diagnostics of the AS-i bus.

The slave devices connected to the bus are displayed in the **AS-i Configuration** zone. On each side of the slave number, two different icons are displayed, indicating that the slave was specified or detected.

**Displaying slave status** For each slave device, one of the following four cases can occur:

Case	Illustration	Explanation
1	<p>Slave status:</p> <p>AS-interface configuration</p>	The slave <b>P</b> specified in configuration and the detected slave <b>D</b> are identical.
2	<p>Slave status:</p> <p>AS-interface configuration</p>	The slave <b>P</b> specified in configuration and the detected slave <b>D</b> are not identical. The slave is declared faulty (1).
3	<p>Slave status:</p> <p>AS-interface configuration</p>	A slave <b>P</b> is specified at configuration but no slave is detected. The slave is declared faulty (1).
4	<p>Slave status:</p> <p>AS-interface configuration</p>	An additional slave, not specified at configuration, is connected to the bus. The slave is declared faulty (1).
<b>Key:</b>		
(1)	When a slave is faulty, the icons situated beside the number as well as the button <b>DIAG</b> turn to red.	
<b>Note:</b> The <b>Profile</b> field in the <b>Slave zone</b> of the debugging screen allows you to check if the profiles of the specified ( <b>Projected</b> ) slave and the <b>Detected</b> slave are really identical.		



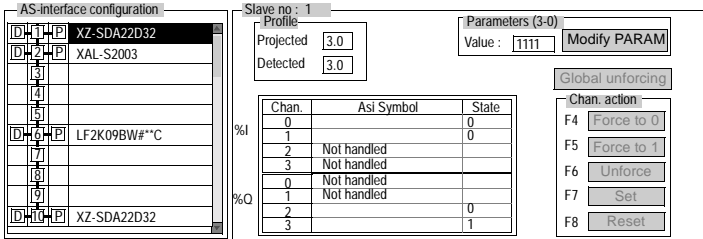
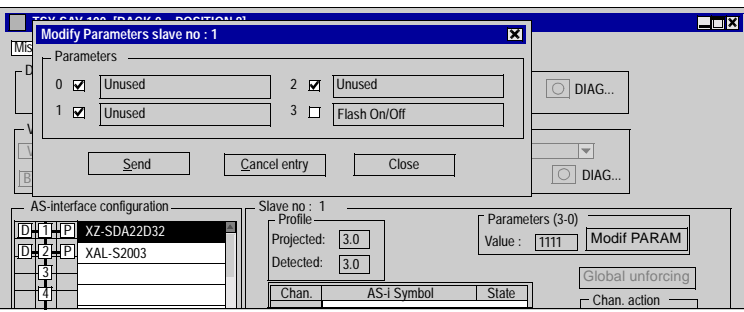
## How to access adjustment of an AS-i device's parameters

### At a Glance

The debugging screen of an AS-i module allows, amongst other things, access to modification of a slave's parameters.

### Procedure

The following table shows the procedure for modifying the parameters of a slave device declared faulty.

Step	Action
1	Access the AS-i module's debugging screen.
2	<p>Select the slave with the error.</p> <p><b>Result:</b> In the slave zone of the debugging screen, it is possible to read all the information relating to the selected slave.</p>  <p>The screenshot shows the 'AS-interface configuration' table with columns ID, P, and Name. The table lists several slaves, including XZ-SDA22D32 and XAL-S2003. The 'Slave no: 1' section displays the 'Profile' (Projected: 3.0, Detected: 3.0) and 'Parameters (3-0)' (Value: 1111). A 'Global unforcing' section is also visible with buttons for 'F4 Force to 0', 'F5 Force to 1', 'F6 Unforce', 'F7 Set', and 'F8 Reset'.</p>
3	<p>Click on the button <b>Modif PARAM</b> situated in the slave zone's <b>Parameters</b> field.</p> <p><b>Result:</b> The window <b>Parameter modification</b> appears.</p>  <p>The screenshot shows the 'Modify Parameters slave no: 1' dialog box. It contains a 'Parameters' section with four checkboxes labeled 0, 1, 2, and 3. Checkboxes 0 and 1 are checked, and checkboxes 2 and 3 are unchecked. There are also buttons for 'Send', 'Cancel entry', and 'Close'. The background shows the same AS-i debugging screen as in step 2.</p>
4	Modify the required parameters.
5	Click on <b>Send</b> to recognize the new values.



## How to access the AS-i channels' forcing/unforcing function

### Introduction

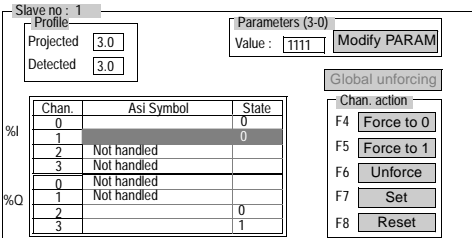
This function supports the modification of the state of the channels, which are linked to the AS-i slave.

The different commands, which are available are:

- for a channel:
  - forcing to 0,
  - forcing to 1,
  - unforcing.
- for all the channels (when at least one channel is forced):
  - global unforcing.

### Procedure

The following table shows the procedure to follow to force or unforce the channels linked to an AS-i slave.

Step	Action for a channel	Action for a group of channels
1	Access the AS-i interface card's debugging screen.	
2	Select a slave in the zone <b>AS-i Configuration</b> .	
3	<p>Select the channel to modify in the slave zone table.  <b>Result:</b> It is possible to modify the channel using the buttons situated in the field <b>Channel action</b> .</p>  <p>The screenshot shows the 'Slave no: 1' configuration. It includes a 'Profile' section with 'Projected' and 'Detected' values of 3.0. A 'Parameters (3-0)' section shows a 'Value' of 1111 and a 'Modify PARAM' button. A 'Global unforcing' button is also present. The 'Chan. action' section contains buttons for 'Force to 0', 'Force to 1', 'Unforce', 'Set', and 'Reset'. A table lists channels 0 through 3 with their 'Asi Symbol' and 'State'.</p>	Click on the button <b>Global unforcing</b> found in the slave zone.
4	Select the required function (buttons <b>Force to 0</b> or <b>Force to 1</b> in the field <b>Channel Action</b> ).	



## How to access the SET and RESET commands of the AS-i channels

### At a Glance

These commands allow the assignment of the values 0 (RESET) or 1 (SET) to the channels of an AS-i slave.

The status of the output affected by one of these commands is temporary and can be modified at any time by the application.

### Procedure

The following table presents the procedure for assigning a value of 0 or 1 to the selected AS-i slave's channels.

Step	Action																											
1	Access the AS-i module's debugging screen.																											
2	Select a slave in the zone <b>AS-i Configuration</b> .																											
3	<p>Select the channel to be modified in the <b>Slave</b> zone table.</p> <p><b>Result:</b> It is possible to modify the channel using the buttons situated in the field <b>channel action</b>.</p> <div><div><div>Slave no : 1</div><div>Profile</div><div><div>Projected3.0</div><div>Detected3.0</div></div></div><div><div>Parameters (3-0)</div><div>Value : 1111</div><div>Modify PARAM</div></div><div><div>Global unforcing</div><div>Chan. action</div><div><div>F4Force to 0</div><div>F5Force to 1</div><div>F6Unforce</div><div>F7Set</div><div>F8Reset</div></div></div><div><table><thead><tr><th>Chan.</th><th>Asi Symbol</th><th>State</th></tr></thead><tbody><tr><td>0</td><td></td><td>0</td></tr><tr><td>1</td><td></td><td>0</td></tr><tr><td>2</td><td>Not handled</td><td></td></tr><tr><td>3</td><td>Not handled</td><td></td></tr><tr><td>0</td><td>Not handled</td><td></td></tr><tr><td>1</td><td>Not handled</td><td></td></tr><tr><td>2</td><td></td><td>0</td></tr><tr><td>3</td><td></td><td>1</td></tr></tbody></table></div></div>	Chan.	Asi Symbol	State	0		0	1		0	2	Not handled		3	Not handled		0	Not handled		1	Not handled		2		0	3		1
Chan.	Asi Symbol	State																										
0		0																										
1		0																										
2	Not handled																											
3	Not handled																											
0	Not handled																											
1	Not handled																											
2		0																										
3		1																										
4	Select the required function (buttons <b>Set</b> or <b>Reset</b> ) in the field <b>Channel action</b> .																											



## Automatic replacement of a faulty AS-i slave

---

### Principle

When a slave has been declared faulty, it can be automatically replaced with a slave of the same type.

This happens without the AS-i bus having to stop, and without any particular manipulation since the configuration mode **Automatic addressing** utility is active (see How to modify AS-i slave general parameters: Automatic addressing, p. 135).

Two options are available:

- The replacement slave is programmed with the same address using the pocket programmer, and has the same profile as the faulty slave. It will then be automatically inserted into the list of detected slaves (LDS) and list of active slaves (LAS).
  - the replacement slave is blank (address 0, new slave) and has the same profile as the faulty slave. It will automatically assume the address of the replaced slave, and will then be inserted into the list of detected slaves (LDS) and the list of active slaves (LAS).
-



## How to insert a slave device into an existing AS-i configuration.

---

### At a Glance

It is possible to insert a device into an existing AS-i configuration without having to use the pocket programmer.

This operation is possible as soon as:

- the service **Automatic addressing** of the configuration mode is active (See How to modify AS-i slave general parameters: Automatic addressing, p. 135),
- a single slave is absent in the physical configuration,
- the slave which is to be inserted is projected in the PL7 configuration,
- the slave has the profile expected by the configuration,
- the slave has the address 0.

Therefore the AS-i interface card will automatically assign to the slave the value predefined in the configuration.

---

### Procedure

The following table presents the procedure for making the automatic insertion of a new slave effective.

Step	Action
1	Add the new slave in the configuration screen in local mode.
2	Do a configuration transfer to the PLC in connected mode.
3	Physically link the new slave with address 0 to the AS-i bus.

<b>Note:</b> It is possible to modify an application by carrying out the above manipulation as many times as necessary.
---

---



---

# Bits and words associated with the AS-i function

12

---

## At a Glance

### Subject of this Chapter

This Chapter introduces the different word and bit objects associated with the AS-i function, as well as how to address them.

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
12.1	Addressing objects associated with the AS-i function	151
12.2	Language objects associated with the AS-i function	152







# 12.1      Addressing objects associated with the AS-i function

## Addressing language objects associated with slave devices connected on the AS-i Bus

**At a Glance**      The addressing of bit and word objects associated with application specific tasks is defined in the part entitled Shared applications.  
This page introduces the specific features linked to the AS-i function.

**Illustration**      Reminder of the principles of addressing:

%	I or Q	\	4.0	\	n	•	i
Symbol	Type of object		TSX SAZ 10 channel/ module address		Number of slaves		Position of bit

**Specific values**      The table below gives the values specific to the AS-i slave objects.

Element	Values	Comment
n	0 to 31	Slot 0 cannot be configured.
i	0 to 3	-



# 12.2                    Language objects associated with the AS-i function

## At a Glance

**Subject of this Section**                    This Section introduces the different language objects associated with the AS-i function.

**What's in this Section?**                    This Section contains the following Maps:

Topic	Page
Implicit exchange objects associated with the AS-i function	153
Managing exchanges: Module %MW4.MOD.0:Xj or channel %MW4.0.0:Xj exchanges in progress	154
Managing exchanges: Module %MW4.MOD.1:Xj or channel %MW4.0.1:Xj report	155
Explicit exchange objects: General	156
Explicit exchange objects: Channel status %MW4.0.2:Xj to %MW4.0.22:Xj	157
Explicit exchange object: Channel command %MW4.0.23:Xj	159
Explicit exchange objects: Adjustment parameters %MW4.0.24 to %MW4.0.39	160
Explicit exchange objects: Status %MW4.MOD.2:Xj	161



## Implicit exchange objects associated with the AS-i function

### At a Glance

These are the objects whose exchanges are carried out automatically on each cycle of the task in which the AS-i function is configured.

### Bit objects

The table below shows the different implicit exchange bit objects.

Address	Function	Meaning when the bit is set to 1
%I4.0.ERR	Channel error bit	Indicates a supply fault or a slave missing on the AS-i bus.
%I4.MOD.ERR	Module error bit	Indicates the module is faulty.
%I4.0	Input validation	Indicates that all inputs are valid. Note: When this bit is set to 0, it indicates that at least one input is not valid: offline mode or channel error.
%I4.0.i	Reserved	-
%I4.0\n.i	input channel bit	Indicates that the input channel i of the device n is activated.
%Q4.0\n.i	output channel bit	Indicates that the output channel i of the device n is activated.
%Q4.0	Reserved	-
%Q4.0.i	Reserved	-

### Word objects

The table below shows the different implicit exchange word objects.

Address	Function	Meaning when $X_j = 1$ (j = position of the bit in the word)
%IW4.0 %IW4.0.1	List of slaves in error	j = 0 to 15 -> respectively slave 0 to 15 in error or missing. j = 0 to 15 -> respectively slave 16 to 31 in error or missing.
%IW4.0.2 %IW4.0.3	List of activated slaves (LAS)	j = 0 to 15 -> respectively slave 0 to 15 activated. j = 0 to 15 -> respectively slave 16 to 31 activated.



## Managing exchanges: Module %MW4.MOD.0:Xj or channel %MW4.0.0:Xj exchanges in progress

### At a Glance

These word type objects carry information about the module or channel exchanges in progress.  
They are updated automatically by the system.

### Description

The table below gives the meaning of the different bits of the word %MW4.MOD.0.

Address	Meaning when Xj = 1
%MW4.MOD.0:X0	Exchange of status words in progress on the module channel.
%MW4.MOD.0:X1	Exchange of command words in progress on the module channel.

### Description

The table below gives the meaning of the different bits of the word %MW4.0.0.

Address	Meaning when Xj = 1
%MW4.0.0:X0	Exchange of status words in progress on the AS-i channel.
%MW4.0.0:X1	Exchange of command words in progress on the AS-i channel.

### Example

The example below shows a possible use for this type of word.

```
(* Request for update of the status words of channel 0 *)
(* of the AS-i module if no exchange is in progress on this
channel*)
IF NOT %MW4.0:X0 THEN READ_STS %CH4.0;
END_IF;
```

**Note:** When the period of the explicit exchange is shorter than the cycle time of the PLC task, the bit %MW4.0:X0 never changes to 1.



---

## Managing exchanges: Module %MW4.MOD.1:Xj or channel %MW4.0.1:Xj report

---

**At a Glance** These word type objects contain information on the module or channel exchange reports.  
They are updated automatically by the system.

---

**Description** The table below gives the meaning of the different bits of the word **%MW4.MOD.1**.

Address	Meaning when Xj = 1
%MW4.MOD.1:X0	Status parameter exchange error on channel 0 of the module.
%MW4.MOD.1:X1	Command parameter exchange error on channel 0 of the module.

---

**Description** The table below gives the meaning of the different bits of the word **%MW4.0.1**.

Address	Meaning when Xj = 1
%MW4.0.1:X0	Status parameter exchange error on AS-i channel.
%MW4.0.1:X1	Command parameter exchange error on AS-i channel.

---

**Example** The example below shows a possible use for this type of word.

```
(* Detection of a status error on the module *)
(* situated in slot 4 *)
IF NOT %MW4.MOD.0:X0 THEN READ_STS %CH4.MOD;
END_IF;
IF %MW4.MOD.1:X0 THEN SET %M100;
END_IF;
```

---



## Explicit exchange objects: General

---

### At a Glance

Explicit exchange objects carry information (e.g.: bus operation, slave status, etc.) and additional commands to carry out advanced programming of the AS-i function.

**Note:** The configuration constants %KW4.0.r, not documented in this manual, can only be accessed in read mode and correspond to the configuration parameters entered using the Configuration editor.

Explicit exchange objects are exchanged at the request of the user program using the instructions:

- READ\_STS (read status words)
  - WRITE\_CMD (write command words),
  - WRITE\_PARAM (write adjustment parameters),
  - READ\_PARAM (read adjustment parameters),
  - SAVE\_PARAM (save adjustment parameters),
  - RESTORE\_PARAM (restore adjustment parameters).
-



## Explicit exchange objects: Channel status %MW4.0.2:Xj to %MW4.0.22:Xj

### At a Glance

These word type objects provide information on all the slaves present on the AS-i Bus.

### Description of the word %MW4.0.2

The table below gives the meaning of the different bits of the word %MW4.0.2.

Address	Function	Meaning when Xj = 1
%MW4.0.2:X0	Standard status	Reserved
%MW4.0.2:X1		One or several slaves in error.
%MW4.0.2:X2		Line error (supply switched off or terminal block error).
%MW4.0.2:X3		Physical configuration different to the PL7 configuration.
%MW4.0.2:X4		Internal software error.
%MW4.0.2:X5		Reserved
%MW4.0.2:X6		Error communicating with the processor.
%MW4.0.2:X7		Parameterizing or command configuration error.

### Description of the word %MW4.0.3

The table below gives the meaning of the different bits of the word %MW4.0.3.

Address	Function	Meaning when Xj = 1
%MW4.0.3:X0	AS-i specific status	Correct configuration.
%MW4.0.3:X1		Slave 0 present.
%MW4.0.3:X2		Automatic addressing active.
%MW4.0.3:X3		Reserved.
%MW4.0.3:X4		Reserved.
%MW4.0.3:X5		Reserved.
%MW4.0.3:X6		AS-i supply failed.
%MW4.0.3:X7		Offline phase active.
%MW4.0.3:X8		Reserved.



**Description of the words %MW4.0.4 and %MW4.0.5**

The table below gives the meaning of the different bits of the words **%MW4.0.4** and **%MW4.0.5**.

Address	Function	Meaning when Xj = 1
%MW4.0.4:Xj	List of detected slaves LDS	j = 0 to 15 -> respectively slave 0 to 15 detected.
%MW4.0.5:Xj		j = 0 to 15 -> respectively slave 16 to 31 detected.

**Description of the words %MW4.0.6 to %MW4.0.21**

The table below gives the meaning of the different bits of the words **%MW4.0.6** to **%MW4.0.21**.

Address	Function	Meaning
%MW4.0.6 to %MW4.0.21	I/O and ID configuration for all detected slaves	Words 6 to 21 -> respectively devices 0-1, 2-3, ..., 28-29, 30-31. The least significant bytes relate to the slaves with an even address. The most significant bytes relate to the slaves with an odd address. For each byte: bit 0-3 = configuration code for input/output (I/O) channels, bit 4-7 = identification code (ID).

**Description of the word %MW4.0.22**

The table below gives the meaning of the word **%MW4.0.22**.

Address	Function	Meaning
%MW4.0.22	Parameterizing data of last slave parameterized.	Contains the reply (value of the parameters sent) from the last slave parameterized. This allows you to verify via PL7 that the slave has received them correctly.



**Explicit exchange object: Channel command %MW4.0.23:Xj**

---

**At a Glance** This type of object word is used to manage the AS-i master's change to Offline mode (See AS-i offline operating mode, p. 168).

---

**Description** The table below shows the coding of the bits 0 and 1 for the word **%MW4.0.23** which gives access to Offline mode.

Bit 1	Bit 0	Function
0	0	Normal operating mode
0	1	Activation of offline mode
1	0	Deactivation of offline mode
1	1	No change

---



**Explicit exchange objects: Adjustment parameters %MW4.0.24 to %MW4.0.39**

---

**At a Glance**                    These objects are used to manage the parameters of the AS-i slave devices.  
They can be modified without stopping the AS-i function.

---

**Description**                    The table below shows the adjustment objects for the AS-i channel.

Address	Function	Meaning
%MW4.0.24 to %MW4.0.39	Adjustment of parameters	Contains respectively the value of the parameters for the slaves 1 to 31. Least significant byte: parameters for slaves with an even address, Most significant byte: parameters for slaves with an odd address.

---



---

## Explicit exchange objects: Status %MW4.MOD.2:Xj

---

**At a Glance** This word type object contains information about the status of the module.

---

**Description** The table below gives the meaning of the different bits of the word %MW4.MOD.2:Xj.

Address	Function	Meaning when Xj = 1
%MW4.MOD.2:X0	Standard module status	Internal error.
%MW4.MOD.2:X1		Configuration error.
%MW4.MOD.2:X2		Line fault.
%MW4.MOD.2:X3		Not used.
%MW4.MOD.2:X4		Not used.
%MW4.MOD.2:X5		Not used.
%MW4.MOD.2:X6		Module missing.
%MW4.MOD.2:X7		Not used.

---







At a Glance

Subject of this Chapter

This Chapter introduces the different AS-i function operating modes.

What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
AS-i operating mode: General	164
AS-i protected mode	166
AS-i wiring test mode	167
AS-i offline operating mode	168



## AS-i operating mode: General

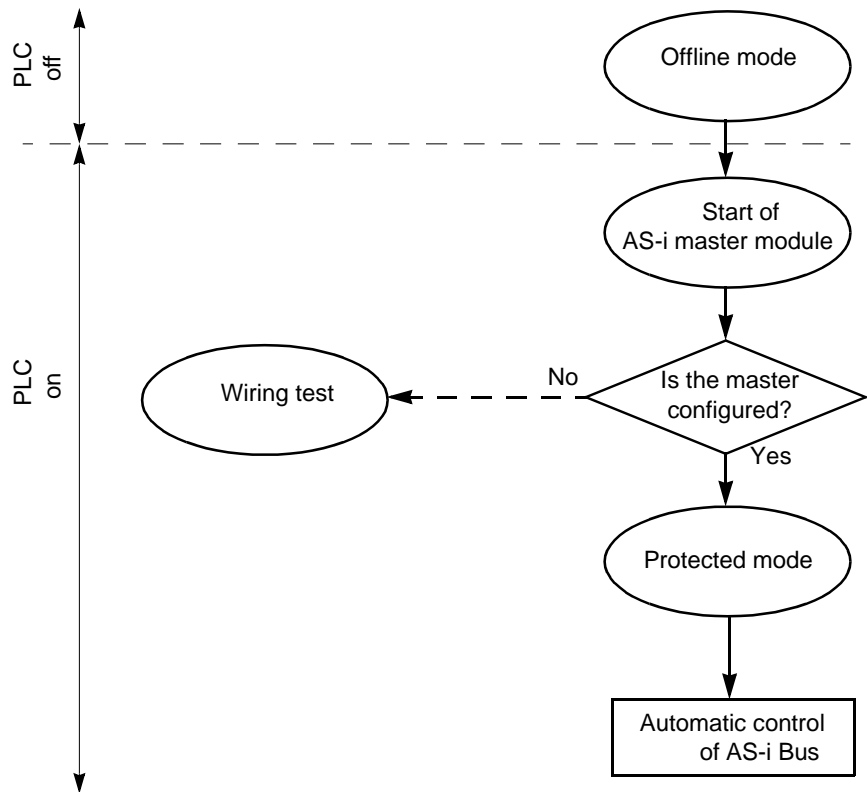
### At a Glance

The AS-i function allows three operating modes, each of which responds to particular needs. These modes are:

- protected mode,
- wiring test mode (which can be accessed using the adjustment terminal FTX 117 Adjust),
- offline mode.

### Operating mode

The diagram below shows the general function diagram of how the AS-i Bus operates.





**Relationship  
between the PLC  
and AS-i Bus  
operating modes**

The table below shows the relationship between the PLC TSX 37 operating modes and those of the AS-i bus.

PLC	AS-i Bus
Configured mode (1)	Protected mode
Non configured mode (1)	Wiring test mode (Configuration mode)
<b>Key:</b>	
(1)	These PLC concepts (configured, non configured) relate to the declaration of the module and the slave devices in the hardware configuration screen in the PL7 application.



## AS-i protected mode

---

### At a Glance

The AS-i protected operating mode is the mode generally used for an application which is running.

It is assumed that the TSX SAY 100 module is configured in PL7.

This:

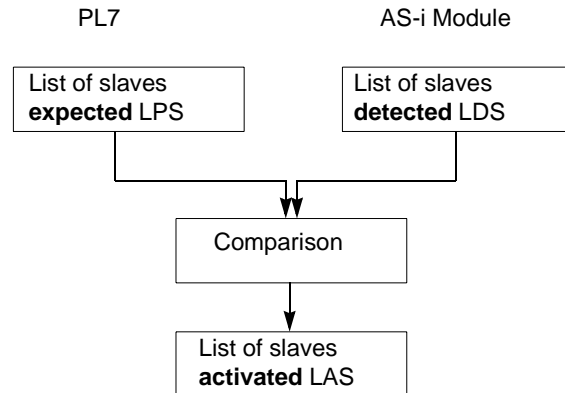
- continually checks that the list of detected slaves is the same as the list of expected slaves.
- monitors the power supply.

In this mode, a slave will only be activated if it has been declared in the configuration and been detected.

---

### Principle of activating a slave

The schema below show the principle of activating AS-i slaves.





## AS-i wiring test mode

### At a Glance

AS-i wiring test mode is used to access the input/output bits of slaves connected to the bus.

**Note:** The LDS and LAS lists, as well as the adjustment parameters, cannot be accessed.

The wiring test requires the adjustment terminal FTX 117 Adjust to be used.

**Note:** When accessing slave I/O objects using the FTX 117 Adjust terminal, personnel safety must be maintained while activating accessible objects, in particular slave outputs.

### Access conditions

This mode can be accessed when the PLC is in a "non-configured" state. This means when:

- the PLC does not have an application,  
or
- the AS-i module has not been configured.

### Procedure

The table below describes the procedure for accessing and exiting Wiring test mode from the adjustment terminal.

Step	Action	Comment
1	Set bit %S8 to 0	Enables wiring test mode.
2	Set bit %S74 to 1	Saves, if necessary, the current module configuration.
3	Set bit %S73 to 1	Activates the module in protected mode.
4	Access I/O via the %I fields for inputs and %Q fields for outputs.	
5	Reset bit %S74 to 0	This bit should normally be set to 0.
6	Set bit %S8 to 1	Disables wiring test mode

**Note:** In this mode, all slaves must have a different address to avoid conflict when replying to the module.



## AS-i offline operating mode

---

### At a Glance

AS-i offline mode is an advanced operating mode, which can be used for debugging or maintenance.

<b>Note:</b> Its use requires good knowledge of AS-i communication.
---

---

### Principles

When the module is put into offline mode, it first resets all the slaves present to zero and stops exchanges on the bus.

During offline mode, the I/O images in the module are frozen in the state in which they were at the time of the change to the mode.

When the module exits, if the list of slaves present (LPS) equals the list of slaves detected (LDS), the system restarts. If this is not the case, a fault is generated, and it then necessary to transfer to diagnostic or configuration mode.

---

### Procedure

Offline operating mode can be accessed either:

- from the PL7 software application by acting on the 0 and 1 bits of the word %MW4.0.23,
- automatically on detecting an AS-i supply fault.

**Reminder:** Bit 7 of the word %MW4.0.3 set to 1 indicates that the AS-i Bus is in **offline** mode.

---



---

## AS-i bus performance

**At a Glance**

The AS-i bus is independently managed by the master. This exchanges data on each cycle with each slave device configured on the bus (in ascending order of slave address number).

**AS-i scanning time**

**t** scanning time represents the exchange time between the master and **n** slaves (31 maximum).

Either:

- $t = 156 \text{ micro seconds} \times (n+2)$ , if  $n < 31$ ,
- $t = 156 \text{ micro seconds} \times (n+1)$ , if  $n = 31$ .

Thus the scanning time cannot exceed 5 ms.

**AS-i response time**

**T** response time represents the AS-i cycle time.

This includes:

- the bus scanning time,
- the update of the AS-I module internal memory,
- the PLC cycle.

**Example**

The table below shows three examples of T response time for a PLC task lasting 10 ms, 30 ms, and 60 ms.

This T time is for a bus loaded with 31 slaves operating normally with no link faults.

PLC task	Typical response time	Maximum response time
10 ms	35 ms	56 ms
30 ms	65 ms	96 ms
60 ms	110 ms	156 ms







---

# Analog application



# IV

---

## At a Glance

**Aim of this part** This part introduces the application-specific analog function on Micro PLCs and describes its implementation using PL7 software.

**What's in this part?** This Part contains the following Chapters:

Chapter	Chaptername	Page
15	The analog application	173
16	Built-in analog interface	175
17	Analog input modules TSX AEZ 801 / TSX AEZ 802	187
18	Analog input modules TSX AEZ 414	197
19	Analog output module TSX ASZ 401	209
20	Analog output module TSX ASZ 200	213
21	TSX AMZ 600 Analog Module	219
22	Configuring the analog application	231
23	Configuration of analog channels	241
24	Debugging function	261
25	Bits and words associated with the analog application	273







---

### Introduction to the analog application on Micro

---

#### At a Glance

The application-specific analog function applies to:

- the analog interface built in to the Micro TSX 37-22 PLC
- the analog input/output modules inserted in a Micro PLC.

Before creating an application program, the physical operating environment in which it will be executed must be defined. In other words, the processor type and the input/output modules used.

The use of analog inputs/outputs (built-in analog interface TSX 37-22 or analog modules), also requires the definition of the analog channel parameters used (input range, filtering level, etc.).

For this, the PL7 Micro software has the configuration editor (See Display of Channel Parameters, p. 246) which is used to easily perform these operations.

When the application is working online this editor also has a debug function (See Display of Channel Parameters, p. 263) which is used to adjust certain parameters (filtering, for example) so as to best adapt them to the application.

---







---

# Built-in analog interface

# 16

---

## Chapter overview

### What's in this Chapter?

This chapter introduces the analog interface built-in to the Micro TSX 37-22 PLC.

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
16.1	Built-in analog interface	177
16.2	Input processing	179
16.3	Output processing	185







## 16.1 Built-in analog interface

### Introducing the built-in analog interface

#### General

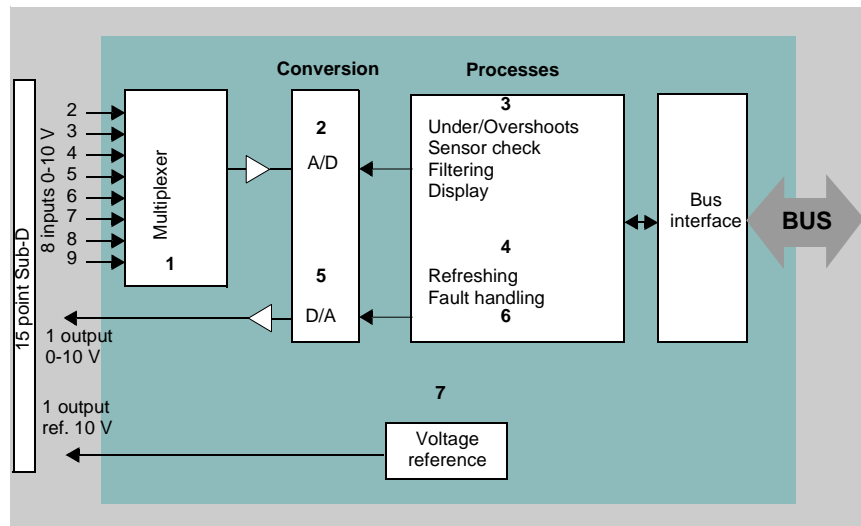
TSX 37-22 PLCs incorporate as standard an analog interface which contains:

- Eight 0-10 V input channels (convertible to 0-20 mA or 4-20 mA inputs by using shunts built in to the TSX ACZ 03 module)
- One 0-10 V output channel

This interface can meet the demands of applications which require analog processing, but where the performance and characteristics of an industrial measurement device are not justified.

#### Diagram

Graphic representation of the interface functions:





**Functions**

The table below details the interface functions:

Numbers	Functions
1	Scanning input channels using static multiplexing and value acquisition.
2	Analog/digital conversion over 8 bits (256 pulses) successive approximation of input measurements.
3	Filtering input measurements.
4	Refreshing the digital output value by the processor.
5	Digital/analog conversion of the output value over 8 bits (256 pulses).
6	Handling of processor dialog faults and particularly setting the output to fallback.
7	Providing a reference voltage for external potentiometers or those in the adjustment and adaptation module TSX ACZ 03.

---



---

## 16.2 Input processing

---

### At a Glance

#### What's in this Section?

This section introduces the different functions of the analog inputs of the interface built in to the Micro PLC.

#### What's in this Section?

This Section contains the following Maps:

Topic	Page
Timing of measurements	180
Under/overshoot monitoring on inputs	181
Sensor link monitoring	182
Measurement filtering	183
Displaying measurements	184



## Timing of measurements

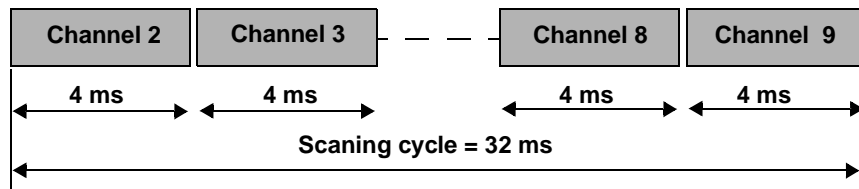
### General

The timing of measurements depends on the cycle used, defined at configuration (See Modification of the Scanning Cycle, p. 251): normal cycle or fast cycle.

### Normal cycle

The input scanning cycle is fixed and has a value of 32 ms, independent of the number of inputs used.

Example of a scanning cycle using only channels 2, 3, 8, and 9:



### Fast cycle

Only the channels used are scanned, even if these are not consecutive. This improves the cycle time for scanning the channels.

The cycle time for scanning the channels is given by the formula:

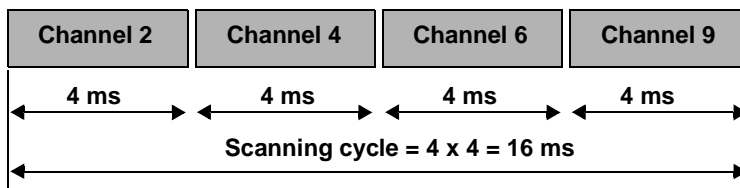
$$\text{Cycle time (ms)} = 4 \text{ ms} \times N$$

$N$  = number of channels used

For example, if 4 channels are used, the scanning cycle time will be:

$$4 \times 4 = 16 \text{ ms}$$

Illustration:



**Note:** In fast cycle, the user can assign the channels to FAST task (See Modification of the Task Assigned to the Module Inputs, p. 252)



## Under/overshoot monitoring on inputs

### At a Glance

For each of the analog inputs the interface carries out a range overshoot check, by checking that the measurement remains below the upper limit. If this is not the case, saturation of the measurement string is likely. An overshoot fault is signaled by a bit which can be used by the system.

Inputs allow a range overshoot of 2% of the upper limits on the full scale.

### Overshoot values

The table below gives the measurement limits for each range:

Range	Lower limit	Upper limit	Whole values available by default
0...10 V	-	+ 10.2 V	0...10200
0...20 mA	-	+ 20.4 mA	0...10200
4...20 mA	3.2 mA	+ 20.4 mA	- 500... + 10250

**Note:** The interface only detects an overshoot on the upper limit in the range 4-20 mA: 3.2 mA corresponds to - 500.

### Under/overshoot identifiers

In the over/undershoot zones, there is a risk of measurement string saturation. On under/overshoot, the acquisition of inputs continues, but they are signaled as invalid.

Limit under/overshoots are signaled by the following bits:

Bit name	Meaning (when = 1)	Exchange type
%I0.i.ERR	Channel i fault	Implicit
%MW0.i.2:X1	Range over/undershoot on channel i	Explicit

**Note:** For inputs integrated into the module, position x of the model is equal to 0 and the i channels are between 2 and 9.



## Sensor link monitoring

---

### At a Glance

This checking is available in the range 4-20 mA. In this range the fault is detected by the interface when the intensity in the loop becomes smaller than 3.2 mA.

---

### Fault bit

The sensor link fault is signaled by a status word bit for explicit exchange which can be used by the system.

Sensor link fault bit:

Bit name	Meaning (for $X_j = 1$ )
%MW0.i.2:X0	Sensor link fault on channel i

---



## Measurement filtering

### At a Glance

The system uses first order filtering.

The filtering coefficient can be modified from the PL7 screen (See Modification of the Filtering Value, p. 256) or via the program.

### Mathematical formula

The mathematical formula used is as follows :

$$\text{Mesf}(n) = \alpha \times \text{Mesf}(n - 1) + (1 - \alpha) \times \text{Valb}(n)$$

with:

$\alpha$  = filter efficiency,

Mesf(n)=measurement filtered at moment n,

Mesf(n-1)=measurement filtered at moment n-1,

Valg(n)=gross value at moment n.

At configuration, the user selects the filter value from 7 possible values (0 to 6). This value can be changed, even when the application is in RUN mode.

**Note:** Filtering is inhibited in fast cycle mode.

### Filter values

The filter values are as follows:

Required efficiency	Value to be selected	$\alpha$ corresponding	Filter response time	Cut-off frequency (Hz)
No filtering	0	0	0	-
Low filtering	1	0,750	111 ms	1,431
	2	0,875	240 ms	0,664
Medium filtering	3	0,937	496 ms	0,321
	4	0,969	1,01s	0,158
High filtering	5	0,984	2.03 s	0,078
	6	0,992	4.08 ms	0,039



## Displaying measurements

---

**At a Glance** All the measurements given to the application are in standardized display, and can be applied directly by the user.

---

**Standardized display** The values are displayed in standardized units (in % with 2 decimals, also with symbol ‰)

from 0 to 10000 (0 ‰ to 10000 ‰)

---



## 16.3 Output processing

### Output characteristics

**Writing output** The application must provide values in a standardized format to the output: 0 to +10000.  
The values must be written in the word %QW0.10.

**Overflow monitoring** If the value provided by the application is less than 0 or greater than 10000, the analog output saturates to 0 V or + 10 V.  
An overflow bit usable by the program is then positioned to 1.

Bit Name	Meaning
%I0.10.ERR	Channel 10 fault (analog output)

**Digital / analog conversion** Digital / analog conversion is carried out on 8 bits:  
The value provided by the application program (0 to 10000) is automatically transformed into a digital value which can be used by the converter.

**Refreshing the output** Output is refreshed at the end of the MAST task or the FAST task, depending on the selection made at configuration (See Modifying the task to which the output is assigned, p. 259).

**Fallback value** The output takes the fallback value:

- If dialog with the PLC is no longer possible
- If the PLC switches to STOP mode

These fallback values are selected at configuration (See Modification of the Fallback Mode, p. 258)  
Possible options are:

- Value 0
- Maintained at the last value sent







---

# Analog input modules TSX AEZ 801 / TSX AEZ 802

17

---

## Chapter overview

**What's in this Chapter?**

This chapter introduces the analog input modules TSX AEZ 801 and TSX AEZ 802 for the Micro PLC.

**What's in this Chapter?**

This Chapter contains the following Maps:

Topic	Page
Introduction to the TSX AEZ 801/TSX AEZ 802 modules	188
Timing of measurements	189
Range selection and input overflow monitoring	190
Sensor link checking on TSX AEZ 802	191
What happens to the module in the event of an overload	192
Measurement filtering	193
Displaying measurements	196



## Introduction to the TSX AEZ 801/TSX AEZ 802 modules

### General

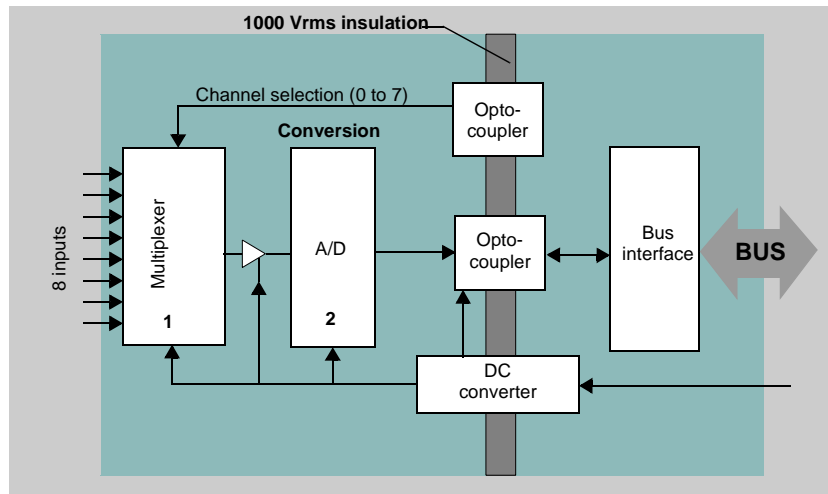
The TSX AEZ 801 and TSX AEZ 802 modules offer eight high level analog inputs, with common features.

For each of its inputs, the TSX AEZ 801 module offers the range 10 v or 0-10 v, depending on the selection made at configuration.

For each of its inputs, the TSX AEZ 802 module offers a range 0-20 mA or 4-20 mA, depending on the selection made at configuration (See Modification of the Input Range, p. 253).

### Diagram

Graphic representation of the module functions:



### Functions

The table below shows the module functions:

Numbers	Functions
1	Scanning input channels using static multiplexing and value acquisition
2	Digital/analog conversion (12 bits) of input measurements

The PLC processor completes the following functions:

Input under/overshoot monitoring
Measurement filtering
User-defined input measurement formatting for display in a unit which can be applied directly



## Timing of measurements

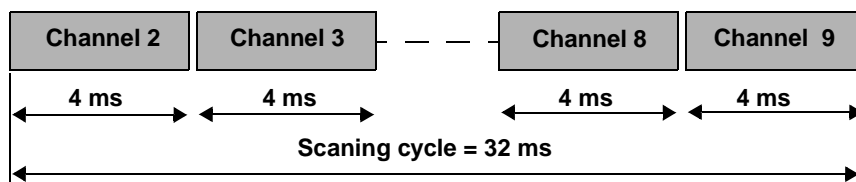
### General

The timing of measurements depends on the cycle used, defined at configuration (See Modification of the Scanning Cycle, p. 251): normal cycle or fast cycle.

### Normal cycle

The input scanning cycle is fixed and has a value of 32 ms, independent of the number of inputs used.

Example of a scanning cycle using only channels 0, 1, 6, and 7:



### Fast cycle

Only the channels used are scanned, even if they are not consecutive. This helps to improve the channel scanning cycle time.

The cycle time for scanning the channels is given by the formula:

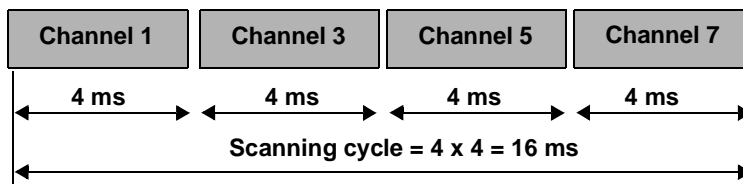
$$\text{Cycle time (ms)} = 4 \text{ ms} \times N$$

$N$  = number of channels used

For example, if 4 channels are used, the scanning cycle time will be:

$$4 \times 4 = 16 \text{ ms}$$

Illustration:



**Note:** In fast cycle, the user can assign the channels to FAST task (See Modification of the Task Assigned to the Module Inputs, p. 252). In this case, it is recommended that you do not assign too many analog input modules to the FAST task,



## Range selection and input overflow monitoring

**Range selection** Each module gives a choice (See Modification of the Input Range, p. 253) between two ranges for each of its inputs.

For the TSX AEZ 801 module:

- +/- 10 V
- 0-10 V

For the TSX AEZ 802 module:

- 0-20 mA
- 4-20 mA

For each of the analog inputs the interface carries out a range overshoot check, by checking that the measurement remains below the upper limit. If this is not the case, saturation of the measurement string is likely. An overshoot fault is signaled by a bit which can be used by the system.

Generally, the modules allow a range overshoot of 5% on the full scale.

### Overshoot values

The table below gives the limits for the TSX AEZ 801 module:

Range	Lower limit	Upper limit	Whole values available by default
+/- 10 V	- 10.5 V	+ 10.5 V	+/- 10500
0...10 V	- 0.5 V	+ 10.5 V	- 500...10500

The table below gives the limits for the TSX AEZ 802 module:

Range	Lower limit	Upper limit	Whole values available by default
0...20 mA	- 1 mA	+ 21 mA	- 500...10500
4...20 mA	+ 3.2 mA	+ 20.8 mA	- 500...10500

**Note:** In the case of unipolar ranges (0...10 V, 0...20 mA), the module detects an undershoot. A fault is signaled at - 5% of the scale, which allows a quicker diagnostic at implementation and in operation.

### Under/overshoot identifiers

In the over/undershoot zones, there is a risk of measurement string saturation. On under/overshoot, the acquisition of inputs continues, but they are signaled as invalid. Under/overshoots are signaled by the following bits (usable by the program):

Bit name	Meaning (when = 1)	Exchange type
%Ix.i.ERR	Fault on channel i of the module in position x	Implicit
%MWx.i.2:X1	Range under/overshoot on channel i of the module in position x	Explicit



---

## Sensor link checking on TSX AEZ 802

---

### At a Glance

This checking is available in the range 4-20 mA. In this range the fault is detected by the TSX AEZ 802 module when the intensity in the loop becomes less than 3.2 mA.

### Fault bit

The sensor link fault is signaled by a status word bit for explicit exchange which can be used by the system.

Sensor link fault bit:

Bit name	Meaning (for Xj = 1)
%MWx.i.2:X0	Sensor link fault on channel i of module x

**Note:** The channels of a TSX AEZ 802 module which have not been hardwired should preferably be set between 0-20 mA. If this is not the case, a "sensor link" fault will be signaled by the module.

---



**What happens to the module in the event of an overload**

---

**General** In the event of a surcharge – an under/overshoot on the upper (10500) or lower limit (-10500), the module signals a range under/overshoot fault.

---

**What happens to the module** What happens to the module in relation to the overload value:

Overload type	What happens to the module	Comments
Overload of less than 14 VDC (positive or negative)	The measurement string is saturated to the value of the passed limit (10500 or -10500).	The under/overshoot is not destructive for the module.
Overload between 14 VDC and 30 VDC (positive or negative)	The measurement provided by the module is non-significant.	The under/overshoot is not destructive for the module.
Overload greater than 30 VDC (positive or negative)	The range overshoot is signaled while the module can.	The under/overshoot can be irreversibly destructive for the module.

---



## Measurement filtering

### At a Glance

The system uses first order filtering.

The filtering coefficient can be modified (See Modification of the Filtering Value, p. 256) from the PL7 screen or via the program.

### Mathematical formula

The mathematical formula used is as follows :

$$\text{Mesf}(n) = \alpha \times \text{Mesf}(n - 1) + (1 - \alpha) \times \text{Valb}(n)$$

with:

$\alpha$  = filter efficiency,

Mesf(n)=measurement filtered at moment n,

Mesf(n-1)=measurement filtered at moment n-1,

Valg(n)=gross value at moment n.

At configuration, the user selects the filter value from seven possible values (0 to 6).

This value can be changed, even when the application is in RUN mode.

**Note:** Filtering is inhibited in fast cycle mode.

### Filter values

The filter values are as follows:

Required efficiency	Value to be selected	$\alpha$ corresponding	Filter response time	Cut-off frequency (Hz)
No filtering	0	0	0	-
Low filtering	1	0,750	111 ms	1,431
	2	0,875	240 ms	0,664
Medium filtering	3	0,937	496 ms	0,321
	4	0,969	1,01s	0,158
High filtering	5	0,984	2.03 s	0,078
	6	0,992	4.08 ms	0,039



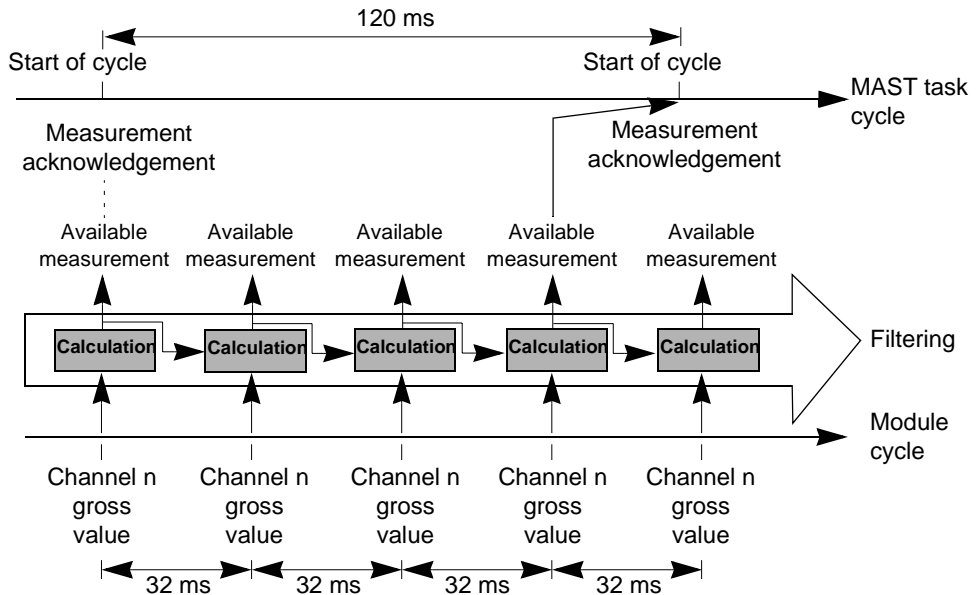
## Filtering and cycle time

The module continues its acquisitions and therefore its filtering calculations regardless of the cycle time of the application task.

For example:

If the cycle of the MAST task is 120 ms (module used in normal cycle), the module will have taken into account three or four new gross values per channel, before the MAST task reads the measurement value.

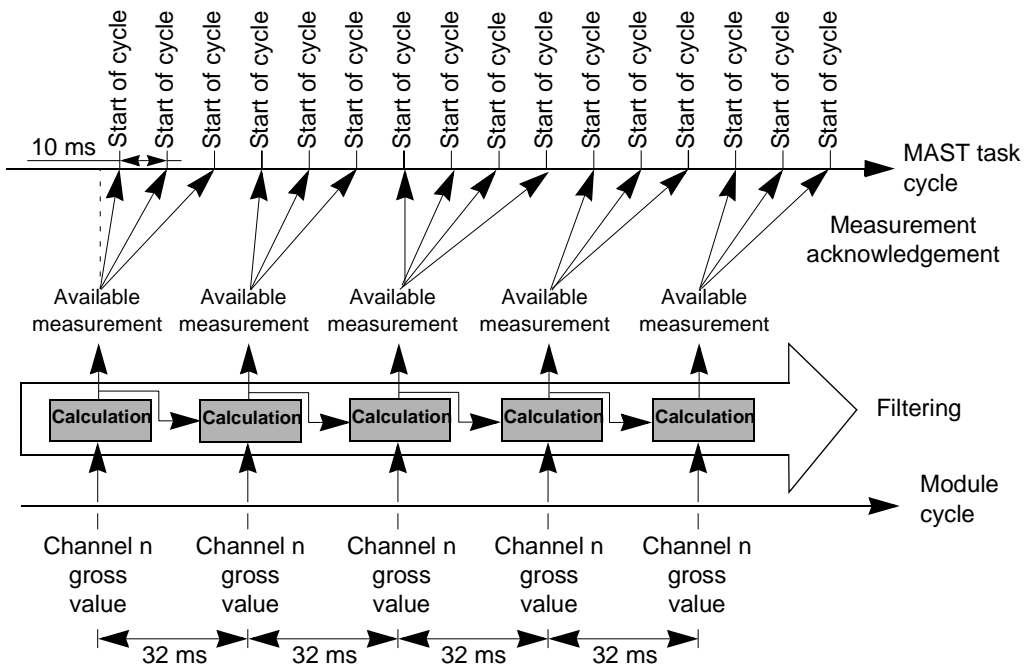
Illustration:



If the cycle of the MAST task is 10 ms, the module will only provide a new value every three or four cycles of the MAST task.



Illustration:





## Displaying measurements

### At a Glance

The measurement provided to the application can be applied directly by the user, who can choose (See Modification of the Display Format, p. 254) between:

- using standardized display 0-10000 (or +/- 10000 for the range +/-10 V)
- creating parameters for its display format by indicating the minimum and maximum values required.

### Standardized display

The values are displayed in standardized units (in % with 2 decimal places, also with symbol  $\text{‰}$ ).

Different standardized formats for each range:

Range	Reference	Display
<b>Unipolar:</b>		
0-10 V	TSX AEZ 801	from 0 to 10000 (0 $\text{‰}$ to 10000 $\text{‰}$ )
0-20 mA	TSX AEZ 802	from 0 to 10000 (0 $\text{‰}$ to 10000 $\text{‰}$ )
4-20 mA	TSX AEZ 802	from 0 to 10000 (0 $\text{‰}$ to 10000 $\text{‰}$ )
<b>Bipolar:</b>		
+/- 10 V	TSX AEZ 801	from -10000 to +10000 (-10000 $\text{‰}$ to +10000 $\text{‰}$ )

### User display

The user can select the value range in which the measurements are expressed, by selecting:

- the minimum limit corresponding to the range minimum 0  $\text{‰}$  (or -10000  $\text{‰}$ )
- the maximum limit corresponding to the range maximum 10000  $\text{‰}$

These minimum and maximum limits should be integers between -30000 and +30000.



---

# Analog input modules

## TSX AEZ 414

18

---

### Chapter overview

**What's in this Chapter?**

This chapter introduces the analog input modules TSX AEZ 414 for the Micro PLC.

**What's in this Chapter?**

This Chapter contains the following Maps:

Topic	Page
Introducing the TSX AEZ 414 module	198
Timing of measurements	200
Range selection and input overshoot monitoring	201
Sensor link monitoring	204
What happens to the module in the event of an overload	205
Measurement filtering	206
Displaying measurements	207



## Introducing the TSX AEZ 414 module

### General

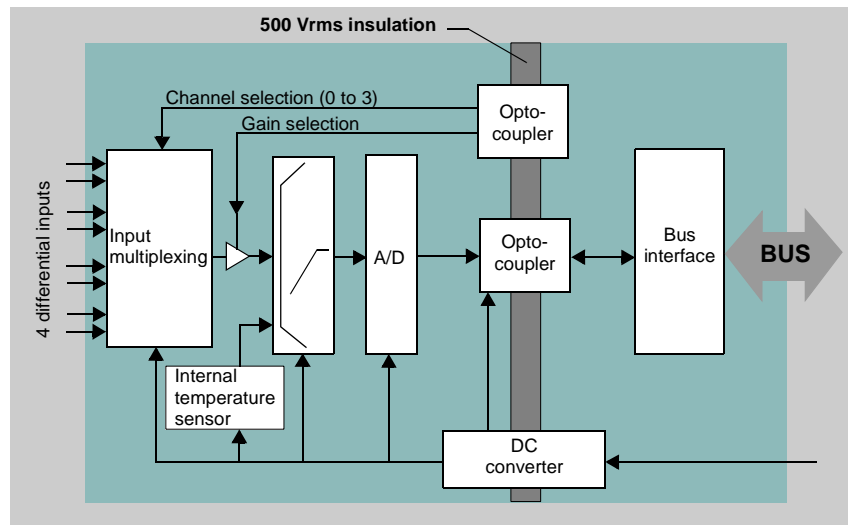
The TSX AEZ 414 module is a multi-range acquisition device with four differential inputs.

The TSX AEZ 414 module offers the following ranges for each of its inputs according to the selection made at configuration (See Modification of the Input Range, p. 253):

- Thermocouple B, E, J, K, L, N, R, S, T or U
- Thermowell Pt100 or Ni100 in two or four wires
- High level  $\pm 10$  V, 0-10 V, 0-5 V (0-20 mA with an external shunt). It should be noted that the external shunts are supplied with the product.

### Diagram

Graphic representation of the interface functions:





**Functions**

The table below details the interface functions:

Input range selection for each channel.
Scanning input channels using multiplexing and value acquisition.
Digital/analog conversion (16 bits) of input measurements.
Under/overshoot checking for input values according to the declared range.
Linearization in the case of thermowells Pt100 and Ni100.
Linearization and internal or external cold junction compensation for thermocouples.
User-defined input measurement formatting for display in units which can be applied directly (physical unit or user range).
Detection of a sensor link fault in thermocouple ranges.



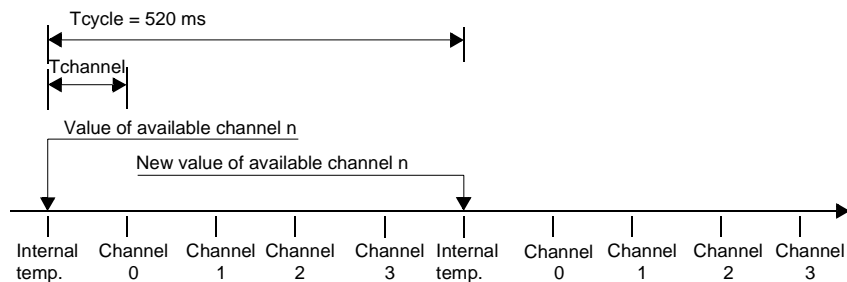
## Timing of measurements

### General

The cycle time of the TSX AEZ 414 module and consequently the sampling period are independent from the power frequency (50 Hz or 60 Hz). Internal module temperature (cold junction) acquisition is added to the full acquisition of the four channels

### Measurement sequence

The measurements are sequenced in the following manner:



The scanning cycle is always identical even if some channels or the internal temperature are not used. It is equal to 520 ms.

### Scanning time

The table below shows the module scanning times:

Time	Abbreviation	Value
Acquisition time for a channel	Channel T	104 ms
Scanning cycle time	T cycle	520 ms

**Note:** Module initialization can take up to 1.5 s. During this time the channels are signaled "not ready" by bit 8 of the channel status word: %MWx.i.2:X8



## Range selection and input overshoot monitoring

---

**Range selection** Using the software the user can select (See Modification of the Input Range, p. 253), for each of the channels, one of the following ranges:

- $\pm 10$  V
- 0-10 V
- 0-5 V (or 0-20 mA with external shunt)
- 1-5 V (or 4-20 mA with external shunt)
- Thermowells Pt100, Ni1000
- Thermocouples B, E, J, K, L, N, R, S, T and U

For thermocouple ranges, the module provides cold junction compensation. However, the cold junction temperature can be measured either on the module terminal block (by a probe inside the module), or remotely by using an external Class A Pt100 probe (not supplied) on channel 0. This parameter is selected at configuration (See Zone 3, p. 247).

---

### Range under/overshoot

Depending on the range the under/overshoot values are as follows:

- For the "bipolar voltage" range  $\pm 10$  V, the range under/overshoot corresponds to a value outside the range  $\pm 105\%$  of the full scale.
- For the "unipolar voltage" range, the range under/overshoot corresponds to a value outside the range - 5% and + 105% of the full scale.
- For temperature measurements using thermocouples, the range under/overshoot corresponds to:
  - either a standardized sensor zone under/overshoot.
  - or under/overshoot of the acquisition device dynamic.
  - or an under/overshoot of the compensation temperature dynamic (- 5 °C to + 85 °C).

Using the internal compensation at a normative ambience (0 °C to + 60 °C) is compatible with the thresholds - 5 °C and 85 °C.

- For temperature measurements using thermowells, the range under/overshoot corresponds to:
    - Either an under/overshoot of the acquisition device dynamic (due to a sensor or cabling anomaly).
    - Or a standardized sensor zone under/overshoot.
-



**Under/overshoot monitoring**

Whatever the chosen range, range under/overshoots are monitored. The module checks that the measurement falls between a lower and upper limit. Outside these limits, saturation of the measurement string is likely.

The table below gives the under/overshoot limits for the electrical ranges:

Range	Lower limit	Upper limit
+/- 10 V	- 10.5 V	+ 10.5 V
0...10 V	- 0.5 V	+ 10.5 V
0...5 V (0...20 mA)	- 0.25 V (-1 mA)	+ 5.25 V (+21 mA)
1...5 V (4...20 mA)	+ 0.8 V (+ 3.2 mA)	+ 5.2 V (+ 20.8 mA)

The table below gives the under/overshoot limits for the thermocouple ranges:

Range	Lower limit	Upper limit
B	0 °C (32 °F)	+ 1802 °C (+ 3276 °F)
E	- 270 °C (- 454 °F)	+ 812 °C (+ 1493 °F)
J	- 210 °C (- 436 °F)	+ 1065 °C (+ 1949 °F)
K	- 270 °C (- 454 °F)	+ 1372 °C (+ 2502 °F)
L	- 200 °C (- 328 °F)	+ 900 °C (+ 1652 °F)
N	- 270 °C (- 454 °F)	+ 1300 °C (+ 2372 °F)
R	- 50 °C (- 58 °F)	+ 1769 °C (+ 3216 °F)
S	- 50 °C (- 58 °F)	+ 1769 °C (+ 3216 °F)
T	- 270 °C (- 454 °F)	+ 400 °C (+ 752 °F)
U	- 200 °C (- 328 °F)	+ 600 °C (+ 1112 °F)
<b>The limits are given for the following conditions:</b> <b>In internal compensation the ambient temperature is 25 °C,</b> <b>In external compensation, the cold junction temperature is 30 °C.</b>		

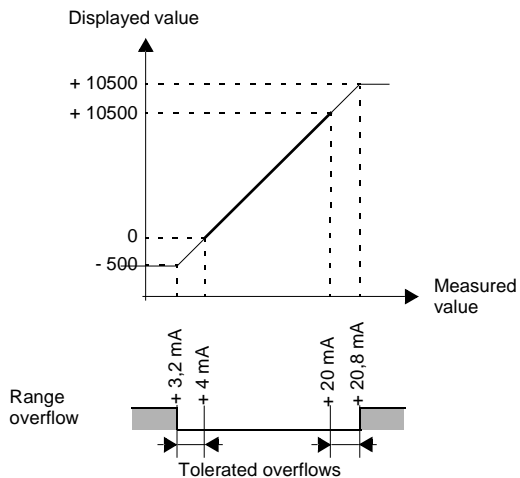
The table below gives the under/overshoot limits for the thermowell ranges:

Range	Lower limit	Upper limit
Pt100	- 200 °C (- 328 °F)	+ 850 °C (+ 1562 °F)
Ni1000	- 60 °C (- 76 °F)	+ 110 °C (+ 230 °F)



**Note:** In the event of a range under/overshoot, the given value saturates to the value of the limit passed:

#### Example for range 4-20 mA



#### Under/overshoot identifiers

The program can use range/under overshoot error bits. Limit under/overshoots are signaled by the following bits:

Bit name	Meaning (when = 1)	Exchange type
%lx.i.ERR	Fault on channel i of the module in position x	Implicit
%MWx.i.2:X1	Range under/overshoot on channel i of the module in position x	Explicit

**Note:** For thermocouple ranges, the %lx.i.ERR bit is also set to 1 for a sensor link anomaly.



## Sensor link monitoring

---

### At a Glance

Sensor link monitoring is only done for measurements by thermocouples. However, a range under/overshoot in the range 4-20 mA (< 3.2 mA) does not cause a sensor link fault.

The sensor link fault corresponds to an open circuit on the thermocouple input. Acknowledgement is not completely synchronous with its appearance; it can be delayed by a maximum of three "module cycles", or 1560 ms. This is the same for the disappearance of the fault.

---

### Fault bit

The sensor link fault is signaled by a status word bit for explicit exchange which can be used by the system.

Sensor link fault bit:

Bit name	Meaning (for Xj = 1)
%MWx.i.2:X0	Sensor link fault on channel i of module x

---



---

## What happens to the module in the event of an overload

---

### General

In the event of a surcharge, that is an under/overshoot on the upper or lower limit, the module signals a range under/overshoot fault.

---

### What happens to the module

What happens to the module in relation to the overload value:

Overload type	What happens to the module	Comments
Overload of less than 15 VDC (positive or negative)	The channel to channel crosstalk is not changed	The under/overshoot is not destructive for the module.
Overload between 15 VDC and 30 VDC (positive or negative)	The channel to channel crosstalk makes all the module inputs unusable.	The under/overshoot is not destructive for the module.
Overload greater than 30 VDC (positive or negative)	The range overshoot is signaled while the module can.	The under/overshoot can be irreversibly destructive for the module.

**Note:** A sensor link fault with a 2 wire thermowell can lead to saturation of the input concerned, at a voltage between 15 VDC and 30 VDC and thus make the module inputs unusable.

---



---

## Measurement filtering

---

### At a Glance

The system uses first order filtering.

The filtering coefficient can be modified (See Modification of the Filtering Value, p. 256) from the PL7 screen or via the program.

At configuration, the user selects the filter value from seven possible values (0 to 6).

---

### Filter values

The filter values are as follows:

Required efficiency	Value to be selected	$\alpha$ corresponding	Filter response time	Cut-off frequency (Hz)
No filtering	0	0	0	Hardware filtering
Low filtering	1	0,750	1.81 s	0,0879
	2	0,875	3.89 s	0,0409
Medium filtering	3	0,937	8.06 s	0,0197
	4	0,969	16.4 s	0,0097
High filtering	5	0,984	33 s	0,0048
	6	0,992	66.3 s	0,0024

**Note:** Measurement filtering is suspended when execution of the MAST task is interrupted at a breakpoint (in the debugging phase). When the breakpoint is cleared, filtering restarts without recognizing the inputs acquired during the stopped period.

---



## Displaying measurements

### At a Glance

This process is used to select (See Modification of the Display Format, p. 254) the display format depending on which formats are provided in the user program:

- Standardized display
- User display

It is necessary to differentiate between the electrical ranges and the thermocouple or thermowell ranges.

### Electrical ranges: Standardized display

The values are displayed in standardized units (in % with 2 decimal places, also with symbol ‰).

Different standardized formats for each range:

Range	Display
<b>Unipolar:</b>	
0-10 V 0-5 V (0-20 mA) 1-5 V (4-20 mA)	from 0 to 10000 (0 ‰ to 10000 ‰)
<b>Bipolar:</b>	
+/- 10 V	from -10000 to +10000 (-10000 ‰ to +10000 ‰)

### Electrical ranges: User display

The user can select the value range in which the measurements are expressed, by selecting:

- the minimum limit corresponding to the range minimum 0 ‰ (or -10000 ‰)
- the maximum limit corresponding to the range maximum 10000 ‰

These minimum and maximum limits should be integers between -30000 and +30000

### Temperature range

The user can choose between two types of display:

- **Temperature display.** The values are provided by default in tenths of a degree:
  - tenth of a degree Celsius, if the unit chosen at configuration is °C
  - tenth of a degree Fahrenheit, if the unit chosen at configuration is °F
- **Standardized display.** The user can select standardized display 0-10000 (i.e. 0 to 10000 ‰), by specifying the minimum and maximum temperatures corresponding to 0 and 10000.







---

# Analog output module

## TSX ASZ 401

19

---

### Chapter overview

**What's in this Chapter?**

This chapter introduces the analog output module TSX ASZ 401 for Micro PLCs.

**What's in this Chapter?**

This Chapter contains the following Maps:

Topic	Page
Introducing the TSX ASZ 401module	210
Writing and updating outputs	211
Fault handling	212



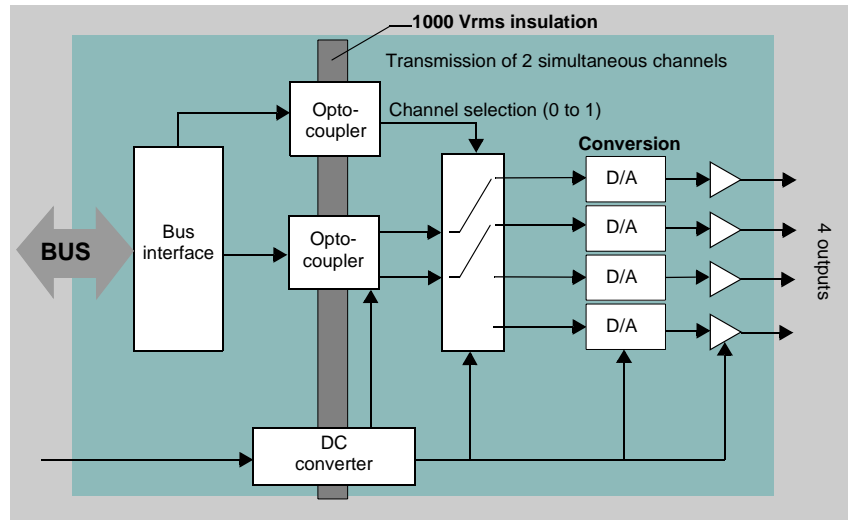
## Introducing the TSX ASZ 401 module

### General

The TSX ASZ 401 module offers four analog outputs with common features and for each of them offers the range  $\pm 10$  V, with no energy supply (no external supply) with a load of at least 2 K $\Omega$ .

### Diagram

Graphical representation of the interface functions:



### Functions

The table below shows the functions of the modules:

Acknowledgment of the digital values corresponding to the analog values to be obtained on exit. These values are calculated by the PLC task to which the channels are assigned.
Handling of processor dialog faults and particularly setting the output to fallback.
Digital/analog output value conversion on 11 bits + sign (-2048 to +2047). Realignment is carried out in the dynamic of the converter.



## Writing and updating outputs

### Writing outputs

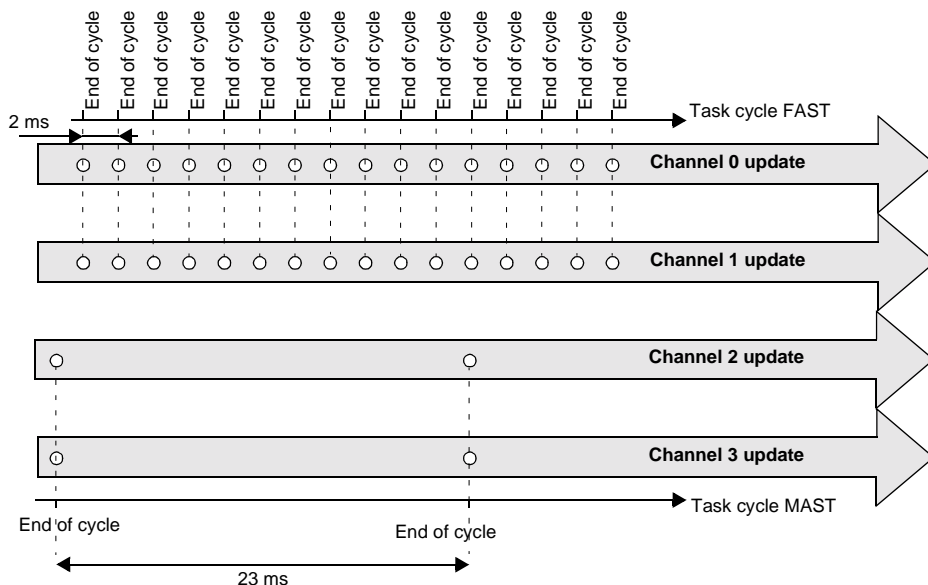
The application must provide values to the outputs in a standardized format - 10000 to +10000.

### Updating outputs

The outputs of the TSX ASZ 401 module are updated two at a time at the end of the task to which they are assigned (See Modifying the task to which the output is assigned, p. 259).

#### Example:

Suppose that channels 0 and 1 are assigned to the FAST task which has a cycle time of 2 ms and channels 2 and 3 are assigned to the MAST task which has a cycle time of 23 ms. Channel updating will be as follows:



**Note:** As the channels are grouped 0/1 and 2/3, it is not possible to assign channels 0 and 2 to a task (for example MAST) and 1, 3 to another task (for example FAST).



## Fault handling

---

### Under/overshoot monitoring

If the values given by the application are less than -10000 or greater than +10000, the outputs saturate to - 10 V or + 10 V.

An overshoot bit which can be used by the program is then set to 1.

Bit address	Meaning (for Xj = 1)
%Ix.i.ERR	Fault on channel i of module x
%MWx.i.2:X1	Range overshoot fault

### Output fallback

When the PLC switches into STOP mode or dialog with the processor is no longer possible, the outputs assume the fallback value 0 or are maintained at the last value transmitted, according to the choice made at configuration (See Modification of the Fallback Mode, p. 258) for the module.

---



---

# Analog output module

## TSX ASZ 200

20

---

### Chapter overview

**What's in this Chapter?**

This chapter introduces the analog output module TSX ASZ 200 for Micro PLCs.

**What's in this Chapter?**

This Chapter contains the following Maps:

Topic	Page
Introducing the TSX ASZ 200 module	214
Writing and updating outputs	215
Fault handling	216
Fault Processing	217

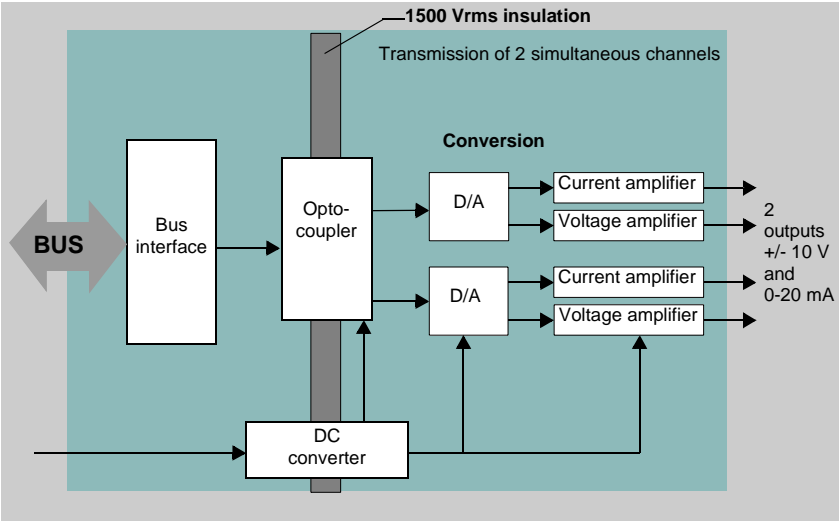


Introducing the TSX ASZ 200 module

- General
- The TSX ASZ 200 module offers two analog outputs with common features and offers for each the following ranges (See Modification of the Output Range (TSX ASZ 200 and TSX AMZ 600), p. 260), with no energy supply (no external supply):
- -/+10 V with a minimum load of 1 K $\Omega$
  - 0-20 mA with a maximum load of 600  $\Omega$
  - 4-20 mA with a maximum load of 600  $\Omega$

Diagram

Graphical representation of the interface functions:



Functions

The table below shows the functions of the modules:

Acknowledgment of the digital values corresponding to the analog values to be obtained on output. These values are calculated by the PLC task to which the channels are assigned.
Handling of processor dialog faults and particularly setting the output to fallback.
Range selection for each output: voltage or current.
Digital/analog conversion of output values: on 11 bits + sign (-2048 to +2047) in the range +/- 10 V, on 11 bits (0 to +2047) in the ranges 0-20 mA and 4-20 mA. Realignment is carried out in the dynamic of the converter by the module.



## Writing and updating outputs

---

### Writing outputs

The application must provide values in a standardized format to the outputs:

- -10000 to +10000 in the range  $\pm 10$  V.
- 0 to + 10000 in the ranges 0-20 mA and 4-20 mA. The value 0 corresponding to 4 mA in the range 4-20 mA.

### Updating outputs

The two outputs of the TSX ASZ 200 module are updated at the end of the task to which they are assigned (See Modifying the task to which the output is assigned, p. 259).

---



## Fault handling

---

### Under/overshoot monitoring

The output saturation values according to the range are as follows:

Ranges	Values provided by the application	Saturation value
+/- 10 V	Less than -10000	- 10 V
	More than + 10000	+ 10 V
0-20 mA	Less than 0	0 mA
	More than + 10000	20 mA
4-20 mA	Less than 0	4 mA
	More than + 10000	20 mA

An overshoot bit which can be used by the program is then set to 1:

Bit address	Meaning
%Ix.i.ERR	Fault on channel i of module x
%MWx.i.2:X1	Range under/overshoot fault

### Output fallback

When the PLC switches into STOP mode, the outputs take the fallback value 0 (4mA in the range 4-20 mA) or are maintained at the last transmitted value, according to the selection made during configuration (See Modification of the Fallback Mode, p. 258) of the module.

When dialog with the processor is no longer possible, the outputs take the fallback value 0 V (voltage range) or 0 mA (current range).

---



## Fault Processing

### Overshoot Monitoring

The saturation values of the outputs according to the range are the following:

Ranges	Values provided by the application	Saturation value
+/- 10 V	Less than - 10000	- 10 V
	More than + 10000	+ 10 V
+/- 10V	Less than 0	0 V
	More than + 10000	+10 V
0-20 mA	Less than 0	0 mA
	More than + 10000	20 mA
4-20 mA	Less than 0	4 mA
	More than + 10000	20 mA

An overshoot bit, which can be run by the program, is then positioned at 1:

Bit address	Meaning
%Ix.i.ERR	Fault in channel i of the module x
%MWx.i.2:X1	Range overshoot fault

### Output Fallback

When the PLC changes to STOP, the outputs take the fallback value 0 (4 mA in the range 4-20 mA) or are kept at the last transmitted value, according to the choice made in configuration (See Modification of the Fallback Mode, p. 258) for the module.

When dialog with the processor is no longer possible, the outputs take the fallback value 0 V (voltage range) or 0 mA (current range).

### Write Output

The application must provide outputs with values in standardized format.

- -10000 to +10000 in the range +/- 10V,
- 0 to +10000 in the ranges 0-10V, 0-20mA and 4-20mA.

### Refresh Outputs

The two outputs of the TSX AMZ 600 module are updated at the end of the task (\*\*\*\*\*see modification of task p255 \*\*\*\*\* ) to which they are assigned.







---

## Introduction to the Chapter

### What's in this Chapter?

This chapter introduces the TSX AMZ 600 analog input/output module for Micro PLCs.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Introduction to the TSX AMZ 600 Module	220
Measuring Speed	222
Selection of Ranges and Monitoring of Input Overshoots	223
Monitoring of Sensor Link on TSX AMZ 600	224
Module Behavior in Event of Overload	225
Filtering of Measurements	226
Display of Current Values	229



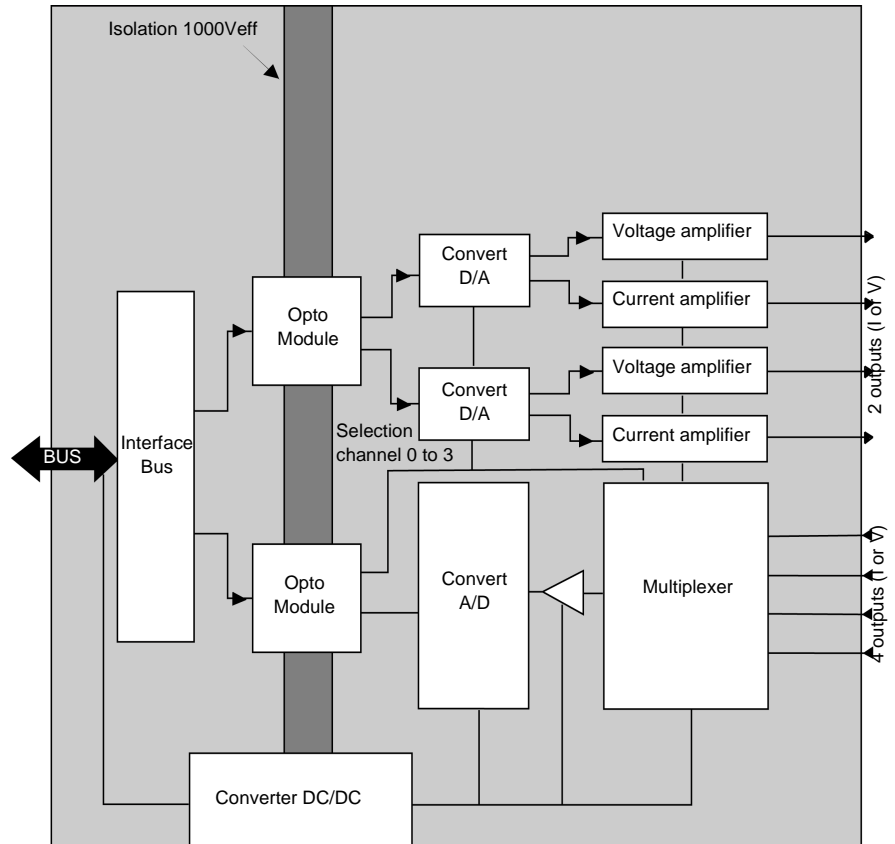
## Introduction to the TSX AMZ 600 Module

### General

The AMZ 600 module offers 4 analog inputs and 2 analog outputs of a high level. This module does not have isolation between input and output channels. For each of its input and output channels the TSX AMZ 600 offers 2 voltage ranges ( $\pm 10V$ ,  $0-10V$ ) and 2 current ranges ( $0-20mA$ ,  $4-20mA$ ) according to the selection made in configuration (See Modification of the Input Range, p. 253).

### Synopsis

The diagram below represents the internal structure of the module:





**Functions**

The paragraph below gives the input and output functions of the module:

- Input functions:
    - scanning of the input channels by static multiplexing and acquisition of the values,
    - analog/digital conversion (11 signed bits) of the input readings
    - monitoring input overshoot,
    - current value filtering,
    - user formatting the current input values for a directly operating individual display.
  - Output functions:
    - accepting the digital values which correspond to the analog values to be obtained at outputs. These values are calculated by the PLC task to which the channels are assigned,
    - the processing of dialog faults with the PLC and in particular setting output fallback,
    - selecting the range for each voltage and current output,
    - numeric/analog conversion of the output values.
-



## Measuring Speed

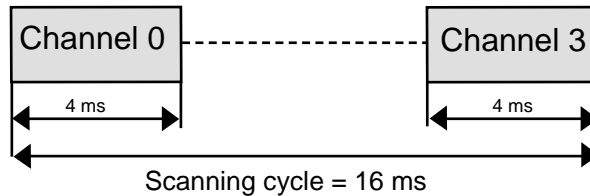
### General

The measuring speed depends on the cycle used, which is defined in configuration (See Modification of the Scanning Cycle, p. 251): normal cycle or fast cycle.

### Normal Cycle

The input scanning cycle is fixed and has a value of 16 ms, irrespective of the number of inputs used.

Example of a scanning cycle with channels 0 and 3 used:



### Fast Cycle

Only the channels being used are scanned, even if these are not consecutive, which gives an improved channel scanning cycle time.

The channel scanning cycle time is given by the formula:

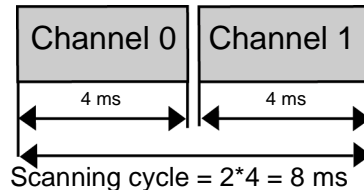
$$\text{Cycle time (ms)} = 4 \text{ ms} \times N$$

$N$  = number of channels used

For example, if 2 channels are used, the scanning cycle time will be:

$$2 \times 4 = 8 \text{ ms}$$

Illustration:



**Note:** In fast cycle, the user is able to assign the channels in FAST task (See Modification of the Task Assigned to the Module Inputs, p. 252). In this case, it is recommended not to assign too many analog input modules to the FAST task.



## Selection of Ranges and Monitoring of Input Overshoots

### Selection of Ranges

Each module offers the choice (See Modification of the Input Range, p. 253) between two ranges for each of its inputs.

For the TSX AMZ 600 module:

- +/- 10 V
- 0-10 V
- 0-20 mA
- 4-20 mA

For each of the analog inputs, the interface carries out monitoring of range overrun, by checking that the current value remains less than the upper limit. If this is not the case, there is the probability of the analog measurement system being saturated. An overshoot fault is shown by a bit that can be used by the system.

In general, the modules allow a range overshoot of 5% on the full scale.

### Overshoot Values

The table below gives the limits for the TSX AMZ 600 module:

Range	Lower limit	Upper limit	Whole values available by default
+/- 10 V	- 10.5 V	+ 10.5 V	+/- 10500
0...10 V	- 0.5 V	+ 10.5 V	- 500...10500
0...20 mA	- 1 mA	+ 21 mA	- 500...10500
4...20 mA	+ 3.2 mA	+ 20.8 mA	- 500...10500

**Note:** In the case of unipolar ranges (0-10 V, 0-20 mA), the module detects a negative overshoot. A fault is indicated at - 5% from the scale, which allows faster diagnostics during set-up and operation.

### Overshoot Indications

In the overshoot zones there is a risk of saturation of the analog measurement system.

On overshoot, the acquisition of inputs continues, but these are denoted as not valid. Overshoot of the limits is indicated by the following bits (can be run by program):

Bit name	Meaning (when = 1)	Type of exchange
%Ix.i.ERR	Fault in channel i of the module in position x	Implicit
%MWx.i.2:X1	Range overshoot on channel i of the module in position x	Explicit



## Monitoring of Sensor Link on TSX AMZ 600

---

**At a Glance** Monitoring is offered in the range 4-20 mA. In this range a fault is detected by the TSX AMZ 600 module when the intensity in the loop falls below 3.2 mA.

---

**Default Bit** The sensor link fault is indicated by an explicit exchange status word bit, which can be run by the system.

Sensor link fault bit:

Bit name	Meaning (for Xj = 1)
%MWx.i.2:X0	Sensor link fault on channel i of the module x

**Note:** Non cabled channels of a TSX AMZ 600 module should preferably be parametrized in 0-20 mA. If this is not the case, a "sensor link" fault will be indicated by the module.

---



---

## Module Behavior in Event of Overload

---

### General

During an overload, i.e. an overshoot of the upper limit (10500) or lower limit (-10500), the module indicates a range overshoot fault.

---

**Module Behavior** Behavior of module according to the overload value:

Type of overload	Module Behavior	Comments
Overload less than 14 VDC (positive or negative)	The analog measurement system is saturated at the value of the limit overshoot (10500 or -10500).	Overshoot is not detrimental to the module.
Overload within 14 VDC and 30 VDC (positive or negative)	The current value provided by the module is not significant.	Overshoot is not detrimental to the module.
Overload more than 30 VDC (positive or negative)	The range overshoot fault is indicated as long as this is possible for the module.	The overshoot can be permanently detrimental to the module.
Overload of +/- 7.5 VDC (current overload)	The analog measurement system is saturated at the value of the limit overshoot (10500 or -500)	Overshoot is not detrimental to the module.

---



## Filtering of Measurements

### Introduction

The filtering done is a first order filtering.  
The filtering coefficient can be modified (See Modification of the Filtering Value, p. 256) from the PL7 screen or by program.

### Mathematical Formula

The mathematical formula used is the following:

$$\text{Mesf}(n) = \alpha \times \text{Mesf}(n-1) + (1 - \alpha) \times \text{Valb}(n)$$

with:

$\alpha$  =efficiency of the filter,

Mesf(n)=measurement filtered at point n,

Mesf(n-1)=measurement filtered at the point n-1,

Valb(n)=gross value at point n.

During configuration, the user selects the filtering value from 7 possibilities (0 to 6).

This value can be modified, even when the application is in RUN.

**Note:** Filtering is inhibited in fast cycle.

### Filtering Values

The filtering values are the following:

Efficiency found	Value to be selected	$\alpha$ corresponding	Filter response time	Outage frequency (Hz)
No filtering	0	0	0	-
Little filtering	1	0,750	111 ms	1,431
	2	0,875	240 ms	0,664
Medium filtering	3	0,937	496 ms	0,321
	4	0,969	1.01s	0,158
Strong filtering	5	0,984	2.03 s	0,078
	6	0,992	4.08 s	0,039



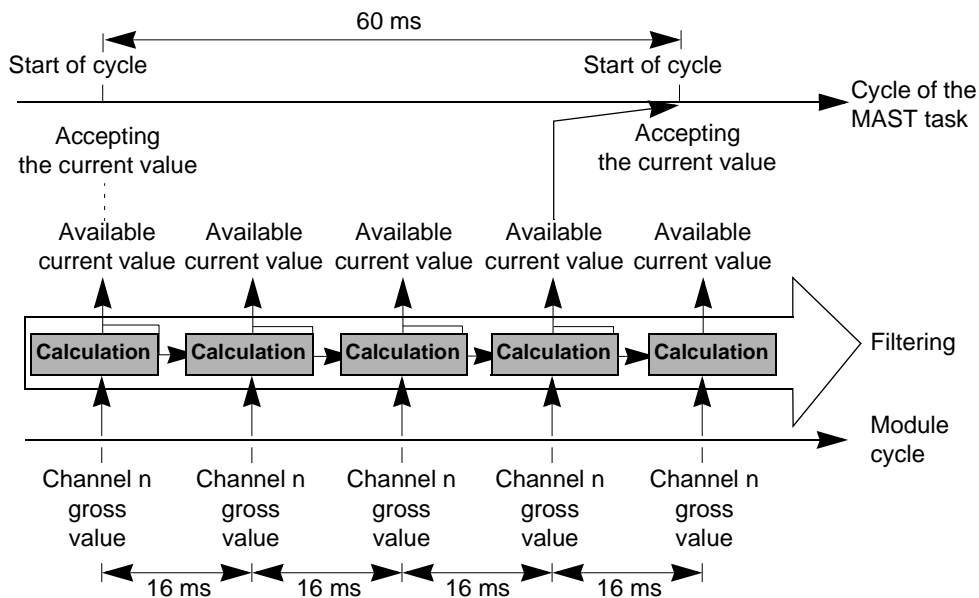
## Filtering and Cycle Time

The module continues its acquisitions and thus its filtering calculation with no regard to the cycle time of the application task.

For example:

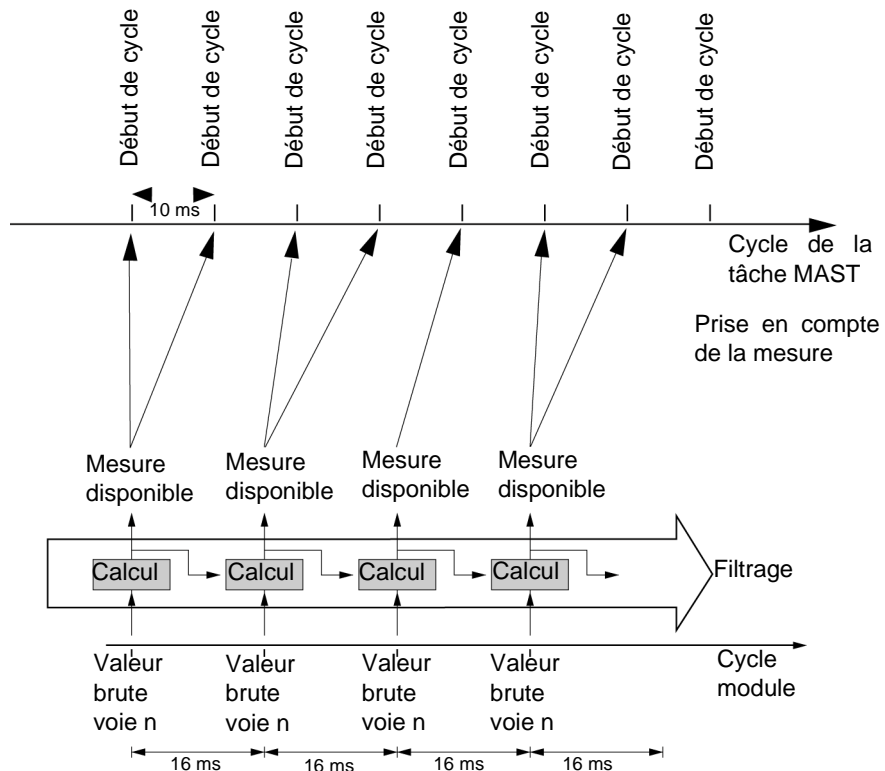
If the MAST task cycle is 60 ms (module used in normal cycle), the module will have accepted 3 or 4 new raw values per channel, before the MAST task comes to read the measurement value.

Illustration:





If the MAST task cycle is 10 ms, the module will only provide a new value every 2 or 3 cycles of the MAST task.





## Display of Current Values

### Introduction

The current value supplied to the application can be directly run by the user who can choose (See Modification of the Display Format, p. 254) between:

- using the standardized display 0-10000 (or +/- 10000 for the range +/- 10V),
- parametrizing the display format by giving the minimum and maximum values required.

### Standardized Display

The values are displayed in standardized units (in % with 2 decimals, also symbolized ‰).

Different formats standardized according to the ranges:

Range	Display
<b>Unipolar:</b>	
0-10 V	from 0 to 10000 (0 ‰ to 10000 ‰)
0-20 mA	from 0 to 10000 (0 ‰ to 10000 ‰)
4-20 mA	from 0 to 10000 (0 ‰ to 10000 ‰)
<b>Bipolar:</b>	
+/- 10 V	from -10000 to +10000 (-10.000 ‰ to +10.000 ‰)

### User Display

The user can select the range of values in which the current values are expressed, by choosing:

- the lower limit corresponding to the minimum value of the range 0 ‰ (or -10000 ‰)
- the upper limit corresponding to the maximum value of the range 10.000 ‰

These lower and upper limits are integers between - 30000 and + 30000.







---

# Configuring the analog application

# 22

---

## At a Glance

### What's in this Chapter?

This chapter describes the configuration of the analog modules on the PLC Micro

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
22.1	Reminder about the configuration editor	233
22.2	Accessing the analog application parameters	237







---

## 22.1      Reminder about the configuration editor

---

### At a Glance

---

#### What's in this Section?

This section contains a reminder about the use of the configuration editor for the analog application.

---

#### What's in this Section?

This Section contains the following Maps:

Topic	Page
Accessing the configuration editor	234
Selection of Modules	235


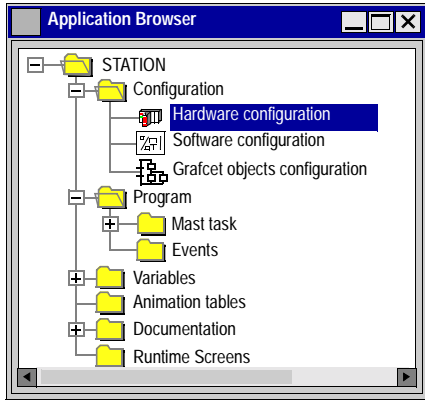
---



## Accessing the configuration editor

### Procedure

To access the configuration editor:

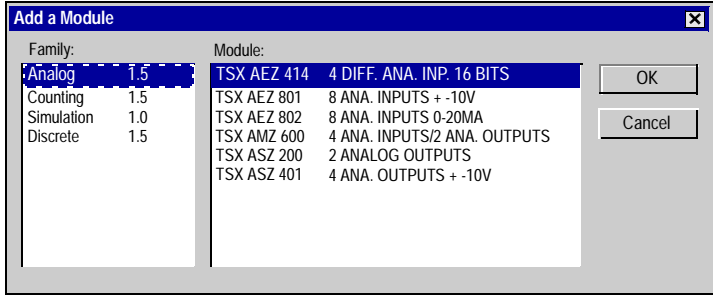
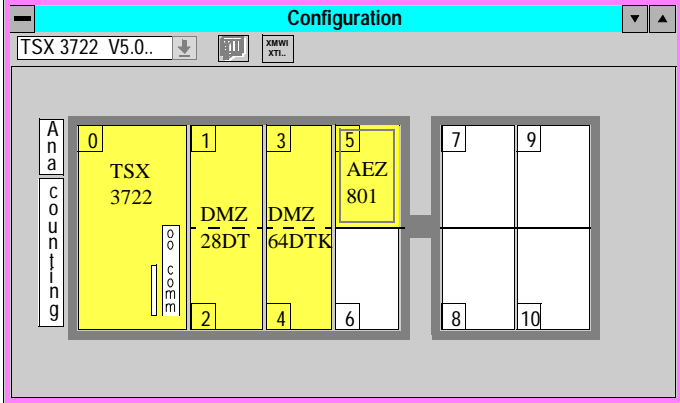
1	<p>Go to the application browser:</p> <ul style="list-style-type: none"><li>• Either by clicking on the application browser icon </li><li>• Or by selecting the command <b>Tools/Browser Application</b></li></ul>
2	Select and click on the <b>Station</b> folder.
3	Select and click on the <b>Configuration</b> folder.
4	<p>Double click on the <b>Hardware configuration</b> icon.</p> <p>Illustration:</p> 



## Selection of Modules

### Procedure

To select a module:

1	Double click on the position of the rack to be configured (for example 5).
2	<p>The following dialog box appears:</p> 
3	In the <b>Family</b> field, select the type of module (for example Analog).
4	In the <b>Module</b> field, select the reference of the module to be configured (for example TSX AEZ 801).
5	<p>Confirm with <b>OK</b>: the module is now declared in its position (which is framed and contains the reference of the module).</p> <p>Illustration:</p>  <p>Note: To delete a module from its position, select by clicking on it then press the &lt;Delete&gt; key, which brings up a dialog box. Then confirm the deletion of the module.</p>



**Limitations**

The limitations on the number of modules are as follows:

PLC type	Max. number of analog modules
TSX 37-10 + Extension	2
TSX 37-21/22 + Extension	4
TSX 37 base	2 modules TSX ASZ 200 and AMZ 600

---



---

## 22.2 Accessing the analog application parameters

---

### At a Glance

#### What's in this Section?

This section describes the methods for accessing the analog application parameters.

#### What's in this Section?

This Section contains the following Maps:

Topic	Page
Accessing Parameterization of the Integrated Analog Interface	238
Accessing Parameterization of an Analog Module	239

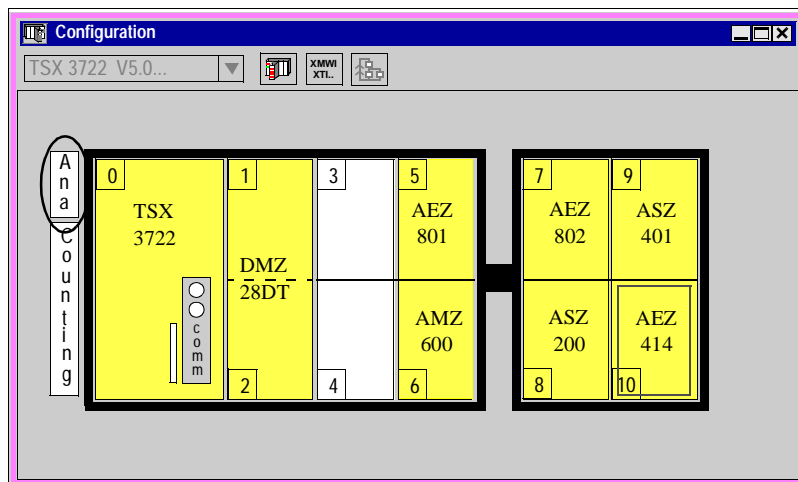


## Accessing Parameterization of the Integrated Analog Interface

---

### Procedure

Access by double clicking on the representation of the analog interface:



**Note:** Only accessible if the PLC is a TSX 37-22

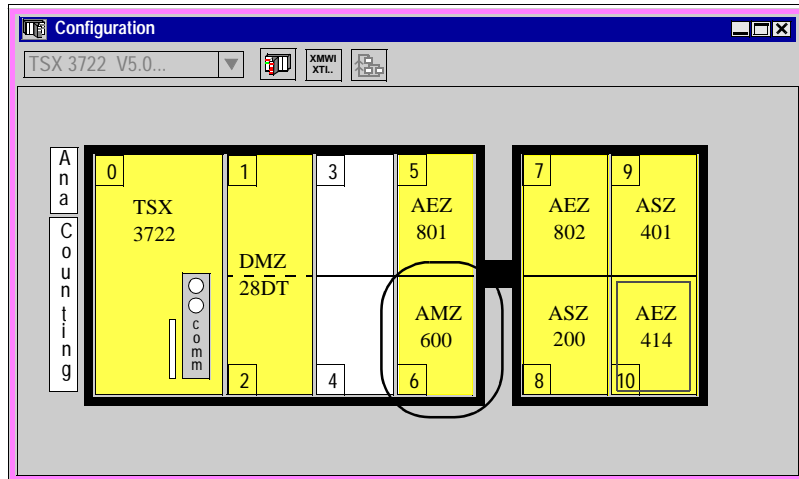
---



## Accessing Parameterization of an Analog Module

### Procedure

access by double clicking on the representation of the module to be configured (for example module TSX AMZ 600, positioned in slot 6):



Parameterization can also be accessed via the **Open module** command on the **Edit** drop-down menu







---

# Configuration of analog channels

23

---

## At a Glance

### What's in this Chapter?

This chapter introduces the configuration of the analog module channels on the Micro PLC

### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
23.1	Channel configuration function - General	243
23.2	Modifying an input channel's parameters	250
23.3	Modifying an output channel's parameters	257







# 23.1

# Channel configuration function - General

## At a Glance

### General

This function, which can be accessed from each module, including the interface integrated into the base module TSX 37-22, is used to display and modify the parameters for each of the module channels (electrical range, measurement filtering, measurement display, etc.)

### What's in this Section?

This Section contains the following Maps:

Topic	Page
Default Configuration of Channels	244
Display of Channel Parameters	246



## Default Configuration of Channels

### General

If the channel parameters are not modified, these are configured with a default value.

### Parameters

The default parameters of each of the analog modules are as follows (shown in bold in the table):

Module	Number of channels	Display format of the current value	Display	Task assignment	Filtering
Interface integrated	8 inputs (3)	<b>0..10 V</b> 0..20 mA 4..20 mA	<b>0 to 10000</b> (1)	<b>MAST</b> FAST (2)	<b>0..6</b>
	1 output	<b>0..10 V</b> (1)	<b>0 to 10000</b> (1)	<b>MAST</b> FAST	-
TSX AEZ 801	8 inputs (3)	<b>+/- 10 V</b> 0..10 V	<b>-10000 to +10000</b> (%..) User	<b>MAST</b> FAST (2)	<b>0..6</b>
TSX AEZ 802	8 inputs (3)	<b>0..20 mA</b> 4..20 mA	<b>0 to 10000</b> (%..) User	<b>MAST</b> FAST (2)	<b>0..6</b>
TSX AEZ 414	4 inputs	<b>+/- 10 V</b> 0..10 V 0.5 V ( 0..20 mA) 1.5 V ( 4..20 mA)	<b>-10000 to +10000</b> (%..) User	<b>MAST</b> FAST (2)	<b>0..6</b>
		Pt100, Ni1000 Thermocouple type B, E, J, K, L, N, R, S, T, U	<b>0 to 10000</b> (1/10 °C ) 1/10 °F %..	-	-
TSX ASZ 401	4 outputs	<b>+/- 10 V</b>	<b>-10000 to +10000</b> (1)	<b>MAST</b> FAST	-
TSX ASZ 200	2 outputs	<b>+/- 10 V</b> 0..20 mA <b>4..20 mA</b>	<b>-10000 to +10000</b> <b>0 to +10000</b> (1) ..	<b>MAST</b> FAST	
<b>(1) This parameter can not be modified</b> <b>(2) Only in fast cycle</b> <b>(3) All the channels are used</b>					



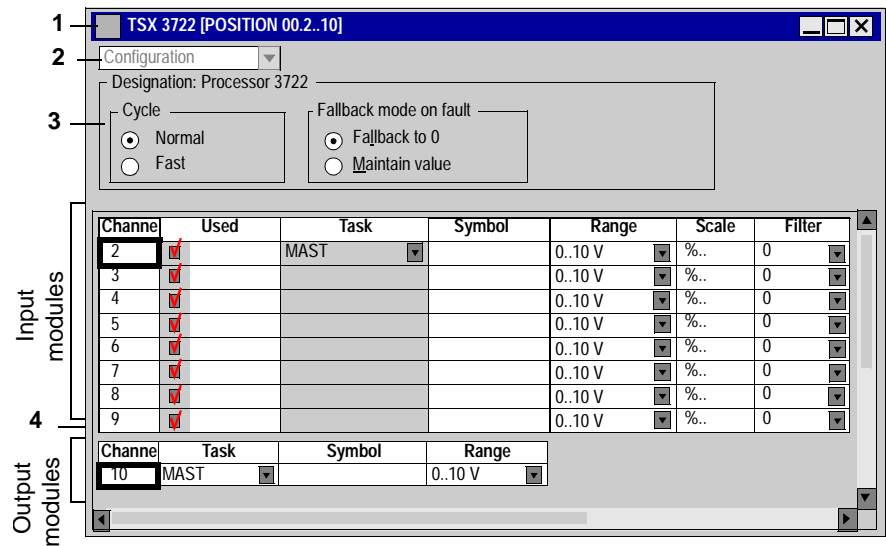
Module	Number of channels	Display format of the current value	Display	Task assignment	Filtering
TSX AMZ 600	4 inputs	+/- 10 V 0..10V 0..20 mA <b>4..20 mA</b>	<b>-10000 to +10000</b> <b>0 to +10000 (1)</b> .. ..	<b>MAST</b> FAST	0..6
	2 outputs	+/- 10 V 0..10V 0..20 mA <b>4..20 mA</b>	<b>-10000 to +10000</b> <b>0 to +10000 (1)</b> .. ..	<b>MAST</b> FAST	
<b>(1) This parameter can not be modified</b> <b>(2) Only in fast cycle</b> <b>(3) All the channels are used</b>					



## Display of Channel Parameters

### Introduction to the Screen

This screen shows the module selected and displays its configured parameters. It also gives access to the Modify Parameters function or to the Debug function. Example of the configuration screen of the 37-22 processor:



**Note:** If there is no mouse, pressing the <Shift><F2> keys allows you to switch between zone 3 and zone 4.

### Zone 1

This bar shows the catalog reference and the position of the module in the PLC. In the case of an integrated interface, the information displayed is the PLC reference (TSX 37-22) and the channel address: 0.2 to 0.10.

### Zone 2

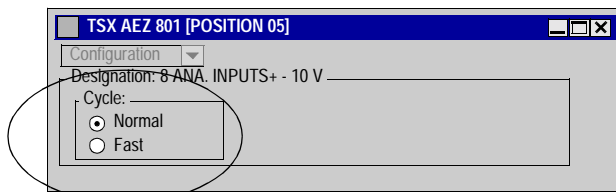
This command zone shows the current function (**Configuration** function) and allows you to choose the **Debug** function via a drop-down list box.



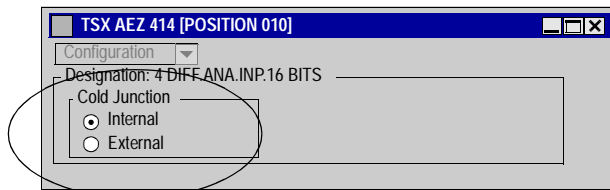
**Zone 3**

This "module" level zone contains the short title of the module (for example: 4 analog outputs +/- 10V) and in certain cases provides additional information, such as:

- For the TSX AEZ 801 / 802 input modules and for the interface integrated in the TSX 37-22 bases, the input scanning cycle:
  - **Normal** (by default): the inputs are sampled every 32 ms
  - **Fast**: only the inputs declared **used** are sampled

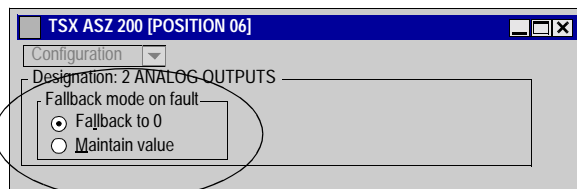


- For the TSX AEX 414 module and if a thermocouple range is selected, the cold junction compensation executed: **Internal** (by default) or **External**.



**In the case of external cold junction, channel 0 is forced after confirmation, in the Pt100 range.**

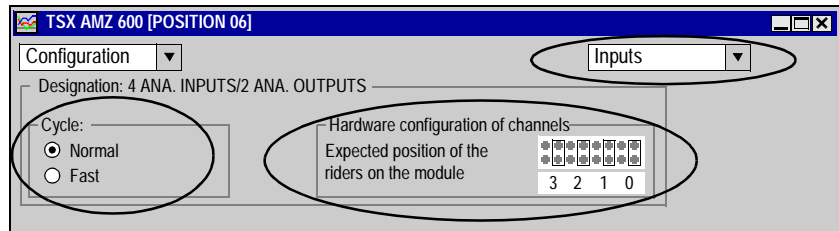
- For the output modules or for the interface integrated in the bases TSX 37-22, the output fallback mode in the case of a fault: **Fallback to 0** (by default) or **Maintain value**.



- For the analog input/output module **TSX AMZ 600** the screens are different between the inputs and outputs.
  - Inputs:
    - Normal** (by default): the inputs are sampled every 16 ms
    - Fast**: only the inputs declared **used** are sampled

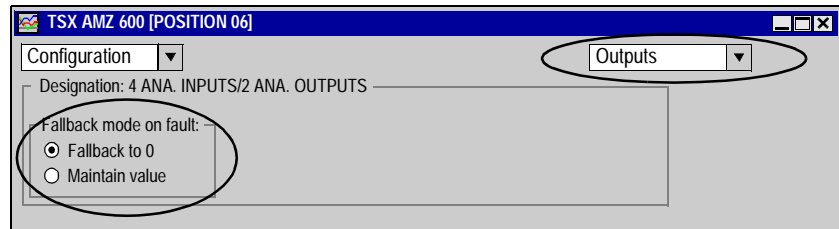


The "channel hardware configuration" gives the slot of the riders to be placed by the user depending on the selected range. Change of configuration between "inputs" and "outputs" is easy to access via the drop-down menu.



- Outputs: the additional information is: the output fallback mode in the case of a fault: **Fallback to 0** (by default) or **Maintain value**.

Change of configuration between "inputs" and "outputs" is easy to access via the drop-down menu.





**Zone 4**

This "channel" level zone shows the parameters configured for each of the **used** channels of the module. For the unused channels, only the channel number and the symbol are displayed.

Details of the different columns:

- **Channel:** number of the input or output channel.
  - **Used:** reduces the module cycle time when the fast scanning cycle is selected (only the used channels are scanned).
  - **Task:**
    - For the input modules, shows the task into which the inputs are assigned: task **MAST** or task **FAST** (only in fast scanning cycle). When this is possible (in fast scanning cycle), a drop-down list box allows the task to be modified.  
**Note:** The TSX AEZ 414 module inputs are always assigned to the MAST task.
    - For the output modules, shows the task at the end of which the outputs will be updated: **MAST** or **FAST**. A drop-down list box allows this task to be modified.
  - **Symbol:** symbol defined by the user and associated with the language object of the channel. If the channel has no associated symbol, this field is empty.
  - **Range:** current input or output channel range. According to the type of module, this may be:
    - Electric (+/- 10 V, 0..10 V, 0..5 V, 1..5 V, 0..20 mA or 4..20 mA)
    - Thermocouple (of type B, E, J, K, L, N, R, S, T and U)
    - Thermostatic probe (Pt100 or Ni1000)
  - **Scale:** display format of the current value This may be:
    - Normalized 0..10000 or +/- 10000 (%..)
    - User
  - **Filter:** value of the filter on the current value:
    - 0 : no filtering
    - 1 and 2: little filtering
    - 3 and 4: medium filtering
    - 5 and 6: strong filtering
-



# 23.2                    Modifying an input channel’s parameters

## At a Glance

**What’s in this Section?**                    This section contains the different procedures for modifying the parameters of an analog module input channel.

**What’s in this Section?**                    This Section contains the following Maps:

Topic	Page
Modification of the Scanning Cycle	251
Modification of the Task Assigned to the Module Inputs	252
Modification of the Input Range	253
Modification of the Display Format	254
Modification of the Filtering Value	256



# Modification of the Scanning Cycle

## General

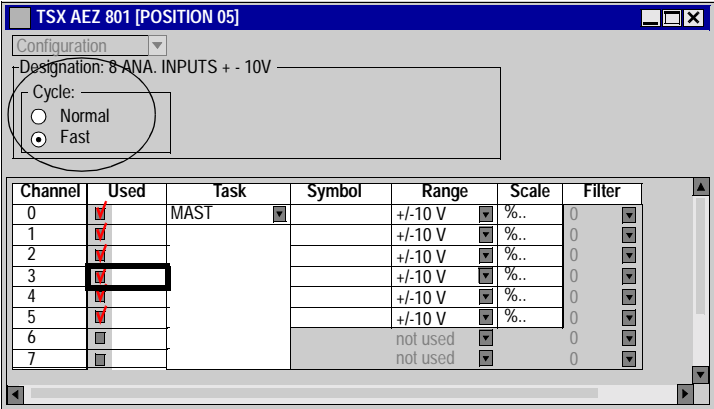
The scanning cycle can be modified on a TSX AEZ 801 / 802, AMZ 600 module or on the interface integrated in the TSX 37-22 bases.

These parameters (**Normal / Fast** and **Used**) can not be modified in online mode, if the application has been transferred into the PLC with their default values (normal cycle and all channels used).

For the TSX AMZ 600, the screen is slightly different but the principle is the same.

## Procedure

To modify the scanning cycle, proceed as follows:

1	Access the configuration editor.
2	<p>Select the type of cycle required using the two command buttons:</p> <ul style="list-style-type: none"> <li>• Normal</li> <li>• Fast</li> </ul> <p>Example of a screen for a TSX AEZ 801 (equivalent to the AMZ 600):</p> 
3	If the fast cycle is selected, in the <b>Used</b> column check only the used channels.



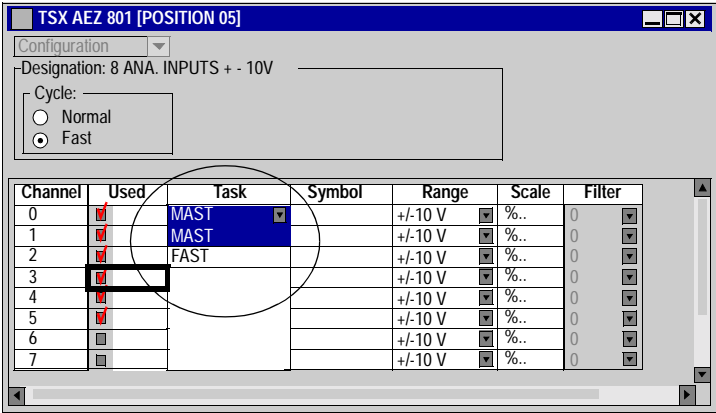
## Modification of the Task Assigned to the Module Inputs

### General

This function can only be modified on the TSX AEZ 801 / 802, AMZ 600 modules or on the interface integrated in the TSX 37-22 bases.  
For the TSX AMZ 600, the screen is slightly different but the principle is the same.

### Procedure

To modify the task assigned to the inputs, proceed as follows:

1	Access the configuration editor.
2	<div>In the drop-down list box click on the <b>Task</b> column: Example of a screen for a AEZ 801 (equivalent to the AMZ 600):</div> <div></div>
3	Select MAST or FAST in the list.



# Modification of the Input Range

## Procedure

To modify the input range, proceed as follows:

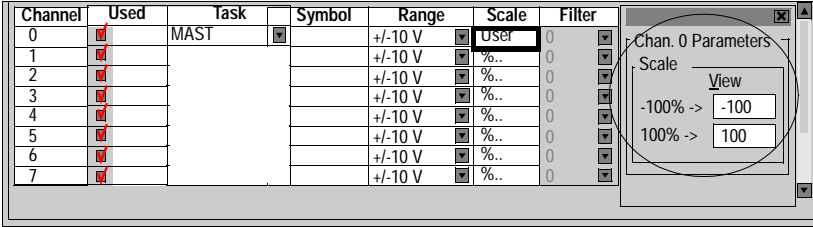
1	Access the configuration editor.																																																															
2	<p>In the drop-down list box click on the <b>Range</b> column:</p> <p>Illustration:</p> <div><table><tr><th>Channel</th><th>Used</th><th>Task</th><th>Symbol</th><th>Range</th><th>Scale</th><th>Filter</th></tr><tr><td>0</td><td><input checked="" type="checkbox"/></td><td>MAST</td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr><tr><td>1</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr><tr><td>2</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr><tr><td>3</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>not used</td><td>%..</td><td>0</td></tr><tr><td>4</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr><tr><td>5</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr><tr><td>6</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr><tr><td>7</td><td><input checked="" type="checkbox"/></td><td></td><td></td><td>+/-10 V</td><td>%..</td><td>0</td></tr></table></div>	Channel	Used	Task	Symbol	Range	Scale	Filter	0	<input checked="" type="checkbox"/>	MAST		+/-10 V	%..	0	1	<input checked="" type="checkbox"/>			+/-10 V	%..	0	2	<input checked="" type="checkbox"/>			+/-10 V	%..	0	3	<input checked="" type="checkbox"/>			not used	%..	0	4	<input checked="" type="checkbox"/>			+/-10 V	%..	0	5	<input checked="" type="checkbox"/>			+/-10 V	%..	0	6	<input checked="" type="checkbox"/>			+/-10 V	%..	0	7	<input checked="" type="checkbox"/>			+/-10 V	%..	0
Channel	Used	Task	Symbol	Range	Scale	Filter																																																										
0	<input checked="" type="checkbox"/>	MAST		+/-10 V	%..	0																																																										
1	<input checked="" type="checkbox"/>			+/-10 V	%..	0																																																										
2	<input checked="" type="checkbox"/>			+/-10 V	%..	0																																																										
3	<input checked="" type="checkbox"/>			not used	%..	0																																																										
4	<input checked="" type="checkbox"/>			+/-10 V	%..	0																																																										
5	<input checked="" type="checkbox"/>			+/-10 V	%..	0																																																										
6	<input checked="" type="checkbox"/>			+/-10 V	%..	0																																																										
7	<input checked="" type="checkbox"/>			+/-10 V	%..	0																																																										
3	Select the range required in the list.																																																															



## Modification of the Display Format

### Electric Ranges

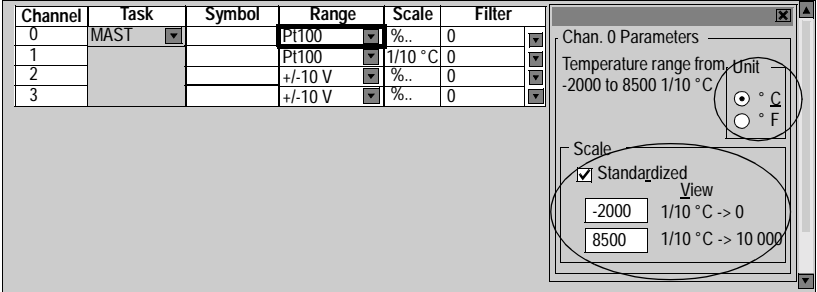
To modify the display format, proceed as follows:

1	Access the configuration editor.
2	Double click in the cell of the channel to be modified in the <b>Scale</b> column or press<Enter> with the cursor positioned on the same cell:
3	<div>The Channel parameters dialog box appears. Illustration:</div> <div></div>
4	If the default values are selected, the channel parameters display zone shows %... In the opposite case (user display), it shows <b>User</b> .



**Thermal Ranges  
(TSX AEZ 414)**

To modify the display format, proceed as follows:

1	Access the configuration editor.
2	Double click in the cell of the channel to be modified in the <b>Scale</b> column or press <Enter> with the cursor positioned on the same cell:
3	<p>The Channel parameters dialog box appears.</p> <p>Illustration:</p> 
4	Select the unit in which the readings are to be displayed (Degrees Celsius or Fahrenheit) using the two command buttons in the Unit zone.
5	<p>Then select the display mode in the Scale zone:</p> <ul style="list-style-type: none"> <li>● In temperature: <b>Standardized</b> box not checked</li> <li>● Standardized: <b>Standardized</b> box checked</li> </ul>
6	If the standardized display is selected, if necessary, modify one or both display limits. In this case, the display becomes user standardized.
7	Close the Channel parameters dialog box to confirm the selections.
8	<ul style="list-style-type: none"> <li>● If temperature display is selected, the channel parameters display zone shows <b>1/10°C or 1/10°F</b></li> <li>● If the standardized display (default or user) is selected, the channel parameters display zone shows %..</li> </ul>



## Modification of the Filtering Value

### Procedure

To modify the filtering value, proceed as follows:

1

Access the configuration editor.

2

In the drop-down list box click on the **Filter** column.

Illustration:

Channel	Used	Task	Symbol	Range	Scale	Filter
0	<input checked="" type="checkbox"/>	MAST		+/-10 V	User	0
1	<input checked="" type="checkbox"/>			+/-10 V	%..	0
2	<input checked="" type="checkbox"/>			+/-10 V	%..	1
3	<input checked="" type="checkbox"/>			+/-10 V	%..	2
4	<input checked="" type="checkbox"/>			+/-10 V	%..	3
5	<input checked="" type="checkbox"/>			+/-10 V	%..	4
6	<input checked="" type="checkbox"/>			+/-10 V	%..	5
7	<input checked="" type="checkbox"/>			+/-10 V	%..	0

3

Select the filtering value required in the list.

4

The alpha coefficient value of the selected filter and the associated response time are then displayed in the status bar at the bottom of the screen.

**Reminder:** In fast scanning cycle, filtering is inhibited on the TSX AEZ 801 / 802, AMZ 600 modules or on the interface integrated in the TSX 37-22 bases.



# 23.3                    Modifying an output channel's parameters

## At a Glance

**What's in this Section?**                    This section contains the different procedures for modifying the parameters of an analog module output channel.

**What's in this Section?**                    This Section contains the following Maps:

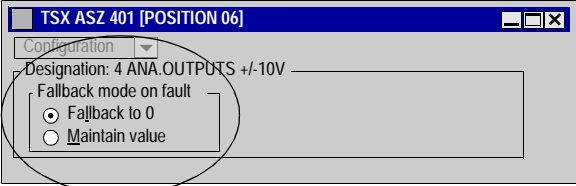
Topic	Page
Modification of the Fallback Mode	258
Modifying the task to which the output is assigned	259
Modification of the Output Range (TSX ASZ 200 and TSX AMZ 600)	260



## Modification of the Fallback Mode

**Procedure**

To modify the fallback mode, proceed as follows:

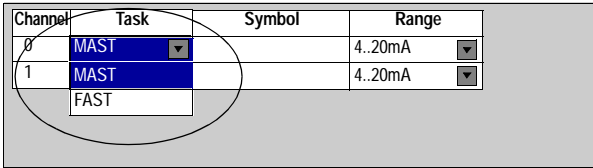
1	Access the configuration editor.
2	<div> <div>Select the fallback mode using the two command buttons.</div> <div>Example of a screen for a TSX ASZ 401 (equivalent to the AMZ 600):</div> <div>  </div> </div>



# Modifying the task to which the output is assigned

## Procedure

To modify the task, proceed as follows:

1	Access the configuration editor.
2	<div>Choose the task from the pull-down box list. Illustration: </div>



## Modification of the Output Range (TSX ASZ 200 and TSX AMZ 600)

### Procedure

To modify the output range, proceed as follows:

1	Access the configuration editor.																				
2	<p>Select the required range in the list in the drop-down list box: Example of a screen for a TSX AEZ 200 (equivalent to the AMZ 600):</p> <div><table><tr><th>Channel</th><th>Task</th><th>Symbol</th><th>Range</th></tr><tr><td>0</td><td>MAST <input type="checkbox"/></td><td></td><td>4..20mA <input checked="" type="checkbox"/></td></tr><tr><td>1</td><td></td><td></td><td>+/- 10V</td></tr><tr><td></td><td></td><td></td><td>0..20mA</td></tr><tr><td></td><td></td><td></td><td>4..20mA</td></tr></table></div> <p>For the TSX AMZ 600 module there is another range: 0..10V.</p>	Channel	Task	Symbol	Range	0	MAST <input type="checkbox"/>		4..20mA <input checked="" type="checkbox"/>	1			+/- 10V				0..20mA				4..20mA
Channel	Task	Symbol	Range																		
0	MAST <input type="checkbox"/>		4..20mA <input checked="" type="checkbox"/>																		
1			+/- 10V																		
			0..20mA																		
			4..20mA																		
3	The new selection appears in the channel parameters display zone.																				



---

# Debugging function

24

---

## At a Glance

### What's in this Chapter?

This chapter describes the Debugging function of the analog modules on the PLC Micro.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Introduction to the Debugging function	262
Display of Channel Parameters	263
Module Diagnostics Display	265
Deletion of Module Channel Forcing	267
Display of Detailed Channel Diagnostics	268
Modification of the Filtering Value	269
Channel Forcing / Delete Forcing	270



## Introduction to the Debugging function

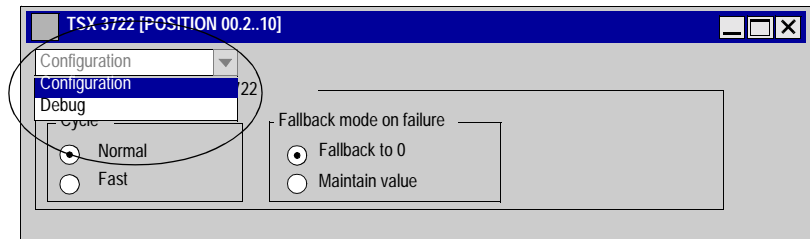
---

### General

This debugging function can be accessed for each module from the channel parameters (See Display of Channel Parameters, p. 246) display screen. This is only possible if the PLC is in on-line.

The debug screen is displayed:

- Either from the pull-down dialog box which is used to alternate access to the parameter **Debugging** and **Configuration** functions.



- Or from the **View** menu in the PL7 Micro software and the **Debugging** option.

The Debugging function is used to display in on-line mode, the parameters for each of the analog module channels, including the interface built-in to the PLC TSX 37-22. This function is also used to access diagnostics for the module and selected channel when a fault is present.

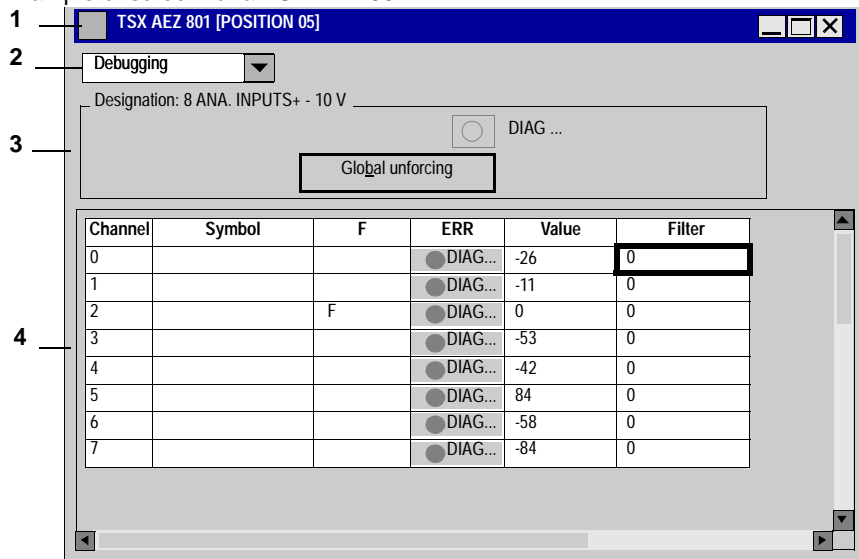
---



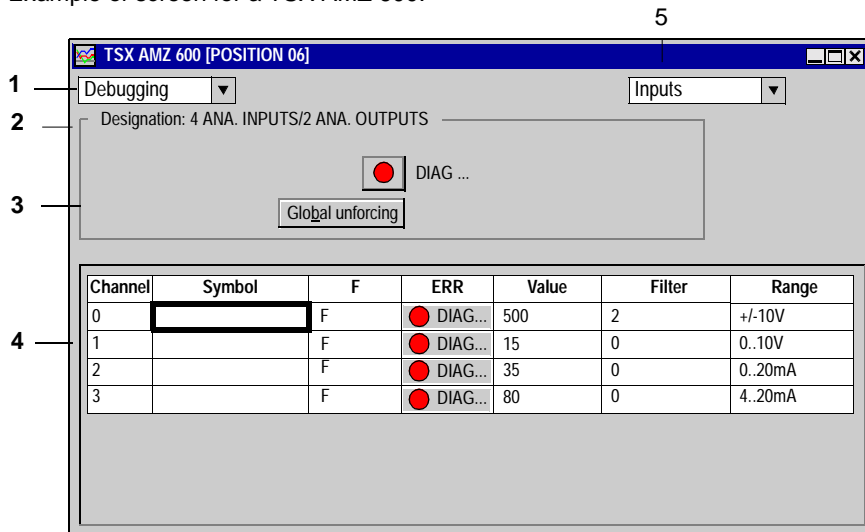
## Display of Channel Parameters

### Introduction to the Screen

This screen shows the module selected and displays in real time the value and state of each of its channels. It also gives access to the adjustment of certain channel parameters (forcing of the input or output value, adjustment of the filtering value, etc.). Example of screen for a TSX AEZ 801:



Example of screen for a TSX AMZ 600:





<b>Zone 1</b>	This bar shows the catalog reference and the position of the module in the PLC. In the case of an integrated interface, the information displayed is the PLC reference (TSX 37-22) and the channel address: 0.2 to 0.10.
<b>Zone 2</b>	This command zone shows the current function ( <b>Debug</b> function) and allows you to choose the <b>Configuration</b> function via a drop-down list box.
<b>Zone 3</b>	<p>This "module" level zone contains the short title of the module (for example: 8 analog inputs +/- 10 V). It also offers two command buttons which allow respectively:</p> <ul style="list-style-type: none"><li>● Access to the module diagnostics when the module has a fault: this is shown by the light that is integrated in the diagnostics access button, which turns red.</li><li>● To globally delete all channel forcing.</li></ul>
<b>Zone 4</b>	<p>This "channel" level zone displays in real time the value and status of each of the module's channels.</p> <p>Details of the different columns:</p> <ul style="list-style-type: none"><li>● <b>Channel</b>: number of the input or output channel.</li><li>● <b>Symbol</b>: symbol defined by the user and associated with the language object of the channel. If the channel has no associated symbol, this field is empty.</li><li>● <b>F</b>: forcing status of the channel: F if the channel is forced or no indication when the channel is not forced.</li><li>● <b>ERR</b>: channel status: the ERR indication indicates that there is a channel fault.</li><li>● <b>Value</b>: channel value.</li><li>● <b>Filter</b>: value of the filter on the current value:<ul style="list-style-type: none"><li>● 0 : no filtering</li><li>● 1 and 2: little filtering</li><li>● 3 and 4: medium filtering</li><li>● 5 and 6: strong filtering</li></ul></li><li>● <b>Range</b>: range value (only on TSX AMZ 600).</li></ul>
<b>Zone 5</b>	This command zone, only on the <b>TSX AMZ 600</b> , allows you to choose between inputs and outputs.



## Module Diagnostics Display

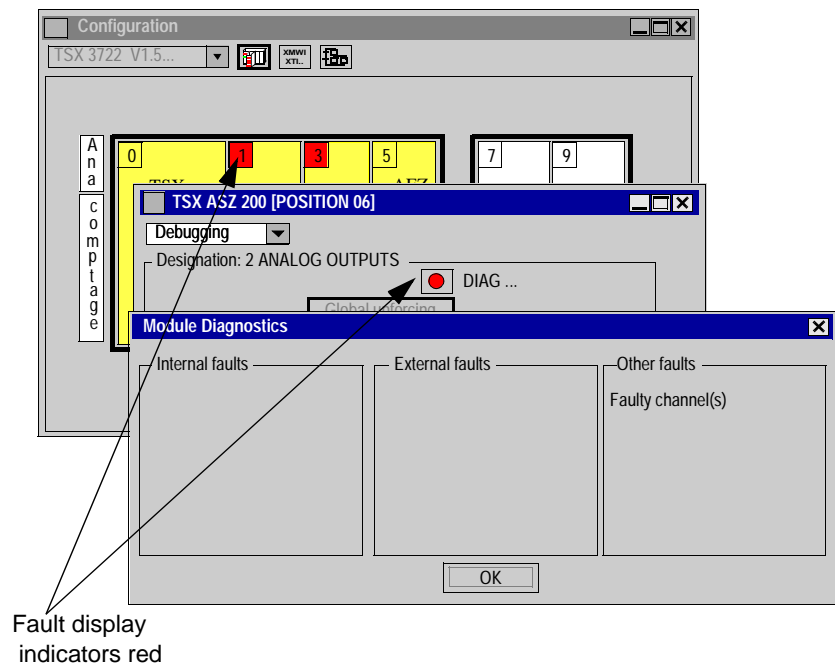
### At a Glance

When there is a module fault, a certain number of LEDs, which can be accessed in the configuration editor screens, turn red.

- Module position LED at the level of the screen representing it (first configuration editor screen).
- LED integrated in the **DIAG** command in the "module" level zone.

In addition, pressing the **DIAG** command button accesses the **Module Diagnostics** screen which displays the current module faults, classed according to category: internal faults, external faults or other faults.

Illustration for a ASZ 200 (identical for other modules):





## List of Module Faults

List of module faults that can appear during diagnostics:

- Module failure
- Faulty channel(s)
- Configuration fault
- Module missing or switched off

**Note:** In the event of a configuration fault or missing module, access to the module diagnostics screen is not possible. In this case the following message appears on the screen: "The module is missing or different from the one configured to this position".

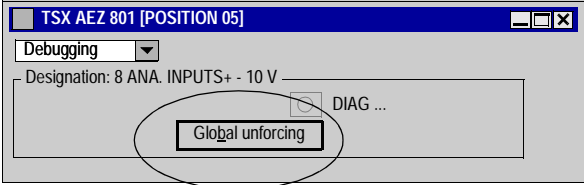
---



# Deletion of Module Channel Forcing

**Procedure**

To delete forcing of the channels of a module, proceed as follows:

1	Access the Module debug screen.
2	<div>Click on the Global unforcing button: Example of a screen for a TSX AEZ 801 (equivalent to the AMZ 600): </div>
3	The forcing of all forced channels is deleted.

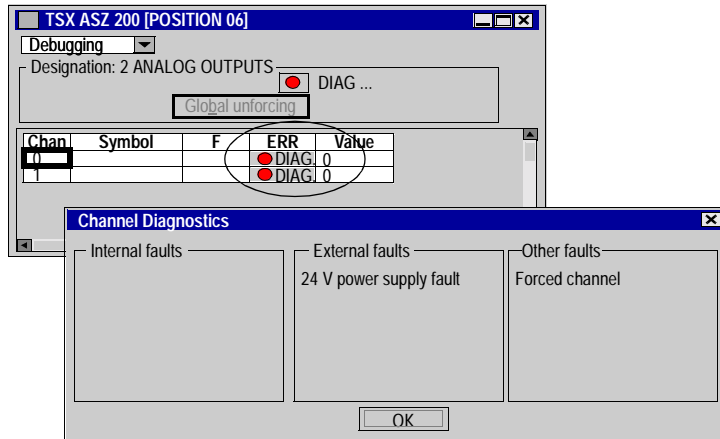


## Display of Detailed Channel Diagnostics

### At a Glance

When a channel has a fault, the **DIAG** command button associated with this channel in the ERR column turns red.

Pressing the **DIAG** command button then accesses a **Channel Diagnostics** screen (identical to the one for Module Diagnostics) which shows the channel faults, classified according to category: internal faults, external faults or other faults. Illustration:



### List of Channel Faults

List of channel faults that can appear during diagnostics:

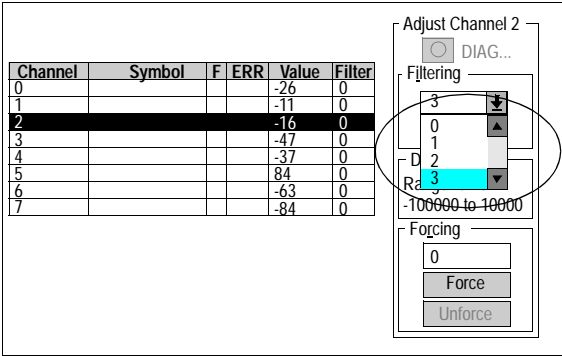
- Sensor link fault
- Range overshoot fault
- Configuration fault
- Application fault
- Forced channel
- Supply fault 24V (for TSX AEX 414 and TSX AMZ 600 outputs).



## Modification of the Filtering Value

### Procedure

To modify the filtering value, proceed as follows:

1	Access the Module debugging screen.
2	Double click on the line of the channel to be modified to bring up the Channel Adjust window.
3	Select a filter value from the list in the Filtering drop-down list dialog box. Illustration:
	 <p>The illustration shows a screenshot of the 'Adjust Channel 2' dialog box. On the left, a table lists channels 0 through 7. Channel 2 is selected and highlighted in black. To the right of the table is the 'Adjust Channel 2' dialog box. It has a 'DIAG...' button at the top. Below it is the 'Filtering' section, which contains a drop-down menu. The menu is open, showing values 0, 1, 2, 3, and 4. The value 3 is selected and highlighted in blue. Below the drop-down menu is a range indicator showing '-100000 to 10000'. At the bottom of the dialog box is the 'Forcing' section, which contains a text box with the value '0' and two buttons: 'Force' and 'Unforce'.</p>
4	Confirm by closing.  <b>Note:</b> Whatever the scanning cycle (normal or fast), modification of filtering is not possible on an unused channel (TSX AEZ 801 / 802, AMZ 600 and interface integrated in bases TSX 37-22). For the used channels, modification of filtering is also impossible in fast cycle.



## Channel Forcing / Delete Forcing

---

### Procedure

To force or delete forcing of a channel, proceed as follows:

1	Access the Module debugging screen.
2	Double click on the line of the channel to be modified to bring up the Channel Adjust window.



3

Entred a value in the **Forcing** entry field.  
Illustration for a TSX AEZ 801:

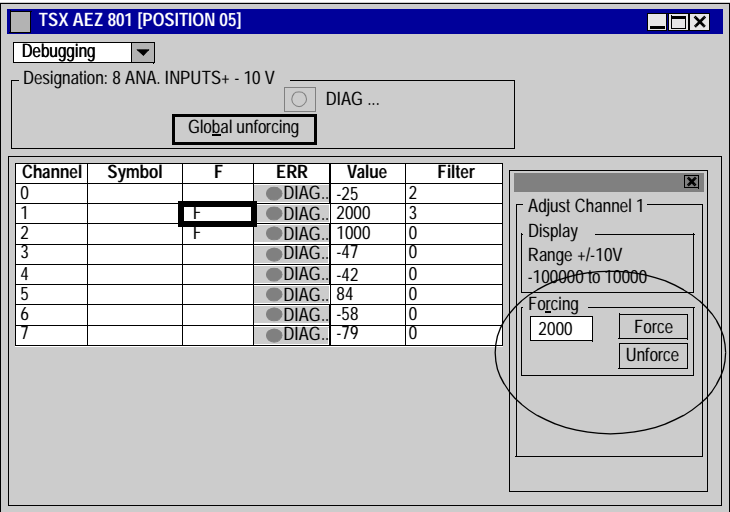
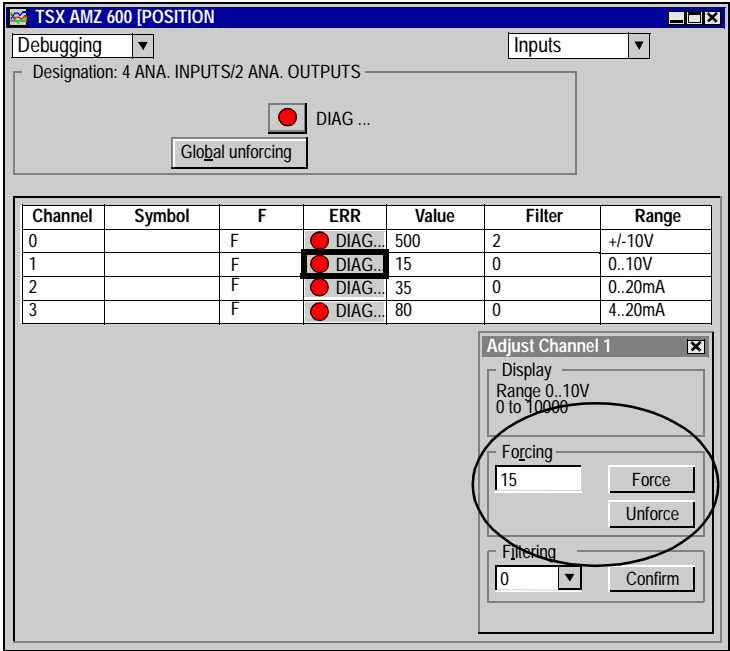


Illustration for a TSX AMZ 600:





4	Press the <b>Force</b> command button to force the selected channel.
5	The information F appears in the cell in the F column that corresponds to the forced channel.
6	To delete forcing, select the channel and press the <b>Unforce</b> command button.

---

**Forcing an Input**

When an analog input is forced, the value present at the input **is not available**. The forced value is shown in the screen's **Values** and **Forcing** fields.

Forcing of inputs is active whether the PLC is in RUN or STOP.

Whatever the scanning cycle (normal or fast), forcing of inputs is not possible on an unused channel (TSX AEZ 801 / 802, TSX AMZ 600 and interface integrated in bases TSX 37-22).

---

**Forcing an Output**

When an analog output is forced, the value present at the module output is shown in the screen's **Forcing** field. The value calculated by the application remains displayed in the screen's **Value** field.

Forcing of outputs is only valid if the PLC is in RUN.

---



---

# Bits and words associated with the analog application

25

---

## At a Glance

### What's in this Chapter?

This chapter introduces the bits and words associated with the analog application on PLC Micro.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Implicit exchange objects associated with the analog application	274
Explicit exchange objects associated with inputs/outputs	275
Configuration objects associated with the analog application	277



## Implicit exchange objects associated with the analog application

---

### At a Glance

These are objects used for programming and diagnostics of analog modules. These objects are exchanged automatically on each task cycle in which the channels of the module or the built-in interface are configured.

---

### Channel values

The following table gives the words containing the analog channel values:

Address	Function
%IWx.i	Value of input channel i of the analog input module in position x. (For inputs integrated into the TSX 37-22 basic module, x = 0 and i is between 2 and 9).
%QWx.i	Value of output channel i of the analog output module in position x. (For the output integrated into the TSX 37-22 base module, x = 0 and i = 10).

Example: the word %IW5.3 contains the value present in input 3 of the module in position 5.

---

### Error bit objects

The following table gives the error bits:

Address	Meaning
%Ix.i.ERR	When set to 1, indicates that the input channel i of the module in position x is faulty.
%Ix.MOD.ERR	When set to 1, indicates that the module in position x is faulty.

---



## Explicit exchange objects associated with inputs/outputs

---

### General

These objects are exchanged via instructions. They are only useful for advanced programming of the application-specific function.

---

### Internal words

Table of words available according to the different types of module:

Address	Meaning	Built-in analog	Input modules	Output modules
%MWx.MOD.2	Module status word	Yes	Yes	Yes
%MWx.i	Exchange in progress	Yes	Yes	Yes
%MWx.i.1	Exchange report	Yes	Yes	Yes
%MWx.i.2	Channel status word	Yes	Yes	Yes
%MWx.i.7	Adjustment (Filtering coefficient)	Yes	Yes	-

---



**Details of the  
explicit  
exchange words**

Analog module x status word:

Address	Meaning (when = 1)
%MWx.MOD.2:X0	Module failure
%MWx.MOD.2:X1	Faulty channel(s)
%MWx.MOD.2:X2	Reserved
%MWx.MOD.2:X3	Self-testing in progress
%MWx.MOD.2:X4	Reserved
%MWx.MOD.2:X5	Configuration fault
%MWx.MOD.2:X6	Module is missing or off
%MWx.MOD.2:X7 to 15	Reserved

Channel i status word

Address	Meaning (when = 1)
%MWx.i.2:X0	Sensor link fault
%MWx.i.2:X1	Range overshoot fault
%MWx.i.2:X2	Reserved
%MWx.i.2:X3	24V supply fault (output modules)
%MWx.i.2:X4	Module failure
%MWx.i.2:X5	Configuration fault
%MWx.i.2:X6	Communication fault
%MWx.i.2:X7	Adjustment parameter value outside limits
%MWx.i.2:X8	Channels not ready (module initialization in progress)
%MWx.i.2:X9 to 12	Reserved
%MWx.i.2:X13	Forced channel
%MWx.i.2:X14 and 15	Reserved

---



## Configuration objects associated with the analog application

---

### At a Glance

These are the objects which can only be accessed in read format and which contain the configuration parameters.

---

### Constant words

Table of constant words available according to the different types of module:

Address	Meaning	Analog built-in	Input modules	Output modules
%KWx.i.0	Channel parameters defined at configuration	Yes	Yes	Yes
%KWx.i.1	Minimal scale values defined at configuration	No	Yes	No
%KWx.i.2	Maximum scale values defined at configuration	No	Yes	No

---



**Details of word**  
**%KW.x.i.0**

Command word for the input channels:

Address	Meaning
%KWx.0:X0 to X5	Range of use coded on 6 bits: 00 0000 = reserved 00 0001 = 4-20 mA or Thermocouple B 00 0010 = 0-20 mA or Thermocouple E 00 0011 = Thermocouple J 00 0100 = Thermocouple K 00 0101 = Thermocouple N 00 0110 = Thermocouple R 00 0111 = 1-5 V or Thermocouple S 00 1000 = Thermocouple T 00 1001 = 5 V or Thermocouple U 00 1010 = 10 V or Thermocouple L 00 1011 to 01 1111 = reserved 10 0000 = Pt100 10 0010 = reserved 10 0010 = Ni1000 10 0011 to 11 1101 = reserved 11 1110 = discrete range 11 1111 = reserved
%KWx.0:X6	Polarity: <ul style="list-style-type: none"> <li>if electrical range:  0 = unipolar, 1 = bipolar</li> <li>if temperature range:  0 = °C, 1 = °F</li> </ul>
%KWx.0:X7	Type of range: 0 = electrical, 1 = temperature
%KWx.0:X8	Scanning cycle 0 = normal, 1 = fast
%KWx.0:X9	Channel used 0 = normal, 1 = fast
%KWx.0:X10	not used
%KWx.0:X11	Not used
%KWx.0:X12	Cold junction compensation: 0 = internal, 1 = external (only on TSX AEZ 414)
%KWx.0:X13	Scale: 0 = manufacturer, 1 = user
%KWx.0:X14 and X15	Not used



Command word for the output channels:

Address	Meaning
%KWx.0:X0 to X5	Output range coded on 6 bits: 00 0000 = reserved 00 0001 = 4-20 mA 00 0010 = 0-20 mA 00 0011 to 00 1001 = reserved 00 1010 = 10 V 00 1011 to 11 1111 = reserved
%KWx.0:X6	Polarity: 0 = unipolar, 1 = bipolar
%KWx.0:X7	Reserved
%KWx.0:X8	Fallback mode 0 = fallback, 1 = maintain
%KWx.0:X9 and X10	Reserved
%KWx.0:X11	Reserved
%KWx.0:X12 to X15	Reserved







---

# Operator Dialog functions



---

## Introduction

### Subject of this part

This part introduces the specific built-in Operator Dialog functions of PL7 and describes the software setup.

### What's in this part?

This Part contains the following Chapters:

Chapter	Chaptername	Page
26	General presentation of the Operator Dialog functions	283
27	Built-in DOP functions	285
28	Appendices	341







---

# General presentation of the Operator Dialog functions

26

---

## General presentation

### Introduction

The PL7 software allows the implementation of specific functions, designed to simplify the use of an operator dialog terminal (version 2.0 and higher) on a TSX Micro PLC.

These functions are the basic elements of the PL7 language.

They allow the following actions to be carried out without referring to the communication media between the command console CCX 17 and the PLC:

- displaying messages,
- displaying message and alarm message groups,
- entering values from the PLC program.

Hence the Operator Dialog functions are completely integrated in the PLC application and support:

- data cohesion,
- unique saving,
- easy maintenance,
- simplified consoles.
- ...

The processing of these functions occurs asynchronously to the processing of the operative task that enabled their activation.

### How to access a DOP function

See Accessing a specific function, method or procedure type instruction, p. 53.

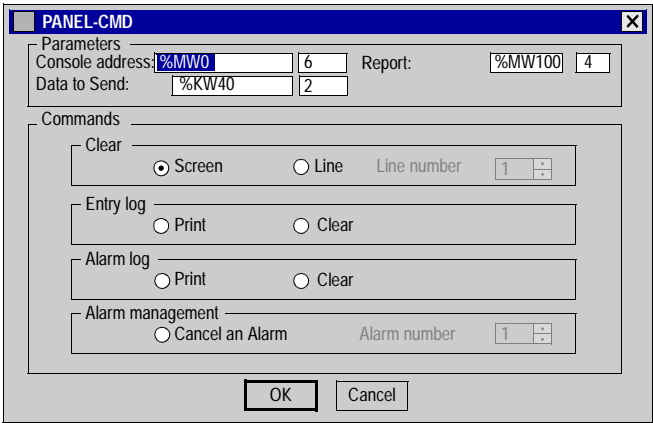


Conditions for using built-in DOP functions

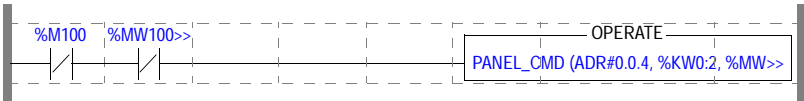
The built-in DOP functions require a program space of 1 Kword (4.7 Kwords for ADJUST). Variables must be reserved for the data you want to display (use the constants %KWi).

Example of a DOP function

The illustration below shows an example of a DOP function written in the different PL7 languages using the assisted entry of the library functions.



LD Language



IL Language

```
LDN      %100
ANDN     %MW100:X0
[PANEL_CMD (ADR#0.0.4, %KW0:2, %MW0: 4 ) ]
```

ST Language

```
IF NOT %M100 AND NOT %MW100. X0 THEN PANEL_CND (%MW0 . 6,%KV0 . 2,%MW100 . 4),
END_-F
```



---

### Introduction

#### Subject of this chapter

This chapter describes the different built-in DOP functions and shows how they are set-up in PL7.

#### What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
27.1	Description of the parameters common to the different DOP functions	287
27.2	Description of the built-in DOP functions	302







## 27.1 Description of the parameters common to the different DOP functions

### Introduction

#### Subject of this section

This section shows the main parameters, divided into zones, of the built-in DOP functions.

**Note:** The parameters specific to a DOP function are shown at the level of the function concerned.

#### What's in this Section?

This Section contains the following Maps:

Topic	Page
General	288
Parameters Zone: Terminal address	289
Parameters field: Data to be sent	291
Parameter field: Data to be received:	293
Parameters field: Report	294
Message field	298
Field zone	300



# General

## Introduction

The Built-in DOP functions are part of the procedure category; they do not return value, but they do possess several parameters, some of which have to be filled in.

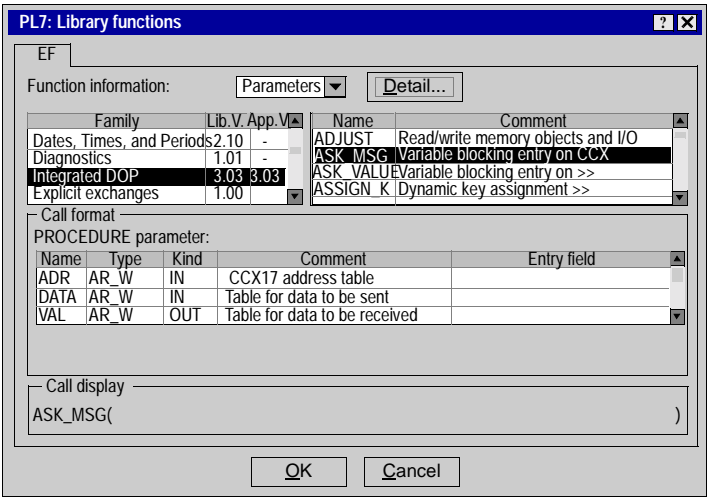
The functions use three types of parameters:

- in read only (IN) format, acknowledged at the start of function execution,
- in write only (OUT) format, set at close of function execution,
- in read and write format (IN/OUT), whose contents have been acknowledged at the start of function execution and are then updated by the function results.

The parameter type is displayed in the **Type of Parameter of the ...** column located in the **Library functions** screen.

## Illustration

The illustration below gives an insight into the **Library functions** screen where the various parameters of the selected function are displayed.





## Parameters Zone: Terminal address

### At a Glance

This parameter contains the access path (addressing) of the CCX 17 interface terminal.

This path can be either:

- located in a table of six internal words (%MW),
- located in a table of six internal constants (%KW),
- entered directly as an immediate value in the form of an ADR# operator.

ADR# is thus assimilated into a table of six internal consecutive words.

### Addressing in words or constants

The table below gives the meaning of the different words composing the terminal address.

Word number	Description	
	Most significant	Least significant
%MWi / %KWi	6 (Uni-telway)	0
%MWi +1 / %KWi+1	254 (1)	0 (1)
%MWi +2 / %KWi+2	Module number	0
%MWi +3 / %KWi+3	CCX 17 address SYS	channel
%MWi +4 / %KWi+4	0	0
%MWi +5 / %KWi+5	0	0
Key		
(1)	CCX 17 terminal operation only authorizes one intra-station address. Consequently, the {Network.Station} pair systematically takes the value {0.254}. Writing to this field is optional.	

**Note:** The keyword SYS (value 254) corresponds to addressing the system channel (UNI-TE server) as a communication channel.

### Uni-telway Addressing

Addressing for a CCX 17 connected to the Uni-telway bus takes the following form:

```
ADR#{<Network>.<Station>}<module>.<channel>.<CCX 17 address>,
ADR#{<Network>.<Station>}<module>.<channel>.<SYS>,
ADR#<module>.<channel>.<CCX 17 address>,
ADR#<module>.<channel>.<SYS>.
```



**Addressing examples**      **A CCX 17 slave at address UTW 4-5 connected to a master TSX Micro PLC via the Uni-telway integrated link (channel 0).**  
An address can be entered in several ways (example using the internal words %MW0 to %MW5):

If you are using:	in the function entry Help screen, ...	you must enter the program ...
internal words (or constants)	Example: <div><div>Parameters</div><div>Terminal address: <input type="text" value="%MW0"/> <input type="text" value="6"/></div></div>	<pre>%MW0:=16#0600; %MW1:=16#FE00; %MW2:=16#0000; %MW3:=16#0400; %MW4:=16#0000; %MW5:=16#0000; or %MW0:6:=ADR#0.0.4;</pre>
ADR syntax	Example: <div><div>Parameters</div><div>Terminal address: <input type="text" value="ADR#0.0.4"/></div></div>	-
	-	FONCTION_DOP (ADR#0.0.4, ...

**Note:** When using constants, you must first initialize the %KW0:6 table in the Data editor, by successively assigning the %KWis of which it is composed.



## Parameters field: Data to be sent

### Introduction

Data to be sent is individual to each type of MMI function.  
The data can be found either in the PLC application, or in the CCX 17 when it has its own application.

### Locating items within the PLC application

In the case where data is to be found in the PLC application, it can be sent from:

- a table of internal words (%MW),
- a table of internal constants (%KW).

The table below shows the structure of data sent in this way.

Word number	Role
1	Contains a contains a marker for value 16#CC17, and has a double role: <ul style="list-style-type: none"> <li>• it allows the Help screen to identify a correct message and to redisplay the values on the entry screen in order to help modify or display default values.</li> <li>• it allows the function being carried out to check that the table received does actually contain a message for a CCX 17. In fact, it is possible to call an Integrated DOP function in a program without going via the Help/Control screens.</li> </ul> In the case of an unmarked message, the function can immediately return an error back to the application without sending unreliable data to the console.
2	Contains the command number.
3	Contains the length of data to be sent.
4, 5, ...	Contains the data to be sent.

**Note:** For reasons of efficiency, the <Data to be sent> parameter can be programmed using %KWi constants. In this way the software automatically initializes this data field with adequate values.  
Selecting %MW bars access to different fields in the entry help box for integrated DOP functions. It is then necessary to establish, either manually or by program, the contents of the data to be sent (see PL7 MMI 17 software documentation).



**Location in a CCX 17 with application**

When data is in a CCX 17, the data to be sent is limited to DOP function execution commands.

This data can be sent from the PLC application:

- from a table of internal words (%MW),
- from a table of internal constants (%KW),
- directly using an immediate integer value.

The table above shows data structure when using a table.

Word number	Role
1	Contains a command number
2	Contains the data to be sent to the console.

---



## Parameter field: Data to be received:

---

### Introduction

This parameter only affects functions **ASK\_MSG** and **ASK\_VALUE**.  
The data is located in an **%MWi** internal word table (length 2 table maximum).

<p><b>Note:</b> By operating the MMI console, the <b>Data to be received</b> parameter contains the value entered. If the status message variable is different from the data to be received, it is not changed by the entry. It only affects the display on the CCX 17.</p>
---

---



## Parameters field: Report

---

### Introduction

The report contains the parameters for managing asynchronous communication functions.

It is common to all integrated DOP functions.

---

### Report structure

The report uses a table of four internal words (%MW) containing different parameters such as:

- information on function activity,
- the exchange number which identifies the transaction in progress (useful when using the CANCEL communication function),
- the exchange report split into two feedback codes:
  - the communication level,
  - the operation level,
- the value of the timeout which is used to monitor absent responses,
- the number of bytes to be sent and/or the number of bytes received.

The table below show the structure of the report.

Word number	Most significant byte	Least significant byte	Report management
%MWi	Exchange number	Bit 0: activity bit	System
%MWi+1	Operation report	Communication report	System
%MWi+2	Timeout		User
%MWi+3	Length		System

---

### %MWi:X0 activity bit

This bit indicates the state of the application of the communication function.

It is set on 1 when the function starts and falls back to 0 when a response is received, at the end of a timeout, or if the operation is cancelled (CANCEL function).

---

### Exchange number

While a function is being sent, the system automatically assigns it a number which is used to identify the exchange.

This number serves as a reference to stop the exchange in progress, if necessary (on using CANCEL).

---



## Communication report

The communication report provides information on the communication aspect of the transaction.

This report is significant when the value of the activity bit changes from 1 to 0.

**Note:** The communication report does not affect the ADJUST function.

The different values of this report are indicated in the following table:

Value	Significance of the communication report (least significant byte)
16#xx00	Successful exchange
16#0001	Stopping the exchange via a timeout
16#0002	Stopping the exchange on user request (CANCEL function)
16#0003	Incorrect address format (length is other than 6)
16#0004	Incorrect destination address (addresses unauthorized for the CCX 17, e.g.: addresses being broadcast)
16#0005	Invalid report
16#xx06	Specific parameters are invalid (particularly those concerning data to be sent)
16#0007	Destination missing
16#0008	Reserved
16#0009	Size of reception buffer is insufficient
16#000A	Size of transmission buffer is insufficient
16#000B	System resources missing (communication saturation)
16#xx14	Negative response from the CCX 17 or from the PLC (ADJUST function)
16#00FF	Message refused (the CCX 17 is not in a state where it can process it)

**Note:** The function can detect an error in the parameters before activating the exchange. In this case the activity bit remains 0, the report is initialized with the values corresponding to the fault.



**Operation report** The operation report details the result of the operation on the remote application. It is significant if the communication report has the following values:

- 16#00,
  - 16#06,
  - 16#14 (except for the ADJUST function).
- In other cases, the operation report is worth 0.

The different values of this report are indicated in the following table:

Communication report (least significant byte)	Significance of the operation report	Integrated DOP functions
16#0000	Generic positive result	All
16#1006	Number of management words below 24	ADJUST only
16#1106	Non-existent type object to be read (greater than 8)	
16#1206	Inconsistency between bits RDEC and SINC	
16#1306	Invalid value to be written	
16#1406	Broadcast address (ALL) prohibited	All except ADJUST
16#6506	{network.station} pair different from {0.254}	
16#6606	Data to be sent does not have a 16#CC17 marker	
16#6706	Invalid size of data to be sent	
16#6806	Invalid CCX 17 response	
16#6906	Length of "Data to be received" is insufficient	
16#FF06	CCX 17 link inoperational	
16#0114	Command not recognized	
16#0214	Command queue capacity has been exceeded	
16#0414	Size of command is less than the minimum required size	
16#0814	Command refused because application transfer in progress	
16#1414	Inaccessible object	ADJUST only
16#1514	System error	
16#2014	Incorrect data	All except ADJUST



## Timeout

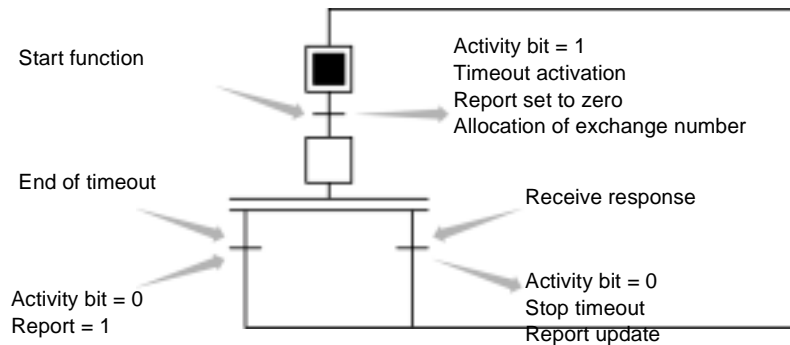
Timeout determines the maximum waiting time for a response. The time base for this parameter is 100 ms.

value 0 corresponds to an infinite wait value. In this case, the **CANCEL** function should be used.

**Note:** Value 0 is mandatory for functions **ASK\_MSG** and **ASK\_VALUE**.

If the timeout period has elapsed, the exchange terminates with an error report (value 1). Similarly, the system refuses any response received at the end of the timeout.

### Example



**Note:** The timeout value of a communication function must be sufficient to guarantee receiving a response to the question asked. This time period depends on the type of network and on the load in effect at the moment of the transaction.

## Length

The length parameter is used to store the number of bytes received after receiving a message for functions **ASK\_MSG** and **ASK\_VALUE**.

For the other functions, this parameter is worth 0.



## Message field

---

### Introduction

The message field groups together the different elements that make up the data to be displayed on the CCX 17 screen.

These elements are:

- the message text,
  - the different display attributes (position, size, etc.).
  - the message print command via the CCX 17.
- 

### Message text

This field is used to enter the message text that is to be displayed on the CCX 17 console.

The length of the message is 40 characters max.

The table below details the types of characters authorized.

Characters	Comments
ASCII code above 32 (20h)	Characters that can be displayed directly or by a combination of the and keys.
The _ sign (underscore)	<p>This character is reserved by the system to specify the optional display field for the variable associated with the message.</p> <p>To specify the position of the variable field, the character must be entered in the appropriate place. The system then automatically calculates the number of necessary for the length of the variable display.</p>

---

### Print

This parameter orders the command console to print a message when it appears on the console screen.

---

### Overprint

This parameter refers to alarm messages only. Enabling this parameter overprints the alarm message as soon as it appears.

---

### Line

This parameter specifies the line where the message must be displayed.

Position	Value
Minimum	1
Default	1
Maximum	16

---



**Column** This parameter specifies the column where the first message character is positioned.

Position	Value(s)
Minimum	1
Default	1
Maximum	40
Automatic (1)	Left, Centered, Right
Key	
(1)	This mode can be accessed by selecting automatic mode. It is local to the function.

**Attributes** This parameter sets the attributes of the message display.  
The different modes are:

- Normal (when no check box has been selected),
- Flashing,
- Reverse video.

**Size** This parameter specifies the format of all the text characters or the variable to be displayed.  
Possible options are:

- Standard,
- Double.

**Clear** This field is used to associate a single command that has been executed with a message before the message has been displayed.  
Options are:

- None (no command is associated with a message),
- Line (clears the line on which the message is to be displayed),
- Screen (completely clears the screen).

**Note:** If no variable has been associated with a message, this command does not run (so use the PANEL\_CMD function).



## Field zone

---

### Introduction

The Field zone is used to set the different parameters for the object which is associated with a message.

These parameters are:

- the type of object,
- the display format,
- ...

### Field type

This parameter sets the type of object which is associated with a message. The length of the message is 40 characters max.

The table below shows the different object types possible.

Type of object	Description
None	No field is associated with the message displayed.
Address	The object associated with the message is a variable.
Date	The message is correlated to the current PLC date.
Time	The message is correlated to the current PLC time.

### Symbol

This parameter specifies the symbol of the variable associated with the message. It must be set in the station database. The address associated with this symbol is automatically taken into account on confirming the screen.

**Note:** In the case of a TSX Agent connected to the FIPIO bus, the variable is read in the bus master PLC, and not in the function sender PLC.

### Address

This parameter specifies the address of the variable associated with the message. When a symbol is associated with the variable, it is automatically taken into account. Authorized objects can be:

- internal bits (%Mi),
- internal words (%MWi),
- double internal words (%MDi).

### Comment

This field displays the variable comment for viewing. This comment is set in the application data editor.



**Refresh**

This function specifies whether the variable contained in the message should be periodically refreshed while being displayed (function is active by default).

**Display format**

This parameter specifies the variable display format.  
The table below shows the different formats available.

Format	Associated parameters
ASCII	-
Numerical	Signed (1)
	Number of digits before the decimal point (1),
	Number of digits after the decimal point (1),
Key	
(1)	these associated parameters can be accessed by clicking on the <b>Modify</b> button.

**Note:** From the parameters chosen, the software automatically calculates the display format.



# 27.2 Description of the built-in DOP functions

## Introduction

**Subject of this section** This section describes the different built-in DOP functions.

**What's in this Section?** This Section contains the following Maps:

Topic	Page
List of the built-in DOP functions	303
SEND_MSG function	304
GET_MSG function	306
ASK_MSG function	309
SEND_ALARM function	311
DISPLAY_MSG function	314
DISPLAY_GRP function	315
DISPLAY_ALRM function	317
ASK_VALUE function	320
GET_VALUE function	321
CONTROL_LEDS function	323
ASSIGN_KEYS function	326
PANEL_CMD function	329
ADJUST function	332



## List of the built-in DOP functions

### Introduction

The built-in DOP functions make it possible to:

- control the main functions of a CCX 17 console not containing an application (this has been neither configured nor loaded by the external design software).
- command a CCX 17 console containing an application created with the MMI17 WIN or PL7 M17 OS/2 products.

The table below shows the various built-in DOP functions.

Function	Description	Application	
		Without	With
SEND_MSG	Displaying status messages contained in the PLC memory on CCX 17 with or without variable.	X	-
GET_MSG	Free entry (asynchronous) of PLC variable values associated with status messages.	X	-
ASK_MSG	Blocking entry (synchronous) of a PLC variable value associated with status messages.	X	-
SEND_ALARM	Displaying alarm messages contained in the PLC memory.	X	-
DISPLAY_MSG	Displaying a status message contained in the memory of the CCX17.	-	X
DISPLAY_GRP	Displaying a group of status messages contained in the CCX17 memory.	-	X
DISPLAY_ALRM	Displaying an alarm message contained in the CCX17 memory.	-	X
ASK_VALUE	Blocking entry (synchronous) of values for the PLC variables associated with a status message contained in the CCX17 memory.	-	X
GET_VALUE	Free entry (asynchronous) of values for the PLC variables associated with a status message contained in the CCX17 memory.	-	X
CONTROL_LEDS	Controlling the CCX 17 LEDs and relays.	X	X
ASSIGN_KEYS	Configuring the CCX 17 command keys.	X	X
PANEL_CMD	Sending a generic command.	X	X
ADJUST	Language object adjustment.	X	X



## SEND\_MSG function

**Role** This function is used to display a message with a dynamic variable (if necessary) on a CCX 17 console screen.

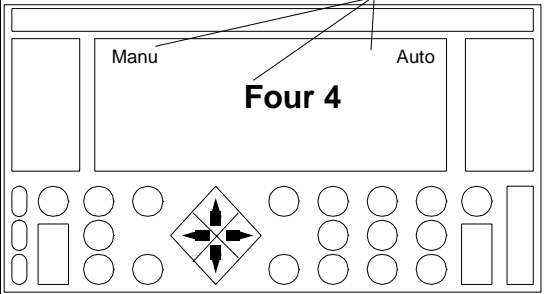
**Implementation** Implementation of the SEND\_MSG function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

**Application example** The example given below makes use of the SEND\_MSG function to display two status messages on the screen of a T CCX 1720 W console without an application. This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

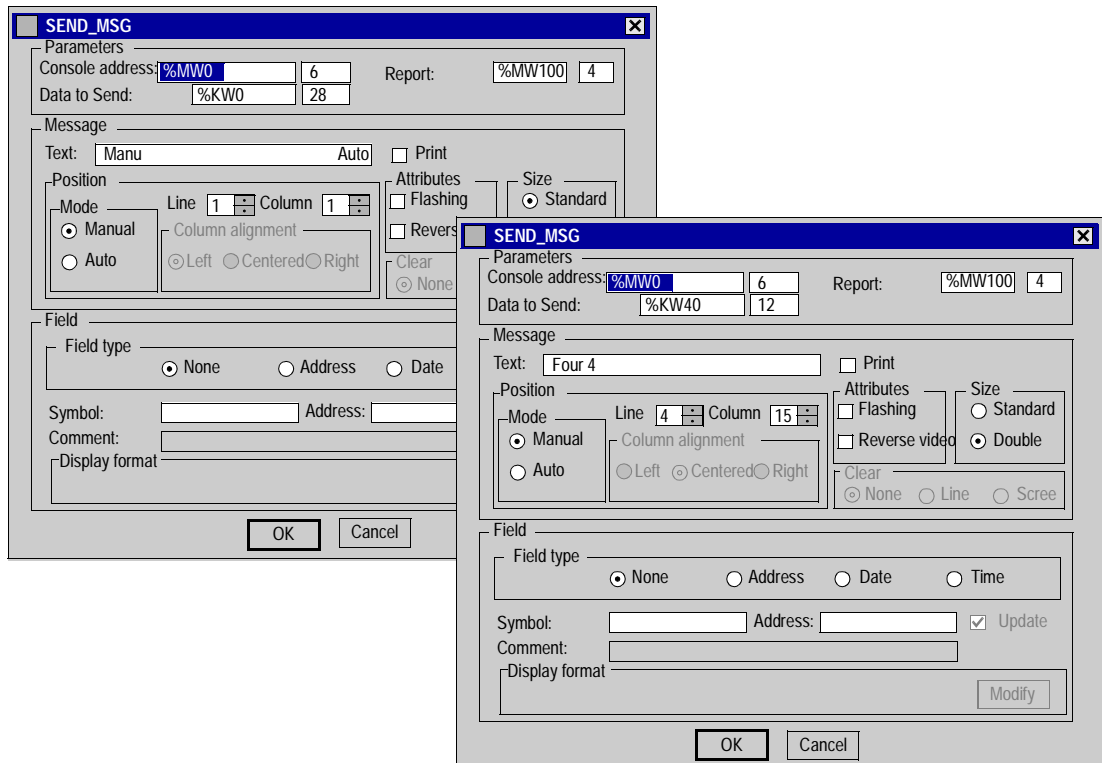
**Note:** This same example, using a CCX 17 with application, is shown using the DISPLAY\_GRP function.

Application description	Variables used
<p>The aim of this example is, after setting application to RUN (%S13 =1):</p> <ul style="list-style-type: none"><li>initializing the PLC variables,<ul style="list-style-type: none"><li>writing the console address (ADR#0.04) in a word table,</li><li>adjustment of timeout to 50 s,</li><li>conditions of execution,</li></ul></li><li>clearing the console screen (see PANEL_CMD function),</li><li>to display status messages on the console screen:<ul style="list-style-type: none"><li><b>Manu</b> and <b>Auto</b> (standard format, positioned on line 1, Column 1),</li><li><b>Four 4</b> (double format, automatic centering, Line 4),</li></ul></li><li>storing the function display execution.</li></ul>	<p>%MW0:6: console address %KW0:x: message 1 data to send %KW40:x: message 2 data to send %MW100:4: report %MW100:X0: activity bit %MW102: timeout %M100:2: conditions of activation,</p>



Introduction to the console	Program corresponding to the application
 <p>Fixed messages</p> <p>Manu</p> <p>Auto</p> <p><b>Four 4</b></p>	<pre> (* INIT console address, condition, timeout *) IF %S1.3 THEN   %MW0:6:=ADR#0.0.4;   %MW102:=500;   %M100:2:=0; END_IF; (*Write Auto, Manu and Four4 messages*) IF NOT %M100 AND NOT %MW100:X0 THEN   SEND_MSG(%MW0:6,%KW0:28,%MW100:4);   SET %M100; END_IF; IF NOT %M101 AND NOT %MW100:X0 THEN   SEND_MSG(%MW0:6,%KW0:28,%MW100:4);   SET %M101; END_IF; </pre>

Entry help screens corresponding to the application:



**SEND\_MSG**

Parameters

Console address:   Report:

Data to Send:

Message

Text:   ☐ Print

Position

Line  Column

Mode

☒ Manual ☐ Auto

Column alignment

☒ Left ☐ Centered ☐ Right

Attributes

☐ Flashing ☐ Revers

Size

☒ Standard

Field

Field type

☒ None ☐ Address ☐ Date

Symbol:  Address:

Comment:

Display format

OK Cancel

**SEND\_MSG**

Parameters

Console address:   Report:

Data to Send:

Message

Text:  ☐ Print

Position

Line  Column

Mode

☒ Manual ☐ Auto

Column alignment

☒ Left ☐ Centered ☐ Right

Attributes

☐ Flashing ☐ Reverse video

Size

☐ Standard ☒ Double

Field

Field type

☒ None ☐ Address ☐ Date ☐ Time

Symbol:  Address:  ☒ Update

Comment:

Display format

Modify

OK Cancel



## GET\_MSG function

---

**Role**

This function is used to display, on the screen of a CCX 17 console, a message with a variable which can be modified by the operator.

The entry is made in multiple mode. The operator thus has the possibility of entering several variables successively, and the PLC program processes the entered value when the variable appears.

---

**Implementation**

Implementation of the GET\_MSG function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

---

**Specific parameter: Value**

The Value parameter specifies the characteristics of the value which is associated to the variable.

If the choice is...	The value...
Not checked	entry on the CCX 17 console keyboard is free (1).
Limited	entry on the CCX 17 console keyboard must be within the limits defined by the Min and Max values in the Entry field in order to be taken into account by the application (1).
Increment	display on the CCX 17 console screen is incremented or decremented from the value of the increment (1).
<b>Key</b>	
(1)	The value or the increment entered on the CCX 17 console keyboard follows the display format (e.g.: 9999.99), this determines the whole parts and decimal parts authorized by the user entry.

---



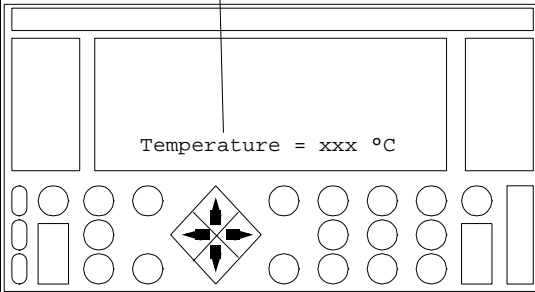
**Example of application**

The example given below makes use of the GET\_MSG function to display a status message containing a variable which can be modified on the screen of a T CCX 1720 W console without application.

This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

**Note:** This same example, using a CCX 17 with application, is shown using the GET\_VALUE function.

Application description	Variables used
<p>Initial conditions</p> <ul style="list-style-type: none"> <li>writing the console address (ADR#0.04) in a table of words,</li> <li>adjustment of timeout to 50 s,</li> <li>condition of execution,</li> </ul> <p>Application: The aim of this example is, on user request:</p> <ul style="list-style-type: none"> <li>to display a status message on the console screen: <ul style="list-style-type: none"> <li><b>Temperature = xxx °C</b> (standard format, automatic centering, Line 6, variable with attributes: increment of 50, three-digit integer, periodic refreshing),</li> </ul> </li> <li>storing the function execution.</li> </ul>	<p>%MW0:6: console address  %KW80:x: data to be sent.  %MW100:4: report  %MW100:X0: activity bit  %MW102: timeout  %M102: activation condition  %MW10: Temperature variable</p>

Introduction to the console	Program corresponding to the application
<p>Message displayed on user request</p>  <p>The diagram shows a console screen with a message display area at the top and a numeric keypad below it. The message display area shows the text "Temperature = xxx °C". A line points from the text "Message displayed on user request" to the message display area. The numeric keypad has a central diamond-shaped button with four arrows pointing up, down, left, and right.</p>	<pre>(* INIT console address, condition, timeout *) IF %S1.3 THEN   %MW0:6:=ADR#0.04;   %MW102:=500;   %M102:=0; END_IF; (* Write message Temp... *) IF NOT %M102 AND NOT %MW100:X0 THEN   GET_MSG(%MW0:6,%KW100:28,%MW100:4);   SET %M102; END_IF;</pre>



Entry help screen corresponding to the application:

GET\_MSG

Parameters

Console address:

%MW0

Report:

%MW100

4

Data to Send:

%KW1000

34

Message

Text:

Temperature= °C

☐ Print

Position

Line

6

Column

11

Mode

☒ Manual

☐ Auto

Column alignment

☐ Left

☐ Centered

☐ Right

Attributes

☐ Flashing

☐ Reverse video

Size

☒ Standard

☐ Double

Field

Symbol:Address:

%MV10

☒ Update

Comment:

Display format

999

Modify

Entry

Value

☐ Not Checked

☐ Limited

☒ Increment

Increment:

50

Ok

Cancel



## ASK\_MSG function

**Role** This function is used to display, on the screen of a CCX 17 console, a message with a variable which can be modified by the operator.  
The entry is made in synchronized mode. Therefore, it is possible to make one single operator entry at each message display.

**Implementation** Implementation of the ASK\_MSG function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

**Note:** It is strongly advisable to assign parameters for the timeout for an infinite period (see Parameters field: Report, p. 294 so as not to invalidate the ASK\_MSG function before the operator entry is made.

**Specific parameter: Value** The Value parameter specifies the characteristics of the value which is associated to the variable.

If the choice is...	The value...
Not checked	entry on the CCX 17 console keyboard is free (1).
Limited	entry on the CCX 17 console keyboard must be within the limits defined by the Min and Max values in the Entry field in order to be taken into account by the application (1).
Increment	display on the CCX 17 console screen is incremented or decremented from the value of the increment (1).
<b>Key</b>	
(1)	The value or the increment entered on the CCX 17 console keyboard follows the display format (e.g.: 9999.99), this determines the whole parts and decimal parts authorized by the user entry.



**Example**                    The illustration below shows an example of ASK\_MSG function entry.

ASK\_MSG

Parameters

Console address:ADR#LL0.1.7.SYS

Data to receive:%KW2002

Data to Send:%KW2034

Report:%KW104

Message

Text:ASK MSG:\_

Print

Position

Mode

Line5Column5

Column alignment

Manual

Auto

Left

Centered

Right

Attributes

Size

Flashing

Reverse video

Standard

Double

Field

Symbol:

Address:%MV100

Comment:

Display format

99

Modify

Entry

Value

Not Checked

Limited

Increment

Ok

Cancel



---

## SEND\_ALARM function

---

### Role

This function is used to activate, on the screen of a CCX 17 console, an alarm message present in the PLC.

**Note:** Alarm messages are always displayed on the second line of the screen (**Superimpose** parameter activated). They are dated and timed by the console which synchronizes them with the PLC clock.

### Implementation

Implementation of the SEND\_ALARM function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

**Note:** It is imperative that the associated message in the CCX 17 console is deactivated on the disappearance of the alarm in the PLC (see function PANEL\_CMD), in order to enable a potential new activation of this alarm.

### Specific parameter: Alarm number

The **Alarm number** parameter defines the alarm message identifier. Its value is between 900 and 999.

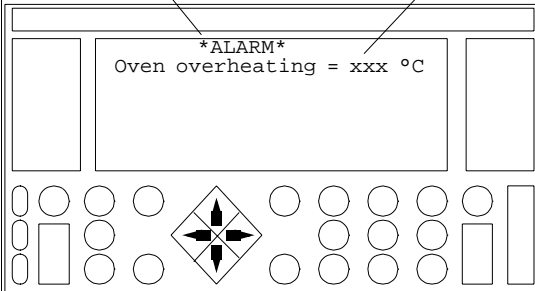
### Example of application

The example given below makes use of the SEND\_ALARM function to display an alarm message on the screen of a T CCX 1720 W console without application. This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

**Note:** This same example, using a CCX 17 with application, is shown using the DISPLAY\_ALARM function.

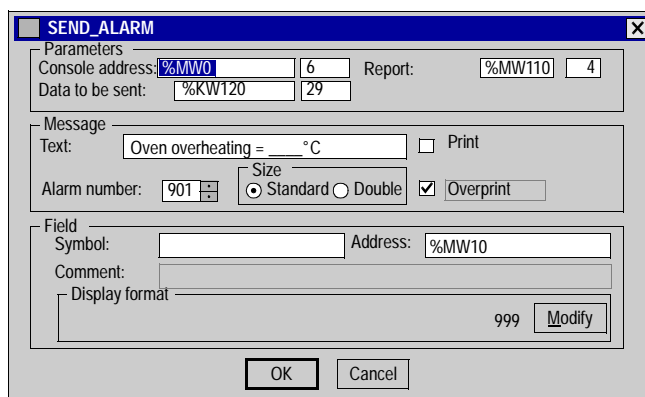


Application description	Variables used
<p>Initial conditions</p> <ul style="list-style-type: none"><li>● writing the console address (ADR#0.04) in a table of words,</li><li>● adjustment of timeout to 50 s,</li><li>● condition of execution,</li></ul> <p>Application: The aim of this example is:</p> <ul style="list-style-type: none"><li>● to detect overshoot of a temperature threshold (500 °C),</li><li>● to display an alarm message on the console screen:<ul style="list-style-type: none"><li>● <b>Overheat four = xxx °C</b> (standard format, automatic setting imposed by the function, variable with attributes: increment of 50, three-digit integer),</li></ul></li><li>● storing the function execution.</li></ul>	<p>%MW0:6: console address %KW80:x: data to be sent. %MW100:4: report %MW100:X0: activity bit %MW102: timeout %M103: activation condition %MW10: Temperature variable</p>

Introduction to the console	Program corresponding to the application
<p>System display</p> <p>Alarm message</p> 	<pre>(*INIT console address, condition, timeout *) IF %S1.3 THEN     %MW0:6:=ADR#0.0.4;     %MW102:=500;     %M103:=0; END_IF; (*Time overrun control*) IF %MW10&gt;500 THEN     SET %M12; ELSE RESET %M12; END_IF; (* Write Surch alarm message... *) IF %M12 AND NOT %M103 AND NOT %MW100:X0 THEN     SEND_ALARM(%MW0:6,%KW0:27,%MW100:4);     SET %M103; END_IF;</pre>



Entry help screen corresponding to the application:



The image shows a software dialog box titled "SEND\_ALARM" with a close button (X) in the top right corner. The dialog is organized into several sections:

- Parameters:** Contains two rows of input fields. The first row has "Console address:" followed by a text box containing "%MW0", a numeric box with "6", and "Report:" followed by a text box containing "%MW110" and a numeric box with "4". The second row has "Data to be sent:" followed by a text box containing "%KW120" and a numeric box with "29".
- Message:** Contains a "Text:" label followed by a text box with "Oven overheating = \_\_\_\_ °C" and a "Print" checkbox. Below this is an "Alarm number:" label followed by a numeric box with "901". To the right of the alarm number is a "Size" section with two radio buttons: "Standard" (which is selected) and "Double". To the right of the size section is an "Overprint" checkbox, which is checked.
- Field:** Contains a "Symbol:" label followed by an empty text box, and an "Address:" label followed by a text box containing "%MW10". Below these is a "Comment:" label followed by an empty text box.
- Display format:** A label followed by a large empty text box. To the right of this box is the number "999" and a "Modify" button.

At the bottom of the dialog are two buttons: "OK" and "Cancel".



## DISPLAY\_MSG function

---

Role	This function is used to display a message contained in the CCX 17 operator dialog console memory.
Implementation	Implementation of the DISPLAY_MSG function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.
Specific parameter: Message number	The <b>Message number</b> parameter defines the alarm message identifier contained in the CCX 17 memory. Its value is between 1 and 300.
Example	The illustration below shows an example of DISPLAY_MSG function entry.

**DISPLAY\_MSG**

Parameters

Console address: ADR# 0.0.4      Report: %MW10 4

Data to be sent: %KW200

Value of data to be sent

Message number: 1

OK      Cancel



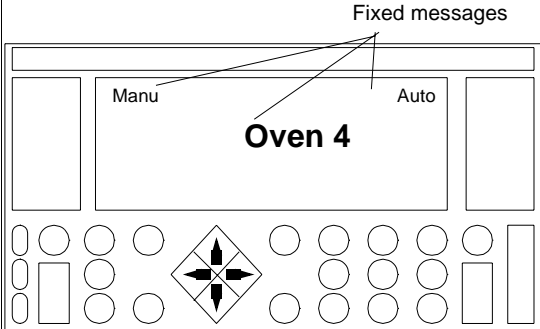
## DISPLAY\_GRP function

<b>Role</b>	This function is used to simultaneously display a group of messages contained in the CCX17 operator dialog console memory.
<b>Implementation</b>	Implementation of the DISPLAY_MSG function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.
<b>Specific parameter: Message group number</b>	The <b>Message group number</b> parameter defines the group message identifier contained in the CCX 17 memory. Its value is between 1 and 100.
<b>Example of application</b>	The example given below makes use of the DISPLAY_GRP function to display a group of two status messages on the screen of a T CCX 1720 W console containing an application. This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

**Note:** This same example, using a CCX 17 without application, is shown using the SEND\_MSG function.

Application description	Variables used
<p>The aim of this example is, after setting application to RUN (%S13 =1):</p> <ul style="list-style-type: none"> <li>initializing the PLC variables, <ul style="list-style-type: none"> <li>writing the console address (ADR#0.04) in word table,</li> <li>adjustment of timeout to 50 s,</li> <li>condition of execution,</li> </ul> </li> <li>clearing the console screen (see PANEL_CMD function),</li> <li>displaying the group of state n°1 messages contained in the CCX 17 application, for: <ul style="list-style-type: none"> <li>message n°1: <b>Manu</b> and <b>Auto</b> (standard format, moved to line 1, Column 1),</li> <li>message n°2: <b>Four 4</b> (double format, automatic centering, Line 4),</li> </ul> </li> <li>storing the function execution.</li> </ul>	<ul style="list-style-type: none"> <li>%MW0:6: console address</li> <li>%KW0:x: message 1 data to send</li> <li>%MW100:4: report</li> <li>%MW100:X0: activity bit</li> <li>%MW102: timeout</li> <li>%M100: activation condition</li> </ul>



Introduction to the console	Program corresponding to the application
 <p>The diagram shows a console interface with a central display area labeled 'Oven 4'. Above the display are two buttons labeled 'Manu' and 'Auto'. To the right of the 'Auto' button, there is a label 'Fixed messages' with arrows pointing to a series of small rectangular boxes. Below the display is a control panel with a central diamond-shaped directional pad and several circular buttons on either side.</p>	<pre>(* INIT console address, condition, timeout *) IF %S1.3 THEN   %MW0:6:=ADR#0.0.4;   %MW102:=500;   %M100:=0; END_IF; (*Write Auto, Manu and Four4 messages*) IF NOT %M100 AND NOT %MW100:X0 THEN   DISPLAY_GRP(%MW0:6,%KW0,%MW100:4);   SET %M100; END_IF;</pre>

Entry help screen corresponding to the application:

DISPLAY\_GRP

Parameters

Console address:

%MW10

6

Report:

%MW100

4

Data to be sent:

%KW0

Value of data to be sent

Message group number:

1

OKCancel



## DISPLAY\_ALARM function

**Role** This function is used to display an alarm message contained in the CCX17 operator dialog console memory.

**Implementation** Implementation of the DISPLAY\_ALARM function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

**Note:** It is imperative that the associated message in the CCX 17 console is deactivated on the disappearance of the alarm in the PLC (see function PANEL\_CMD), in order to enable a potential new activation of this alarm.

**Specific parameter: Alarm message number** The Alarm message number parameter defines the alarm message identifier contained in the CCX 17 memory. Its value is between 1 and 300.

An alarm can be activated exclusively by:

- letter box (requires MMI 17 WIN software),
- built-in DOP functions

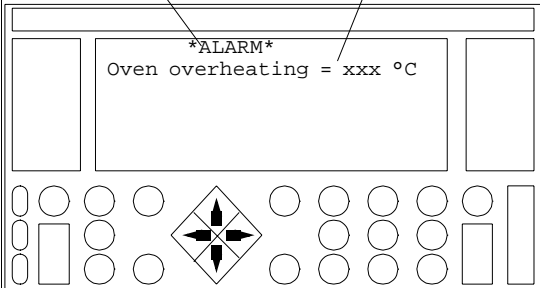
**Note:** Only the alarms with numbers higher than the length of the letterbox can be accessed by the DISPLAY\_ALARM function.

**Application example** The example given below makes use of the DISPLAY\_ALARM function to display an alarm message on the screen of a T CCX 1720 W console containing an application. This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

**Note:** This same example, using a CCX 17 with application, is shown using the SEND\_ALARM function.



Application description	Variables used
<p>Initial conditions</p> <ul style="list-style-type: none"><li>● writing the console address (ADR#0.04) in a word table,</li><li>● adjustment of timeout to 50 s,</li><li>● condition of execution,</li></ul> <p>Application: The aim of this example is:</p> <ul style="list-style-type: none"><li>● to detect overshoot of a temperature threshold (500 °C),</li><li>● to display the alarm message contained in the CCX 17 application, for:<ul style="list-style-type: none"><li>● <b>Overheat four = xxx °C</b> (standard format, automatic setting imposed by the function, variable with attributes: increment of 50, three-digit integer),</li></ul></li><li>● storing the function execution.</li></ul>	<p>%MW0:6: console address %KW80:x: data to be sent. %MW100:4: report %MW100:X0: activity bit %MW102: timeout %M103: activation condition %MW10: Temperature variable</p>

Introduction to the console	Program corresponding to the application
<p>System display</p> <p>Alarm message</p> 	<pre>(*INIT console address, condition, timeout *) IF %S1.3 THEN     %MW0:6:=ADR#0.0.4;     %MW102:=500;     %M103:=0; END_IF; (*Time overrun control*) IF %MW10&gt;500 THEN     SET %M12; ELSE RESET %M12; END_IF; (* Write Surch alarm message... *) IF %M12 AND NOT %M103 AND NOT %MW100:X0 THEN     DISPLAY_ALRM(%MW0:6,%KW0,%MW100:4);     SET %M103; END_IF;</pre>



Entry help screen corresponding to the application:

The screenshot shows a dialog box titled "DISPLAY\_ALARM" with a close button (X) in the top right corner. The dialog is divided into two main sections: "Parameters" and "Value of data to be sent".

**Parameters section:**

- Console address:** A text field containing "%MW0" followed by a numeric field containing "6".
- Data to be sent:** A text field containing "%KW150".
- Report:** A text field containing "%MW100" followed by a numeric field containing "4".

**Value of data to be sent section:**

- Number of alarm message:** A numeric field containing "1" with a small up/down arrow icon to its right.

At the bottom of the dialog, there are two buttons: "OK" and "Cancel".



## ASK\_VALUE function

**Role** This function is used to display a status message contained in the CCX17 operator dialog console memory on a CCX 17 console screen. This message includes a variable, which can be modified by the operator. The entry is made in synchronized mode. Therefore, it is possible to make one single operator entry at each message display.

**Implementation** Implementation of the ASK\_VALUE function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

**Note:** It is strongly advisable to assign parameters for the timeout for an infinite period (see Parameters field: Report, p. 294 so as not to invalidate the ASK\_VALUE function before the operator entry is made.

**Specific parameter: Status message number** This parameter defines the message identifier contained in the CCX 17 memory. Its value is between 1 and 300.

**Example** The illustration below shows an example of ASK\_VALUE function entry.

The screenshot shows a dialog box titled "ASK\_VALUE". It is divided into two main sections. The top section, labeled "Parameters", contains several input fields: "Console address" with the text "ADR# 0.0.4", a numeric field with the value "6", "Data to receive" with a dropdown menu showing "%MW10" and a numeric field with "2", and "Data to be sent" with a text box containing "%KW20" and "Report" with a dropdown menu showing "%MW20" and a numeric field with "4". The bottom section, labeled "Value of data to be sent", contains a "Status message number" field with the value "3". At the bottom of the dialog are "OK" and "Cancel" buttons.



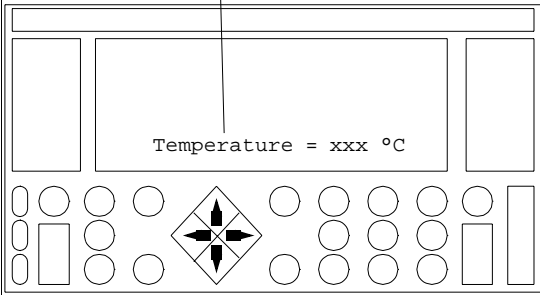
## GET\_VALUE function

<b>Role</b>	<p>This function is used to display a status message contained in the CCX17 operator dialog console memory on a CCX 17 console screen.</p> <p>This message includes a variable, which can be modified by the operator.</p> <p>Entry is made in multiple mode. The operator thus has the possibility of entering several variables successively, and the PLC program processes the entered value when the variable appears.</p>
<b>Implementation</b>	<p>Implementation of the GET_VALUE function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.</p>
<b>Specific parameter:</b> <b>Status message number</b>	<p>This parameter defines the message identifier contained in the CCX 17 memory. Its value is between 1 and 300.</p>
<b>Application example</b>	<p>The example given below makes use of the GET_VALUE function to display a status message containing a variable on the screen of a T CCX 1720 W console containing an application.</p> <p>This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).</p>

**Note:** This same example, using a CCX 17 without application, is shown using the GET\_MSG function.

Application description	Variables used
<p>Initial conditions</p> <ul style="list-style-type: none"> <li>writing the console address (ADR#0.04) in a word table,</li> <li>adjustment of timeout to 50 s,</li> <li>condition of execution,</li> </ul> <p>Application: The aim of this example is, on user request:</p> <ul style="list-style-type: none"> <li>initializing the PLC variables,</li> <li>displaying message n°4 contained in the CCX17 application, for:             <ul style="list-style-type: none"> <li><b>Temperature = xxx °C</b> (standard format, automatic centering, Line 6, variable with attributes: increment of 50, three-digit integer, periodic updating),</li> </ul> </li> <li>storing the function execution.</li> </ul>	<p>%MW0:6: terminal address</p> <p>%KW80:x: data to be sent.</p> <p>%MW100:4: report</p> <p>%MW100:X0: activity bit</p> <p>%MW102: timeout</p> <p>%M102: activation condition</p> <p>%MW10: Temperature variable</p>



Introduction to the console	Program corresponding to the application
<p>Message displayed on user request</p> 	<pre>(*INIT console address, condition, timeout *) IF %S1.3 THEN     %MW0:6:=ADR#0.0.4;     %MW102:=500;     %M102:=0; END_IF; (* Write message Temp... *) IF NOT %M102 AND NOT %MW100:X0 THEN     GET_VALUE(%MW0:6,%KW100:28,%MW100:4);     SET %M102; END_IF;</pre>

Entry help screen corresponding to the application:

GET VALUE

Parameters

Console address: %MW0 6 Report: %MW100 4

Data to be sent: %KW100

Value of data to be sent

Status message number: 4

OK

Cancel



---

## CONTROL\_LEDS function

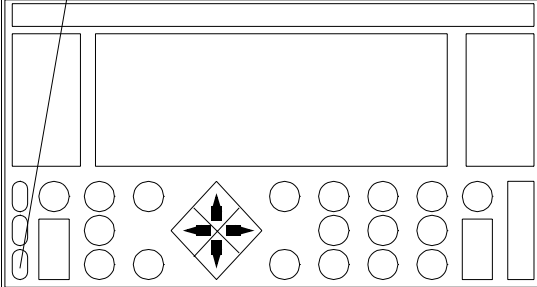
---

<b>Role</b>	<p>This function is used to control some CCX 17 console functions. These functions are:</p> <ul style="list-style-type: none"><li>● relay state (version 2.1 and above),</li><li>● state of the small luminous column LEDs.</li></ul> <p>The CONTROL_LEDS function is available whether or not the CCX 17 has an application.</p>
<b>Implementation</b>	<p>Implementation of the CONTROL_LEDS function is developed in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.</p>
<b>Specific parameter: LED status</b>	<p>This parameter defines the current status of each LED. the various states can be:</p> <ul style="list-style-type: none"><li>● Unchanged,</li><li>● Off,</li><li>● Flashing,</li><li>● On.</li></ul>
<b>Specific parameter: Relay status</b>	<p>This parameter defines the relay status. the various states can be:</p> <ul style="list-style-type: none"><li>● Unchanged,</li><li>● Open,</li><li>● Closed.</li></ul>



**Application example**      The example given below makes use of the CONTROL\_LEDS function to control the T CCX 1720 W console's green LED status.  
This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

Application description	Variables used
Initial conditions <ul style="list-style-type: none"><li>writing the console address (ADR#0.04) in a word table,</li><li>adjustment of timeout to 50 s,</li><li>condition of execution,</li></ul> Application: The aim of this example is, on rising edge of the variables: <ul style="list-style-type: none"><li><b>Manu</b>: controlling the green LED in <b>Flashing</b> mode,</li><li><b>Auto</b>: controlling the green LED in <b>On</b> mode</li></ul>	%MW0:6: terminal address %KW200:x:Manu status data to send %KW210:x: Auto status data to send %MW100:4: report %MW100:X0: activity bit %MW102: timeout %M0:Manu variable %M2: Auto variable

Introduction to the console	Program corresponding to the application
<p>Green LED</p>  The diagram shows the front panel of the T CCX 1720 W console. It features a large rectangular display area at the top. Below the display is a control panel with a central diamond-shaped button with four arrows pointing up, down, left, and right. To the left of this button are four circular buttons arranged in a vertical column. To the right of the button are four circular buttons arranged in a horizontal row. There are also two rectangular buttons on the far left and right of the control panel. A line points from the text 'Green LED' to a small green LED indicator located on the left side of the console, above the vertical column of buttons.	<pre>(*INIT console address, condition, timeout *) IF %S1.3 THEN     %MW0:6:=ADR#0.0.4;     %MW102:=500;     %M100:2:=0; END_IF; (*State of green LED*) IF RE %M0 AND NOT %MW100:X0 THEN     CONTROL_LEDS(%MW0:6,%KW200:2,%MW100:4); END_IF; IF RE %M2 AND NOT %MW100:X0 THEN     CONTROL_LEDS(%MW0:6,%KW210:2,%MW100:4); END_IF; %M0:=%M0; %M2:=%M2;</pre>



Entry help screens corresponding to the application:

The image displays two screenshots of the **CONTROL\_LEDS** dialog box, which is used for configuring LED and relay status. Both screenshots show the same dialog box with the following fields and options:

- Parameters:**
  - Console address:
  - Report:
  - Data to Send:
- LED status:**
  - Red LED: ☒ Unchanged ☐ Off ☐ Flashing ☐ On
  - Yellow LED: ☒ Unchanged ☐ Off ☐ Flashing ☐ On
  - Green LED: ☐ Unchanged ☐ Off ☒ Flashing ☐ On
- Relay status:**
  - Relay icon: ☒ Unchanged ☐ Open

The left screenshot shows the 'Data to Send' field set to '%KW200' and the 'Relay status' field set to 'Unchanged'. The right screenshot shows the 'Data to Send' field set to '%KW210' and the 'Relay status' field set to 'Unchanged'. Both screenshots have 'OK' and 'Cancel' buttons at the bottom.



## ASSIGN\_KEYS function

---

**Role**

This function is used to configure all or some of the command keys of a CCX 17 console by linking them to internal bits from the communication master PLC. Using this function globally modifies the configuration of the affected command keys. The maximum number of keys that can be configured is twelve, however, only the keys present on the console receiving the command are taken into account. The ASSIGN\_KEYS function is available whether or not the CCX 17 has an application.

---

**Parameters Zone**

The principle of implementing the Parameters zone is elaborated in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

---

**Zone: Set by**

This zone is used to specify which out of the PLC or CCX 17 contains the key configuration data.

- PLC: the data considered is that entered in the **Command key** zones,
  - CCX 17 the data considered is that from the application contained in the console.
- 

**Zone: Command key x**

This zone is used to define the characteristics associated with each key. The table below shows the various possible characteristics.

Characteristics	Meaning
Inhibit	Invalidates the key status. Its confirmation inhibits the action and address / symbol fields.
Action	Defines the operating mode of the key. Selecting the mode on edge means that pushing in the key causes the associated bit to be set to 1, and releasing it causes it to be set to 0. Choosing counter output mode means that pushing in the key changes the bit status. The default value is on edge.
Address	Specifies the %Mi internal bit address associated with the key. If the symbol associated with this bit exists in the station database, it is automatically taken into account during address confirmation.
Symbol	Specifies the symbol associated with the bit. The address associated with this symbol is automatically taken into account.

---

**Zone: Display command keys**

Command keys are configured in groups of four. This zone is used to access the various groups of keys.

---

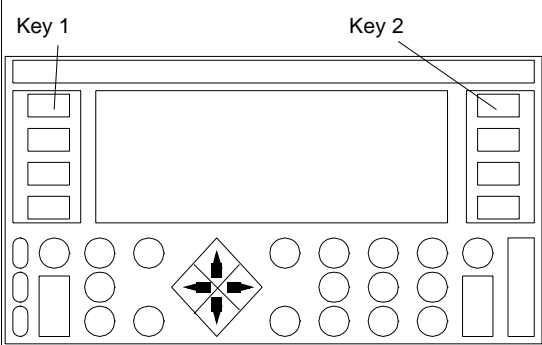


## Application example

The example given below makes use of the ASSIGN\_KEYS function to assign a function to the T CCX 1720 W console command keys1 and 2. This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

**Note:** For a CCX 17 with application, the assignment of command keys can be included here, making this PLC application unnecessary.

Application description	Variables used
<p>Initial conditions</p> <ul style="list-style-type: none"> <li>writing the console address (ADR#0.0.4) in a word table,</li> <li>adjustment of timeout to 50 s,</li> <li>condition of execution,</li> </ul> <p>Application: The aim of this example is:</p> <ul style="list-style-type: none"> <li>to assign the %M0 variable to command key 1,</li> <li>to assign the %M2 variable to command key 2,</li> <li>storing the function execution.</li> </ul>	<p>%MW0:6: terminal address          %KW240:x: data to be sent.          %MW100:4: report          %MW100:X0: activity bit          %MW102: timeout          %M0: Manu variable          %M2: Auto variable          %M12: condition of execution          %M108: condition of execution</p>

Introduction to the console	Program corresponding to the application
	<pre>( * INIT console address, condition, timeout * ) IF %S1.3 THEN     %MW0:6:=ADR#0.0.4;     %MW102:=500;     %M108:=0; END_IF; ( * Assign command keys * ) IF %M12 AND NOT %M108 AND NOT %MW100:X0 THEN     ASSIGN_KEYS(%MW0:6,%KW240:16,%MW100:4); END_IF;</pre>



Entry help screens corresponding to the application:

ASSIGN\_KEYS

Parameters

Console address:

%MW0

6

Report:

%MW100

4

Data to Send:

%KW240

16

Set by

☒ PLC

☐ CCX17

Command key 1

☐ Inhibit

Symbol:

Manual

Address:

%MV0

Action

☒ On edge

☐ On state

Command key 2

☐ Inhibit

Symbol:

Auto

Address:

%M2

Action

☒ On edge

☐ On state

Command key 3

☒ Inhibit

Symbol:

Address:

Action

☒ On edge

☐ On state

Command key 4

☒ Inhibit

Symbol:

Address:

Action

☒ On edge

☐ On state

Key 1 to 4...

Key 5 to 8...

Key 9 to 12...

Display command keys

OK

Cancel

328

TLXDS37PL7xx



## PANEL\_CMD function

---

**Role** This function is used to send various simple commands of the following type to the MMI console:

- clearing a line or the screen,
- printing or clearing the operator entry log,
- printing or clearing the alarms log,
- alarm management.

**Note:** The PANEL\_CMD function is available whether or not the CCX 17 has an application.

**Parameters field** The principle of implementing the Parameters field is elaborated in the Description of parameters (See Description of the parameters common to the different DOP functions, p. 287) section.

**Field: Command** This field is used to set the command associated with the PANEL\_CMD function.

For:

- clearing a line; the line number must be specified,
  - canceling an alarm; you must designate the alarm number which corresponds to identifier given while using SEND\_ALARM or DISPLAY\_ALARM.
-



**Application example**

The example below shows the implementation of the PANEL\_CMD function for the purpose of canceling a T CCX 1720 W console alarm.

This is connected to a TSX Premium via the AUX port (configured in UNI-TELWAY link, addresses 4 - 5).

Application description	Variables used
Initial conditions <ul style="list-style-type: none"> <li>● writing the console address (ADR#0.04) in a word table,</li> <li>● adjustment of timeout to 50 s,</li> <li>● condition of execution,</li> </ul> Application: The aim of this example is: <ul style="list-style-type: none"> <li>● wiping the console screen on user demand,</li> <li>● canceling the alarm on fault acknowledgement,</li> <li>● storing the function execution.</li> </ul>	%MW0:6: console address %KW350:x: Wipe data to be sent, %KW360:x: Cancel data to be sent, %MW100:4: report %MW100:X0: activity bit %MW102: timeout %M102: activation condition %M100: fault acknowledgement

**Program corresponding to the application**

```
(*INIT console address, condition, timeout *)
IF %S1.3 THEN
    %MW0:6:=ADR#0.0.4;
    %MW102:=500;
    %M120:2:=0;
END_IF;
(* Clear alarm ... *)
IF %M100 AND NOT %M120 AND NOT %MW100:X0 THEN
    PANEL_CMD(%MW0:6,%KW360:3,%MW100:4);
    SET %M120
END_IF;
* Clear screen *
IF %M102 AND NOT %M121 AND NOT %MW100:X0 THEN
    PANEL_CMD(%MW0:6,%KW350:3,%MW100:4);
    SET %M121;
END_IF;
```



Entry help screens corresponding to the application:

The image displays two overlapping 'PANEL-CMD' dialog boxes. The background dialog has the following settings:

- Parameters:**
  - Console address:   Report:
  - Data to be sent:
- Commands:**
  - Clear: ☒ Screen ☐ Line Line number
  - Entry log: ☐ Print ☐ Clear
  - Alarm log: ☐ Print ☐ Clear
  - Alarm management: ☐ Cancel an Alarm Alarm number

The foreground dialog has the following settings:

- Parameters:**
  - Console address:   Report:
  - Data to be sent:
- Commands:**
  - Clear: ☒ Screen ☐ Line Line number
  - Entry log: ☐ Print ☐ Clear
  - Alarm log: ☐ Print ☐ Clear
  - Alarm management: ☐ Cancel an Alarm Alarm number

Both dialogs feature 'OK' and 'Cancel' buttons at the bottom.



## ADJUST function

---

**Role**

This function is used to adjust (read and write) language objects (one object at a time) from a CCX 17 or a MAGELIS, by controlling internal words in the PLC memory. The language objects which can be adjusted are:

- internal bits (%Mi),
- internal words or double words (%MWi, %MDi),
- rack or remote inputs/outputs (%I, %Q, %IW, %QW, %ID, %QD).

**Note:** It is strongly advised:

- to only execute one instance of the ADJUST function per cycle,
- to only execute the ADJUST function every n cycles.
- to assign parameters to the ADJUST function with consecutive words, so as to optimize the reading of internal words on CCX 17 and MAGELIS.

**Activating the function (EN)**

This parameter is used to execute the ADJUST function.

The types of objects assigned to this parameter may be:

- an internal bit (%Mi),
- an internal word extract bit (%MWi:Xj).

**Read / Write (R\_W)**

This parameter defines the type of operation to be carried out:

- read: bit = 0,
- write: bit = 1.

The types of objects assigned to this parameter may be:

- an internal bit (%Mi),
- an internal word extract bit (%MWi:Xj).



**Type of object  
(TYPE)**

This parameter defines the type of object to be read or written.

The types of objects assigned to this parameter may be:

- an internal word (%MWi),
- an immediate value.

The table below shows the different types of objects which can be controlled using the ADJUST function.

Type of object	Value of internal word or immediate value	Type of object	Value of internal word or immediate value
%Mi	0	%IW	5
%MWi	1	%QW	6
%MDi	2	%ID	7
%I	3	%QD	8
%Q	4		



**Object address (ADR)** This parameter contains the address of the object to be read or written.  
The object type assigned to this parameter is a table of eight internal words (%MWi).  
The table below shows the content of the different words in the table.

Word order number	This word contains...	Possible values of the word
Word 0	the rack number where the language object concerned is situated	<b>0:</b> bit objects, internal words or double words, I/O objects whose operations function is registered in rack 0. <b>n:</b> other rack I/O objects.
Word 1	the number corresponding to the position in the rack of the I/O module or of the processor in which the language object concerned is situated	<b>0:</b> bit objects, internal words and double words, I/O objects whose operations function is associated to channels 1 and 2 of the processor registered at position 0 in the rack. <b>1:</b> I/O objects whose operations function is associated to channels 1 and 2 of the processor registered at position 1 the rack. <b>n:</b> other rack I/O objects.
Word 2	the channel number in the module where the language object concerned is situated	<b>0:</b> bit objects, internal words and double words. <b>1:</b> I/O objects whose operations function is associated to channel 1 of the processor (communication functions with PCMCIA). <b>2:</b> I/O objects whose operations function is associated to channel 2 of the processor (FIPIO link). <b>n:</b> other rack I/O objects.
Word 3	the rank of the I/O object or the number of the internal language object concerned.	<b>0 or n:</b> bit objects, internal words or double words, I/O objects with significant rank. <b>0:</b> Other I/O objects.
Word 4	the connection point number of the device on the FIPIO bus or the rank of the NANET object.	<b>n:</b> NANET or FIPIO objects. <b>0:</b> not significant.
Word 5	the position of the FIPIO module.	<b>0:</b> basic or not significant module. <b>1:</b> extension module
Word 6	the channel number in the FIPIO module or the slave bit on the AS-i bus.	<b>n:</b> AS-i or FIPIO objects. <b>0:</b> not significant.
Word 7	the slave number of the AS-i and NANET buses.	<b>n:</b> AS-i or NANET objects. <b>0:</b> not significant.



<b>Value to write (VAL)</b>	<p>This parameter contains the value to write in the object. The object type assigned to this parameter is a double word (%MDi).</p>
<b>Set at 1, or Incrementation (SINC)</b>	<p>Depending on the type of object to be written, this parameter is used:</p> <ul style="list-style-type: none"><li>• to set the bit value (%Mi, %Q),</li><li>• to increment by 1 the value of the word or double word (%MWi, %MDi, %QW, %QD).</li></ul> <div><b>Note:</b> the R_W parameter must be set to 1.</div> <p>The object type assigned to this parameter is an internal bit (%Mi).</p>
<b>Set to 0 or Decrementation (RDEC)</b>	<p>Depending on the type of object to be written, this parameter is used:</p> <ul style="list-style-type: none"><li>• to clear the bit value (%Mi, %Q),</li><li>• to decrement by 1 the value of the word or double word (%MWi, %MDi, %QW, %QD).</li></ul> <div><b>Note:</b> the R_W parameter must be set to 1.</div> <p>The object type assigned to this parameter is an internal bit (%Mi).</p>
<b>Value of the object read (VRET)</b>	<p>This parameter contains the parameter value which has just been read. The object type assigned to this parameter is a double word (%MDi).</p>
<b>Management parameters (GEST)</b>	<p>The object type assigned to this parameter is a table of 24 internal words (%MWi).</p>



**Examples**

The illustration below shows an example of ADJUST function entry.

Display the call

```
ADJUST( %MW20:X0.%MW20:X1.%MW21.%MW22:8.%MD30.%MW20:X2.%MW20:X3.%MD32.%MW34:24 )
```

To read the internal double word %MD12, the following values would need to be entered:

Parameter	Language object	Value to be entered :	Comment
EN	%MW20:X0	1	Executing the ADJUST function
R_W	%MW20:X1	0	Read operation
TYPE	%MW21	2	Type of object: %MD
ADR	%MW22	0	Not significant
	%MW23	0	Not significant
	%MW24	0	Not significant
	%MW25	12	Object number (%MD12)
	%MW26	0	Not significant
	%MW27	0	Not significant
	%MW28	0	Not significant
	%MW29	0	Not significant
VAL	%MD30	0	Not significant
SINC	%MW20:X2	0	Not significant
RDEC	%MW20:X3	0	Not significant
VRET	%MD32		Value of the object read
MAN	%MD34:24		Buffer parameter for receiving and sending requests

To write value 15 in the rack output word %QW3.2, the following values need to be entered:

Parameter	Language object	Value to be entered :	Comment
EN	%MW20:X0	1	Executing the ADJUST function
R_W	%MW20:X1	1	Write operation



Parameter	Language object	Value to be entered :	Comment
TYPE	%MW21	6	Type of object: %QW
ADR	%MW22	0	Rack number
	%MW23	3	Module position
	%MW24	2	Channel number
	%MW25	0	Not significant
	%MW26	0	Not significant
	%MW27	0	Not significant
	%MW28	0	Not significant
	%MW29	0	Not significant
VAL	%MD30	15	Value to write
SINC	%MW20:X2	0	Not significant
RDEC	%MW20:X3	0	Not significant
VRET	%MD32		Value of the object read
MAN	%MD34:24		Buffer parameter for receiving and sending requests

To increment the output word on FIPIO %QW1.2.12\0.1, the following values would need to be entered:

Parameter	Language object	Value to be entered :	Comment
EN	%MW20:X0	1	Executing the ADJUST function
R_W	%MW20:X1	1	Write operation
TYPE	%MW21	6	Type of object: %QW
ADR	%MW22	0	Not significant
	%MW23	1	Processor address
	%MW24	2	Channel number of built-in FIPIO link
	%MW25	0	Not significant
	%MW26	12	Connection point number
	%MW27	0	Module number: Standard
	%MW28	1	Channel number
	%MW29	0	Not significant



Parameter	Language object	Value to be entered :	Comment
VAL	%MD30	0	Not significant
SINC	%MW20:X2	0	Incrementing the word value by 1
RDEC	%MW20:X3	0	Not significant
VRET	%MD32		Not significant
MAN	%MD34:24		Buffer parameter for receiving and sending requests

To clear the output bit on the AS-i %Q\105.0\7.2 bus, the following values need to be entered:

Parameter	Language object	Value to be entered :	Comment
EN	%MW20:X0	1	Executing the ADJUST function
R_W	%MW20:X1	1	Write operation
TYPE	%MW21	4	Type of object: %Q
ADR	%MW22	1	Rack number
	%MW23	5	Module position
	%MW24	0	Channel number
	%MW25	0	Not significant
	%MW26	0	Not significant
	%MW27	0	Not significant
	%MW28	2	Bit rank (slave input/output)
	%MW29	7	Slave number
VAL	%MD30	0	Not significant
SINC	%MW20:X2	0	Not significant
RDEC	%MW20:X3	1	Output RESET
VRET	%MD32		Not significant
MAN	%MD34:24		Buffer parameter for receiving and sending requests



To decrement the output word on NANET %QW4.0\2.1, the following values would need to be entered:

Parameter	Language object	Value to be entered :	Comment
EN	%MW20:X0	1	Executing the ADJUST function
R_W	%MW20:X1	1	Write operation
TYPE	%MW21	6	Type of object: %QW
ADR	%MW22	0	Rack number
	%MW23	4	Module position
	%MW24	0	Channel number
	%MW25	1	NANET object rank (word number)
	%MW26	0	Not significant
	%MW27	0	Not significant
	%MW28	0	Not significant
	%MW29	2	Slave number
VAL	%MD30	0	Not significant
SINC	%MW20:X2	0	Not significant
RDEC	%MW20:X3	1	Decrementing the word value by 1
VRET	%MD32		Not significant
MAN	%MD34:24		Buffer parameter for receiving and sending requests







Introduction

Subject of this chapter

This chapter introduces the supplementary information for installing DOP functions.

What's in this Chapter?

This Chapter contains the following Sections:

Section	Topic	Page
28.1	Precautions for DOP use	343
28.2	Description of the built-in DOP functions "Data to send" parameter coding	344







## 28.1 Precautions for DOP use

### Precautions for DOP use

#### Introduction

The list below is not exhaustive, however, it does group together the errors most frequently encountered whilst installing DOP functions in an application.

#### List of precautions

- Do not forget to initialize the Time-out parameter (%MWi+2) before starting the Operator Dialogue function. For the ASK\_MSG or ASK\_VALUE functions, the value has to be 0.
- If you use the same word to save each function's report, you must then test the activity bit (%MWi:X0) at value 0 before starting another function.
- Synchronize the start of the different Built-in DOP functions in order that the CCX 17 console command queue does not become full.
- Remember to correctly calculate the size of the %KWi internal constants for displaying the data to be sent.
- Do not hesitate to allow a margin for error when allocating the data to be sent (%KWi:n). In fact, if you need to modify the text name, there will be no overlap between the different %KWi:n references.
- The maximum size for the Built-in DOP functions is 47 words.
- Warning, it is impossible to create a Built-in DOP function in line modification mode, if the application located in the PLC does not already contain a copy of this function.
- During power outage or communication loss, the application has to reset the CCX 17 in a coherent state (assigning buttons and messages to the screen).
- Canceling modifications or deleting a rung or phrase (List or Text) does not cancel initialization of the %KWi variables.



28.2

Description of the built-in DOP functions "Data to send" parameter coding

Introduction

Subject of this section

This section describes the **Data to send** parameter.  
It is intended for application developers who want to install built-in DOP functions without using the input help screens.

What's in this Section?

This Section contains the following Maps:

Topic	Page
PLC status message display: SEND_MSG function	345
PLC checked status message entry: ASK_MSG and GET_MSG function	348
PLC alarm message display: SEND_ALARM function	353
Display of status, alarm or a group of messages contained in the CCX 17 memory: ASK_VALUE, DISPLAY_MSG, GET_VALUE, DISPLAY_ALARM and DISPLAY_GRP functions	357
Display of luminous column LEDS: CONTROL_LEDS function	358
Configuring command keys: ASSIGN_KEYS function	359
Generic send command: PANEL_CMD function	361



## PLC status message display: SEND\_MSG function

### Introduction

Status messages can be constructed from the PLC application by creating the send using internal words (%Mwi) as intermediaries to display them on a CCX17 console screen. This is the role of the SEND\_MSG function.

### Data to send parameter coding

The table below shows the significance of the different words which constitute the following parameter: **Data to send** (47 words maximum).

Word number	Meaning	
1	contains a marker of value 16#CC17,	
2	contains value 0	
3	contains the length in bytes and the zone of the following words,	
4 to P	contain the message text to send, this being the characters that are <b>underlined</b> representing the characters expected during the display of a variable. The maximum length of this text is 40 characters. If the text is composed of an odd number of characters, the last byte is worth 0, if the text is of even length and if it is less than 39 characters, the last byte must contain the value 0,	
P +1	contains the line number where the message must be displayed,	
P +2	contains the column number where the start of the message must be displayed,	
P +3	This two word zone (four characters) contains the message characteristics and is structured as follows:	
P +4		
	Character 1	This character (in capitals) corresponds to a video attribute: <ul style="list-style-type: none"> <li>● B = blinking,</li> <li>● R = reverse video,</li> <li>● A = blinking and reverse video,</li> <li>● N = no attribute</li> </ul>
	Character 2	This character (in capitals) corresponds to the character font size: <ul style="list-style-type: none"> <li>● S = normal size,</li> <li>● D = double width or height</li> </ul>
	Characters 3 and 4	corresponds to the printing option: <ul style="list-style-type: none"> <li>● Y followed by a space = yes,</li> <li>● N followed by a space = no.</li> </ul>
P+5	<b>If you do not want to display a variable, the following word must be at 0 (in this instance, following parameters is ignored), otherwise the following parameters must be added:</b>	
P+6	contains the position of the variable to display, counted in number of characters in relation to the start of the message.	
P+7	contains the number of characters to display for the variable.	
P	<b>Number of words containing the text of the message to send (see line 4 to P).</b>	



Word number	Meaning			
P+8	contains a supplementary command: <ul style="list-style-type: none"><li>● 0: no command,</li><li>● 1: clear screen,</li><li>● 2: clear the line before the display.</li></ul>			
P+9	contains the value 16#0030.			
P+10 and 11	contain the type of entry field: <ul style="list-style-type: none"><li>● BIT +space = bit type,</li><li>● ANA + space = word type,</li><li>● LNG + space = double word type,</li><li>● DAY + space = date type,</li><li>● HOU + space = hour type.</li></ul>			
P+12	contains value 0.			
P+13	contains the type of variable to display: <ul style="list-style-type: none"><li>● B +space = bit type,</li><li>● W + space = word type,</li><li>● DW = double word type,</li></ul> <b>Note:</b> for a Date or Hour type, this word contains value 0.			
P+14	contains: <ul style="list-style-type: none"><li>● the address index of the variable to display for a bit, word or double word type,</li><li>● value –1 for a date or hour type.</li></ul>			
P+15 and 16	contain the display format for the variable and are structured in the following ways:			
	Byte 1	specifies if the variable is signed (sign +), or not (space),		
	Bytes 2 and 3	specifies the display format:		
			Byte 2	Byte 2
		ASCII or DIGITAL without decimal places	the ASCII code for the space, that being 20	
		DIGITAL with less than ten decimal places	the ASCII code for the space, that being 20	the ASCII code for the number of decimal places
		DIGITAL with at least ten decimal places	the ASCII code for the number of tens of decimal places	the ASCII code for the number of decimal place units
	Byte 4	specifies the display type: <ul style="list-style-type: none"><li>● N = no format,</li><li>● D = Digital,</li><li>● A = ASCII</li></ul>		
P	Number of words containing the text of the message to send (see line 4 to P).			



Word number	Meaning
P+17	define if the variable must be refreshed or not: <ul style="list-style-type: none"> <li>● Y + space = yes,</li> <li>● N + space = no.</li> </ul>
P+18	contains the value N + space,
P+19 to 25	contain the value 0 (7 words).
<b>P</b>	<b>Number of words containing the text of the message to send (see line 4 to P).</b>

### Example of use

The example below shows the values corresponding to the word table %MW0:12 (data to send) of the SEND\_MSG(ADR#0.0.4,%MW0:12,%MW100:4) function;

Word number	Value	Comment
%MW0	<b>16#CC17</b>	Marker
%MW1	<b>0</b>	Imposed value
%MW2	<b>18</b>	Byte size of the following zone which includes nine words
%MW3	<b>Fo</b>	Message text
%MW4	<b>ur</b>	Message text (continued)
%MW5	<b>sp4</b>	Message text (end)
%MW6	<b>0</b>	End of message mark
%MW7	<b>2</b>	Message position (line number)
%MW8	<b>15</b>	Message position (column number)
%MW9	<b>ND</b>	Message characteristics (no attribute or double font)
%MW10	<b>Nsp</b>	Message characteristic (no printing)
%MW11	<b>0</b>	The message includes no variable



## PLC checked status message entry: ASK\_MSG and GET\_MSG function

---

### Introduction

Checked status messages can be constructed from the PLC application by creating the send using internal words (%MWi) as intermediaries to display them on a CCX17 console screen. This is the role of the ASK\_MSG and GET\_MSG functions.

---



# Data to send parameter coding

The table below shows the significance of the different words which constitute the following parameter: **Data to send** (47 words maximum).

Word number	Meaning
1	contains a marker of value 16#CC17,
2	contains the command type: <ul style="list-style-type: none"> <li>● 33 = command number for ASK_MSG,</li> <li>● 6 = command number for GET_MSG,</li> </ul>
3	contains the length in bytes and the zone of the following words,
4 to P	contain the message text to send, this being the characters that are <b>underlined</b> representing the characters expected during the display of a variable. The maximum length of this text is 40 characters. If the text is composed of an odd number of characters, the last byte is worth 0, if the text is of even length and if it is less than 39 characters, the last byte must contain the value 0,
P +1	contains the line number where the message must be displayed,
P +2	contains the column number where the start of the message must be displayed,
P +3	This two word zone (four characters) contains the message characteristics and is structured as follows:
P +4	
	Character 1      This character (in capitals) corresponds to a video attribute: <ul style="list-style-type: none"> <li>● B = blinking,</li> <li>● R = reverse video,</li> <li>● A = blinking and reverse video,</li> <li>● N = no attribute</li> </ul>
	Character 2      This character (in capitals) corresponds to the character font size: <ul style="list-style-type: none"> <li>● S = normal size,</li> <li>● D = double width or height</li> </ul>
	Characters 3 and 4      corresponds to the printing option: <ul style="list-style-type: none"> <li>● Y followed by a space = yes,</li> <li>● N followed by a space = no.</li> </ul>
P+5	contains the position of the variable to display, counted in number of characters in relation to the start of the message.
P+6	contains the number of characters to display for the variable.
P+7	contains a supplementary command: <ul style="list-style-type: none"> <li>● 0: no command (ASK_MSG synchronized entry),</li> <li>● 24: free entry authorized after display (GET_MSG multiple entry).</li> </ul>
P+8	contains the value 16#0030.
P+9 P+10	contain the type of entry field (in Upper case): <ul style="list-style-type: none"> <li>● BIT +space = bit type,</li> <li>● ANA + space = word type,</li> <li>● LNG + space = double word type.</li> </ul>
P	<b>Number of words containing the text of the message to send (see line 4 to P).</b>



Word number	Meaning			
P+11	contains value 0.			
P+12	contains the type of variable to display (in Upper case): <ul style="list-style-type: none"><li>● B +space = bit type,</li><li>● W + space = word type,</li><li>● DW = double word type,</li></ul>			
P+13	contains the address index of the variable to display.			
P+14	contain the display format for the variable and are structured in the following ways:			
P+15	Byte 1	specifies if the variable is signed (sign +), or not (space),		
	Bytes 2 and 3	specifies the display format:		
			Byte 2	Byte 3
		ASCII or DIGITAL without decimal places	the ASCII code for the space, that being 20	
		DIGITAL with less than ten decimal places	the ASCII code for the space, that being 20	the ASCII code for the number of decimal places
		DIGITAL with at least ten decimal places	the ASCII code for the number of tens of decimal places	the ASCII code for the number of decimal place units
Byte 4	specifies the display type: <ul style="list-style-type: none"><li>● N = no format,</li><li>● D = Digital,</li><li>● A = ASCII</li></ul>			
P+16	defines whether or not the variable must be updated: <ul style="list-style-type: none"><li>● Y + space = yes,</li><li>● N + space = no.</li></ul>			
P+17	defines the field attributes: <ul style="list-style-type: none"><li>● I + space = increment,</li><li>● L + space = others.</li></ul>			
P+18 and 19	define the limit type: <ul style="list-style-type: none"><li>● 0 = unlimited,</li><li>● 1 = minimum limit only,</li><li>● 2 = maximum limit only,</li><li>● 3 = minimum and maximum limit.</li></ul>			
P+20 and 21	contain the minimum limit value.			
P+22 and 23	contain the maximum limit value.			
P+24 and 25	contain the increment value.			
P	Number of words containing the text of the message to send (see line 4 to P).			



## Example of use

The example below shows the values corresponding to the word table %MW0:38 (data to send) of the GET\_MSG(ADR#0.0.4,%MW0:38,%MW100:4) function;

Word number	Value	Comment
%MW0	<b>16#CC17</b>	Marker
%MW1	<b>6</b>	Command number for GET_MSG
%MW2	<b>70</b>	Byte size of the following zone which includes 35 word
%MW3	<b>Te</b>	Message text
%MW4	<b>mp</b>	Message text (continued)
%MW5	<b>er</b>	Message text (continued)
%MW6	<b>at</b>	Message text (continued)
%MW7	<b>ur</b>	Message text (continued)
%MW8	<b>eesp</b>	Message text (continued)
%MW9	<b>=sp</b>	Message text (continued)
%MW10	<b>—</b>	Message text (continued)
%MW11	<b>_sp</b>	Message text (continued)
%MW12	<b>°C</b>	Message text (end)
%MW13	<b>0</b>	End of message mark
%MW14	<b>3</b>	Text position (line number)
%MW15	<b>11</b>	Text position (column number)
%MW16	<b>NS</b>	Message characteristics (no attribute or double font)
%MW17	<b>Nsp</b>	Message characteristic (no printing)
%MW18	<b>15</b>	Position of the variable from the start of the message
%MW19	<b>3</b>	Number of characters to display
%MW20	<b>24</b>	Additional command (entered after display)
%MW21	<b>16#0030</b>	Reserved value
%MW22	<b>AN</b>	Type of entry field (AN = start of ANA)
%MW23	<b>Asp</b>	Type of entry field (continued)
%MW24	<b>0</b>	Reserved value
%MW25	<b>Wsp</b>	Type of variable to display (W = word type variable)
%MW26	<b>10</b>	Address index for the variable to display (%MW10)
%MW27	<b>espesp</b>	Display format (sp for a non signed variable, sp for the start of the decoding of decimal places after the point)
%MW28	<b>spD</b>	Display format continued (sp for the end of the coding of the number of digits after the point, D for the decimal format)
%MW29	<b>Yesp</b>	The variable must be updated



Word number	Value	Comment
%MW30	lesp	Entry is incremental
%MW31	0	The variable is not limited
%MD32	0	Minimum limit value
%MD34	0	Maximum limit value
%MD36	50	Increment value

---



## PLC alarm message display: SEND\_ALARM function

---

### Introduction

Alarm messages can be constructed from the PLC application by creating the send using internal words (%MWi) as intermediaries to display them on a CCX17 console screen. This is the role of the SEND\_ALARM function.

---



# **Data to send parameter coding**

The table below shows the significance of the different words which constitute the following parameter: **Data to send** (37 words maximum).

Word number	Meaning						
1	contains a marker of value 16#CC17,						
2	contains value 0						
3	contains the length in bytes and the zone of the following words,						
4	contains the imaginary number given to the alarm message (this word is used after deactivating the alarm if necessary). The value of this word must be between 900 and 999.						
5 to P	contain the message text to send, this being the characters that are <b>underlined</b> representing the characters expected during the display of a variable. The maximum length of this text is 40 characters. If the text is composed of an odd number of characters, the last byte is worth 0, if the text is of even length and if it is less than 39 characters, the last byte must contain the value 0,						
P +1 and 2	This two word zone (four characters) contains the message characteristics and is structured as follows: <table> <tr> <td>Character 1</td><td>This character (in capitals) corresponds to the character font size: <ul style="list-style-type: none"> <li>● S = normal size,</li> <li>● D = double width or height.</li> </ul> </td></tr> <tr> <td>Character 2</td><td>corresponds to the printing option: <ul style="list-style-type: none"> <li>● Y = yes,</li> <li>● N = no.</li> </ul> </td></tr> <tr> <td>Characters 3 and 4</td><td>correspond to the overprinting option: <ul style="list-style-type: none"> <li>● Y followed by a space = yes,</li> <li>● N followed by a space = no.</li> </ul> </td></tr> </table>	Character 1	This character (in capitals) corresponds to the character font size: <ul style="list-style-type: none"> <li>● S = normal size,</li> <li>● D = double width or height.</li> </ul>	Character 2	corresponds to the printing option: <ul style="list-style-type: none"> <li>● Y = yes,</li> <li>● N = no.</li> </ul>	Characters 3 and 4	correspond to the overprinting option: <ul style="list-style-type: none"> <li>● Y followed by a space = yes,</li> <li>● N followed by a space = no.</li> </ul>
Character 1	This character (in capitals) corresponds to the character font size: <ul style="list-style-type: none"> <li>● S = normal size,</li> <li>● D = double width or height.</li> </ul>						
Character 2	corresponds to the printing option: <ul style="list-style-type: none"> <li>● Y = yes,</li> <li>● N = no.</li> </ul>						
Characters 3 and 4	correspond to the overprinting option: <ul style="list-style-type: none"> <li>● Y followed by a space = yes,</li> <li>● N followed by a space = no.</li> </ul>						
P+3	<b>If you do not want to display a variable, the following word must be at 0 (in this instance, following parameters is ignored), otherwise the following parameters must be added:</b>						
P+4	contains the position of the variable to display, counted in number of characters in relation to the start of the message.						
P+5	contains the number of characters to display for the variable.						
P+6	contains the value 16#0030.						
P+7 and 8	contain the type of entry field: <ul style="list-style-type: none"> <li>● BIT +space = bit type,</li> <li>● ANA + space = word type,</li> <li>● LNG + space = double word type,</li> </ul>						
P+9	contains value 0.						
P+10	contains the type of variable to display: <ul style="list-style-type: none"> <li>● B +space = bit type,</li> <li>● W + space = word type,</li> <li>● DW = double word type,</li> </ul>						
<b>P</b>	<b>Number of words containing the text of the message to send (see line 5 to P).</b>						



Word number	Meaning			
P+11	contains the address index of the variable to display.			
P+12and 13	contain the display format for the variable and are structured in the following ways:			
	Byte 1	specifies if the variable is signed (sign +), or not (space),		
	Bytes 2 and 3	specifies the display format:		
			Byte 2	Byte 3
		ASCII or DIGITAL without decimal places	the ASCII code for the space, that being 20	
		DIGITAL with less than ten decimal places	the ASCII code for the space, that being 20	the ASCII code for the number of decimal places
		DIGITAL with at least ten decimal places	the ASCII code for the number of tens of decimal places	the ASCII code for the number of decimal place units
Byte 4	specifies the display type: <ul style="list-style-type: none"><li>● N = no format,</li><li>● D = Digital,</li><li>● A = ASCII</li></ul>			
P	Number of words containing the text of the message to send (see line 5 to P).			



**Example of use** The example below shows the values corresponding to the word table %MW0:29 (data to send) of the SEND\_ALARM(ADR#0.0.4,%MW0:29,%MW100:4) function;

Word number	Value	Comment
%MW0	<b>16#CC17</b>	Marker
%MW1	<b>0</b>	Imposed value
%MW2	<b>52</b>	Byte size of the following zone which includes 26 word
%MW3	<b>900</b>	Alarm message number
%MW4	<b>Su</b>	Message text
%MW5	<b>rc</b>	Message text (continued)
%MW6	<b>ha</b>	Message text (continued)
%MW7	<b>uf</b>	Message text (continued)
%MW8	<b>fe</b>	Message text (continued)
%MW9	<b>spf</b>	Message text (continued)
%MW10	<b>or</b>	Message text (continued)
%MW11	<b>rsp</b>	Message text (continued)
%MW12	<b>=sp</b>	Message text (continued)
%MW13	<b>—</b>	Message text (continued)
%MW14	<b>_sp</b>	Message text (continued)
%MW15	<b>°C</b>	Message text (end)
%MW16	<b>0</b>	End of message mark
%MW17	<b>SY</b>	Message characteristic (Normal size, printing)
%MW18	<b>Ysp</b>	Message characteristic (overprinting)
%MW19	<b>19</b>	Position of the variable from the start of the message
%MW20	<b>3</b>	Number of characters to display
%MW21	<b>16#0030</b>	Reserved value
%MW22	<b>AN</b>	Type of entry field (AN = start of ANA)
%MW23	<b>Asp</b>	Type of entry field (continued)
%MW24	<b>0</b>	Reserved value
%MW25	<b>Wsp</b>	Type of variable to display (W = word type variable)
%MW26	<b>10</b>	Address index for the variable to display (%MW10)
%MW27	<b>espesp</b>	Display format (sp for a non signed variable, sp for the start of the decoding of decimal places after the point)
%MW28	<b>spD</b>	Display format continued (sp for the end of the coding of the number of digits after the point, D for the decimal format)



## Display of status, alarm or a group of messages contained in the CCX 17 memory: ASK\_VALUE, DISPLAY\_MSG, GET\_VALUE, DISPLAY\_ALARM and DISPLAY\_GRP functions

### Introduction

For these functions it is possible to send them by using internal words (%MWi) as intermediaries.

The **Data to send** parameter requires one word which contains accordingly:

- The status message number,
- Alarm message number,
- Message group number

### Example of use

The example below shows an example of a function using the word %MW0 (data to send).

DISPLAY\_GRP(ADR#0.0.4,%MW0,%MW100:4) with %MW0:=3



## Display of luminous column LEDS: CONTROL\_LEDS function

### Introduction

It is possible to set the relay status (version 2.1 and above) and the LEDs on the luminous screen of a CCX17 console, then to send them using internal words (%MWi) as intermediaries. This is the role of the CONTROL\_LEDS function.

### Data to send parameter coding

The table below shows the significance of the different words which constitute the following parameter: **Data to send** (2 words).

Word number	Meaning
1	contains a marker of value 16#CC17,
2	<p>indicates the coding of each LED as well as the relay state to send to the terminal.</p> <ul style="list-style-type: none"> <li>bits 0 to 3: green LED status,</li> <li>bits 4 to 7: yellow LED status,</li> <li>bits 8 to 11: red LED status,</li> <li>bits 12 to 15: relay status,</li> </ul> <p>the state of each of the LEDS is coded on four bits as follows:</p> <ul style="list-style-type: none"> <li>0000: unchanged LED status,</li> <li>0001: LED off,</li> <li>0010: LED on,</li> <li>1111: LED flashing.</li> </ul> <p>the relay status is coded on bits 12 to 15 as follows:</p> <p>the state of each of the indicators is coded on four bits as follows:</p> <ul style="list-style-type: none"> <li>0000: unchanged relay status,</li> <li>0001: open relay status,</li> <li>0010: closed relay status.</li> </ul>

### Example of use

The example below shows the values corresponding to the word table %MW0:2 (data to send) of the CONTROL\_1EDS(ADR#0.0.4,%MW0:2,%MW100:4) function;

Word number	Value	Comment
%MW0	16#CC17	Marker
%MW1	16#1112	Green LEDS on, yellow and red off, open relay status



## Configuring command keys: ASSIGN\_KEYS function

### Introduction

It is possible to set the configuration of the command keys (bit associated or not, key operating mode, assignment by CCX17...) and send it using internal words (%MWi) as intermediaries to display them on a CCX17 console screen. This is the role of the ASSIGN\_KEYS function.

### Data to send parameter coding

The table below shows the significance of the different words which constitute the following parameter: **Data to send** (12 words).

Word number	Meaning	
1	contains a marker of value 16#CC17,	
	If a PLC carries out the assignment...	If a CCX 17 carries out the assignment...
2	this word contains the list of keys to configure. Each command key is coded on a bit (0: non configured, 1: configured): <ul style="list-style-type: none"> <li>bit i (i = 1 to 8) command key no. 1.</li> </ul>	this word contains 16#F000
	the following two words indicate the operating mode of each of the command keys. Each key is coded by two bits <ul style="list-style-type: none"> <li>00: Reset,</li> <li>01: functioning mode on edge,</li> <li>10: toggle functioning mode,</li> <li>11: no action.</li> </ul> Bits 2j,2j+1 = command key i+1 (i = 0 to 11)	the values of the following ten words are indifferent and will be ignored by the CCX 17 console.
	the following eight words contain, accordingly: <ul style="list-style-type: none"> <li>a value -1 when the command keys are not affected,</li> <li>the index of the internal bits affected by the command keys</li> </ul> Word i = command key i (i = 1 to 8)	



**Example of use**      The example below shows the values corresponding to the word table %MW0:16 (data to send) of the ASSIGN\_KEYS (ADR#0.0.4,%MW0:16,%MW100:4) function;

Word number	Value	Comment
%MW0	<b>16#CC17</b>	Marker
%MW1	<b>16#000F</b>	Command keys 1 to 4 are configured, the others are not.
%MW2	<b>16#FF09</b>	Key 1 in edge mode (01), key 2 in toggle mode (02), keys 3 and 4 inhibited (00), the others not programmed (11)
%MW3	<b>16#00FF</b>	
%MW4	<b>10</b>	Key 1 affected by bit %M10
%MW5	<b>rc</b>	Key 1 affected by bit %M11
%MW6 to %MW15	<b>-1</b>	Keys 3 to 8 not affected

---



## Generic send command: PANEL\_CMD function

### Introduction

It is possible to manage different command types (clearing, printing) and to send them by using internal words (%MWi) as intermediaries. This is the role of the PANEL\_CMD function.

### Data to send parameter coding

The table below shows the significance of the different words which constitute the following parameter: **Data to send** (3 words maximum).

Word number	Meaning
1	contains a marker of value 16#CC17,
2	contains the command number: <ul style="list-style-type: none"> <li>● 1: clear screen,</li> <li>● 2: clear a line,</li> <li>● 9: print the alarm message log,</li> <li>● 10: clear the alarm message log,</li> <li>● 11: print the alarm log</li> <li>● 13: clear the alarm log,</li> <li>● 29: clear the alarm (from 1 to 300) destined for CCX 17,</li> <li>● 30: clear the alarm (from 900 to 999) destined for the PLC.</li> </ul>
3	indicates the command parameter. <ul style="list-style-type: none"> <li>● clear line = line number,</li> <li>● cancel alarm = alarm number,</li> <li>● other commands: no object.</li> </ul>

### Example of use

The example below shows the values corresponding to the word table %MW0:2 (data to send) for the PANEL\_CMD(ADR#0.0.4,%MW0:2,%MW100:4) function;

Word number	Value	Comment
%MW0	<b>16#CC17</b>	Marker
%MW1	<b>1</b>	Clear screen







---

## Process control functions

# VI

---

### At a Glance

#### Overview

This Part gives you information on the process control functions on Micro PLCs and describes how to install them using PL7 Micro, Junior and Pro software.

#### What's in this part?

This Part contains the following Chapters:

Chapter	Chaptername	Page
29	General on the PID	365
30	Description of the regulation functions	369
31	Operator dialogue on CCX 17	391
32	Example of application	403
33	Appendices	413







---

### Introduction

#### Subject of this chapter

This chapter introduces the **standard regulation functions** of the PL7 software.

#### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
General introduction	366
Principal of the regulation loop	367
Development methodology of a regulation application	368



## General introduction

---

### General

The regulation functions are the **standard elements** of the PL7 language. They support programming of the regulation loops of the Micro and Premium PLC's.

These functions are particularly adapted to:

- answering the needs of the sequential process which need the auxiliary adjustment functions (examples: Plastic film packaging machine, finishing treatment machine, presses...),
- respond to the needs of the simple adjustment process' (examples: metal furnaces, ceramic furnaces, small refrigerating groups...),
- respond to the automatic control features or to the mechanical regulation which has a critical sampling time (examples: torque regulation, speed regulation).

A pre-configured interface with the CCX\_17 range supports the driving and the adjustment of the regulation loops. In this area, up to 9 regulation loops are accessible via the CCX\_17.

**Note:** There is not a limit to the number of PID functions that are available in a function. In practice, it is the maximum number of input and output modules which are accepted by the PLC that limits the number of loops.

### Available functions

The standard regulation functions are split into two categories:

- a family of algorithm functions:
  - PID function to execute a mixed PID correction (serial – parallel),
  - PWM function to execute the modulation adjustment period on the discrete outputs,
  - SERVO function to execute the motor command adaptations,
- an operator dialogue function (PID\_MMI) which integrates a driving and adjustment application for the PID application on CCX\_17 version 2.

The PID\_MMI function is linked to 3 types of pre-configured screens.

---



## Principal of the regulation loop

### Introduction

The working of a regulation loop has three distinct phases:

- the acquisition of data and of:
  - measurements from the process' sensors (analog, encoders),
  - setpoint(s) generally from PLC internal variables or from data from the CCX\_17.
- execution of the PID regulation algorithm,
- the sending of orders adapted to the characteristics of the actuators to be driven via the discrete or analog outputs.

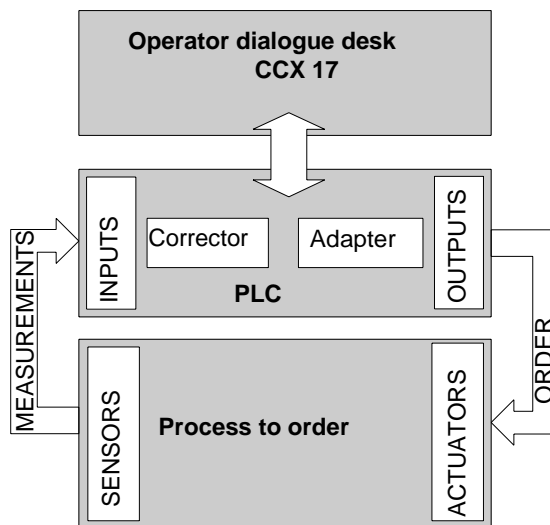
The PID algorithm generates the command signal from:

- the measurement sampled by the input module,
- the setpoint value fixed by either the operator or the program,
- the values of the different corrector parameters.

The signal from the corrector is either directly handled by an analog output card of the PLC linked to the actuator, or handled via the PWM or SERVO adjustments in function with the kinds of actuator to be driven on a PLC discrete output card.

### Illustration

The following diagram schematizes the principal of a regulation loop.



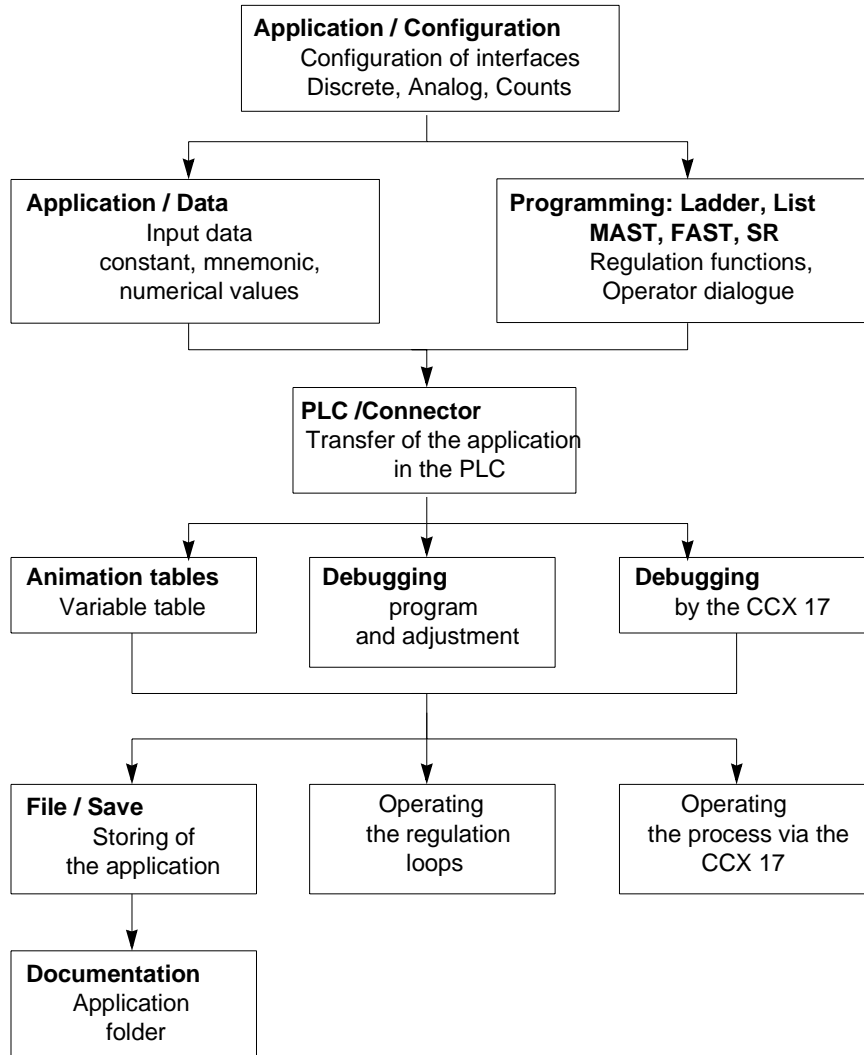


## Development methodology of a regulation application

### Diagram of the principal

The following diagram describes the linking of tasks to be carried out during the creation and debugging of a regulation application.

**Note:** The defined order is given for information only.





---

# Description of the regulation functions

30

---

## Introduction

### Subject of this chapter

This chapter describes the regulation functions.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Programming a regulation function	370
PID Function	371
Programming the PID function	373
PWM Function	378
Programming the PWM function	380
SERVO Function	382
Programming the SERVO function	386
Performance of the functions in the operating mode	389



## Programming a regulation function

---

### Programming rules

The regulation function parameters must all be informed. The functions use 3 kinds of parameters:

- read only parameters, considered at the beginning of the function's execution,
- write only parameters, positioned at the conclusion of the function's execution,
- the read and write only parameters, whose contents are considered at the beginning of the function's execution, are then updated by the results of the function.

**Note:** The regulation functions must be programmed in a **periodic** task (MAST periodic or FAST). They must not be conditioned.

### Parameterizing

The word type input parameters are the analog sizes expressed in the scale [0, +10000] and can be directly connected to measurement sensors via the words %IWxy.i des analog inputs.

The bit type output parameters support ordering discrete actuators and can be directly connected to the %Qxy.i. type variables.

In the same way, the word type output parameters support ordering of analog actuators on the scale [0, +10000] and can be directly appointed to the %QWxy.i. type variables.

The MWi:L word table type parameters regroup the user parameters and the data necessary to the internal working of the function.

**If the length of a table is sufficient, the function is not executed.**

**Note:** In order to keep the adjustment parameters of the OF adjustment on cold start, it is necessary to delete the %MWi reset to zero option (in the processor's configuration screen).



## PID Function

---

### General

The PID function completes a PID correction via an analog measurement and setpoint on the [0-10000] format and provides an analog command to the same format.

---

### Available functions

The PID OF is composed of the following functions:

- serial / parallel PID algorithm,
- forward / backward action (according to the KP gain sign),
- action derived from measurement or from distance,
- high and low limitation of the setpoint to [0-10000],
- high and low limitation of the output in automatic mode,
- anti-saturation of the integral action,
- Manual/Automatic operating modes without definitely changing,
- PID access control through the dialog operator,
- operating in integrator for (KP = TD =0).

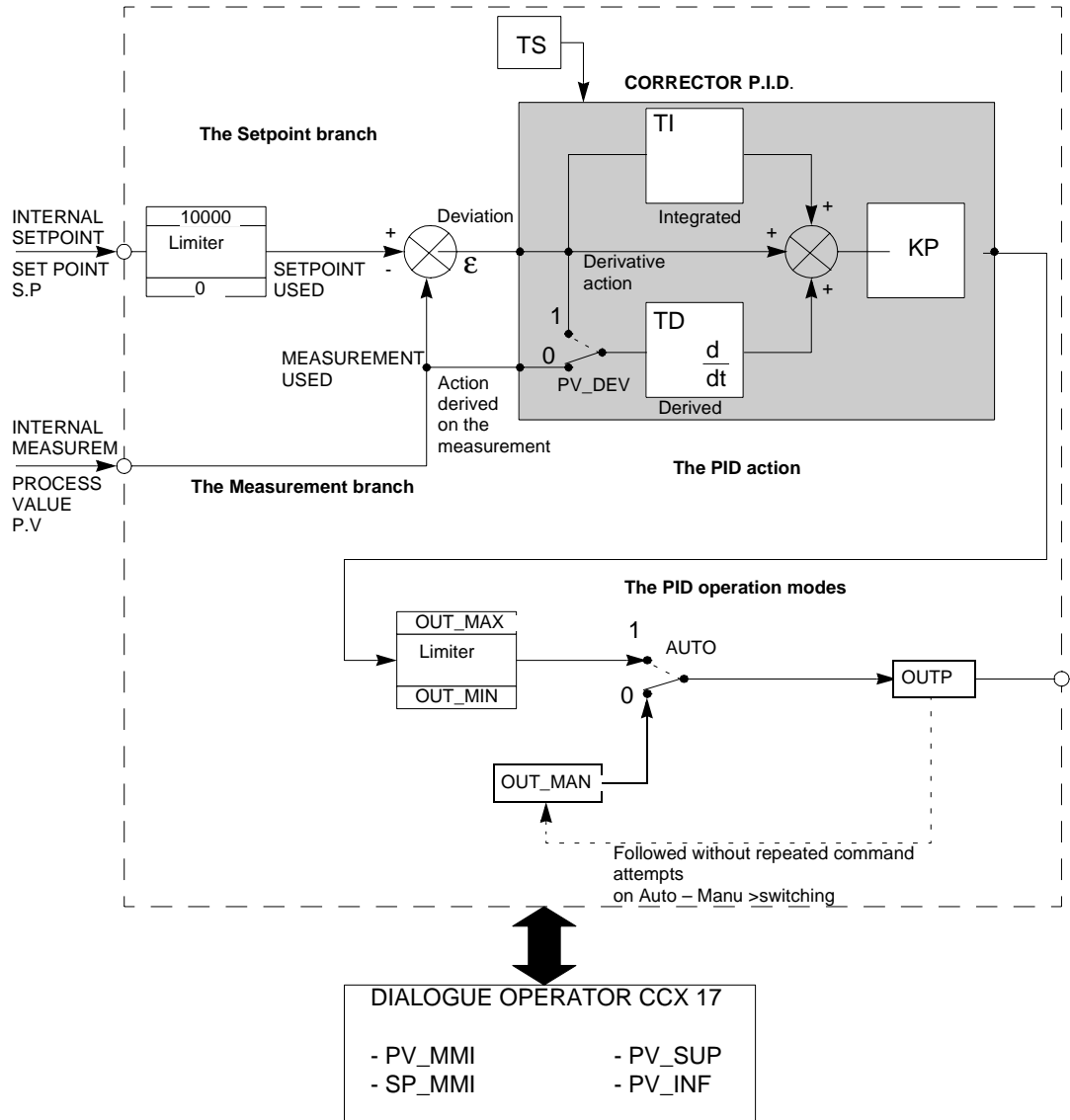
**Note:**

- The display parameters used by the CCX 17 are shown in physical units,
  - For a correct PID operation, you must stay within the scale of [0-1000] for the measurement and the setpoint.
-



## Operating principles

The following diagram presents the operating principle of the PID function.



**Note:** The description of the parameters used is presented in the (See Programming the PID function, p. 373) module.



# Programming the PID function

## Introduction

The PID functions are the standard functions of the PL7. As such, they are available from the functions library.

From the language editors there, it is possible to use the help of a PID function's input to facilitate its programming.

**Note:** The PID function's input can be done in any periodic task (MAST or FAST). The function does not have to be conditioned.

## Illustration

The illustration below gives you a general idea of the Functions screen in the library supporting the implementation of the PID function.

EF

DFB

Information Functions:

Parameters

Family	Lib.V	App.V.	Name	Comment
Orphee Function	2.10	-	PID	Mixed PID regulator
Stalling functions	2.00	-	PID_MMI	Management of operator dialogue dedicated to CCX17 from PID
GRAFCET	1.00	-	PWM	Modulated in the pulse width of a numerical size
Single precision reals	2.22	-	SERVO	PID output stage for open/close valve order
Process Control	2.01	2.01		
Integer tables	2.10	-		

Call format

Parameters of the FUNCTION:

Name	Type	Kind	Comment	Entry field
TAG	STRING	IN	PID label (8 charac), used by DOP on CCX17	"TEMP"
UNIT	STRING	IN	Measurement unit (6 charac), used by the DOP>>	"DEGRES"
PV	WORD	IN	Measurement, format [0; +10000]	%MW10
OUT	WORD	OUT	Output, format [0; +10000]	%MW11

Call display

PID ( "TEMP","DEGRES",%MW10,%MW11,TRIG\_PROD\_A,%MW20:43 )

## Syntax

The PID function's call syntax is:

**PID(TAG,UNIT,PV,OUT,AUTO,PARA)**



**Parameters of the PID function** The table below presents the different parameters of the PID function.

Parameter	Type	Kind IN = Input OUT = Output	Default value	Description
<b>TAG</b>	8 maximum characters or %MBi:L with L less than or equal to 8	IN	-	The used name of the PID used by the CCX 17.
<b>UNIT</b>	6 maximum characters] or %MBi:L with L less than or equal to 6	IN	-	The measurement unit of the PID used by the CCX 17.
<b>PV</b>	%Mwi or %IWxy.i.j	IN	-	Input indicating the <b>measurement</b> for the function.
<b>OUT</b>	%Mwi or %QWxy.i.j	OUT	0	Analog output of the PID. If TI = 0, an offset of 5000 is added to the OUT output in Auto mode.
<b>AUTO</b>	%Mi , %lxy.i or %Qxy.i	IN / OUT	0	Operating mode of the PID and the CCX 17. 0 : manual, 1 = Auto.
<b>PARA</b>	%MWi:43	IN / OUT	-	(See the table below for the breakdown of the PARA table).

The table below presents the different parameters of the PARA table:

Parameter	Position	Function
SP	%MWi	Proportional gain of the PID (x100), signed without unit (-10000<KP<+10000). The Kp sign determines the direction of the PID's action (negative: forward, positive: reverse)
OUT_MAN	%MW(i+1)	The PID's integral time (between 0 and 20000) is shown in 10 <sup>-1</sup> second
KP	%MW(i+2)	The PID's derivative time (between 0 and 10000) is shown in 10 <sup>-1</sup> second
TI	%MW(i+3)	The PID's sampling period (between 1 and 32000) is shown in 10 <sup>-2</sup> second. The real sampling period will be the multiple of the period of the task in which the PID closest to the TS is introduced
TD	%MW(i+4)	Upper limit of the PID's output in automatic. (between 0 and 10000)



Parameter	Position	Function
TS	%MW(i+5)	Lower limit of the PID's output in automatic. (between 0 and 10000)
OUT_MAX	%MW(i+6)	Proportional gain of the PID (x100), signed without unit (-10000<KP<+10000). The Kp sign determines the direction of the PID's action (negative: forward, positive: reverse)
OUT_MIN	%MW(i+7)	The PID's integral time (between 0 and 20000) is shown in 10 <sup>-1</sup> second
PV_DEV	%MW(i+8):X0	Derived action choice 0 = on the measurement, 1 = on the distance
NO_BUMP	%MW(i+8):X4	Mode with or without step by step. 0 = with step by step, 1 = without step by step
DEVAL_MMI	%MW(i+8):X8	= 1 : disables the acknowledgement of the PID by the dialog operator. = 0 : the PID is exploited by the dialog operator. This bit allows you to not do the conversion scales on the PIDs not exploited by the CCX_17, and to select the exploited PIDs, especially when there are more than 9 PIDs in the PL7 application.
PV_SUP (CCX 17)	%MW(i+9)	Upper limit of the measurement scale's range, in a physical unit (x100) (between -9 999 999 and + 9 999 999).
PV_INF (CCX 17)	%MD(i+11)	Lower limit of the measurement scale's range, in a physical unit (x100) (between -9 99 999 and + 9 999 999).
PV_MMI (CCX17)	%MD(i+13)	The measurement's image in a physical unit (x100)
SP_MMI (CCX 17)	%MD(i+15)	The setpoint operator and the image of the setpoint in a physical unit (x100)

**Note:**

- The other parameters that are used by the PID's internal management must never be modified by the application.
- The values used by the CCX 17 are multiplied by 100 in order to support a display with 2 figures after the comma on the CCX 17 (the CCX 17 does not exploit the floating point format but manages a fixed comma format).



**Rules**

There is no internal setpoint alignment on the measurement in manual mode.

The settings on the scale only take place on modification of one of the setpoints (SP or DOP\_SP).

The algorithm without the integral action ( $TI = 0$ ) carries out the following operation:

For	Then the output ...	With ...
$\varepsilon t = SP - PV$	$OUT = KP [\varepsilon t + Dt] / 100 + 5000$	$Dt = \text{derived action}$

The algorithm with the integral action ( $TI < 0$ ) carries out the following operation:

For	Then the output ...	With ...
$\varepsilon t = SP - PV$	$\Delta OUT = KP [\Delta \varepsilon t + (TS/10 \cdot TI) \cdot \varepsilon t + \Delta Dt] / 100$ $OUT = OUT + \Delta OUT$	$Dt = \text{derived action}$

On a cold start, the PID starts off again in manual, with the output at 0. To impose the automatic mode or a manual output that is not at 0 after a cold start, you will have to program the initialization sequence **after** the PID call.

---

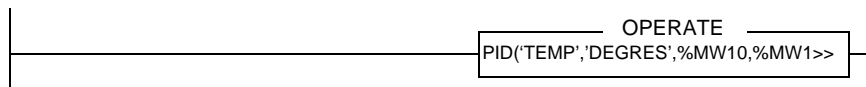


**Examples**

The examples offered below are done in Ladder language.

When the dialog operator regulation is used (DEVAL\_MMI = 0)

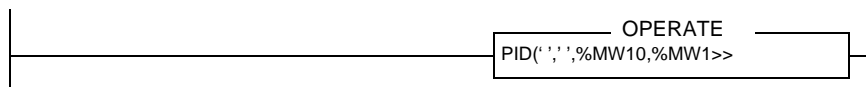
(\*PID correction of the temperature regulator loop \*)



with PID('TEMP','DEGRES',%MW10,%MW11,%M10,%MW20:43)

When there is no dialog operator DEVAL\_MMI = 1.

(\* PID correction of the regulation loop without built-in DOP



with PID(' ',' ',' ,%MW10,%MW11,%M10,%MW20:43)

**Note:** In this example, the TAG and UNIT parameters have no direction, so it is OK to only put the dimensions.



## PWM Function

---

### General

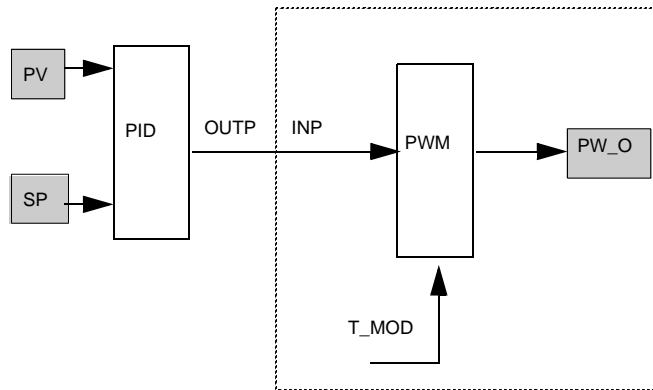
The PWM function supports regulation of a pulse width on a TOR output. It is a function that formats the PID's output.

The pulse width depends on the PID's output (The PWM function's INP input) and the modulation period.

---

### Operating principles

The circuit diagram of the function's operation is as follows:



**Note:** The description of the parameters used is presented in the (See Programming the PWM function, p. 380) module.

---

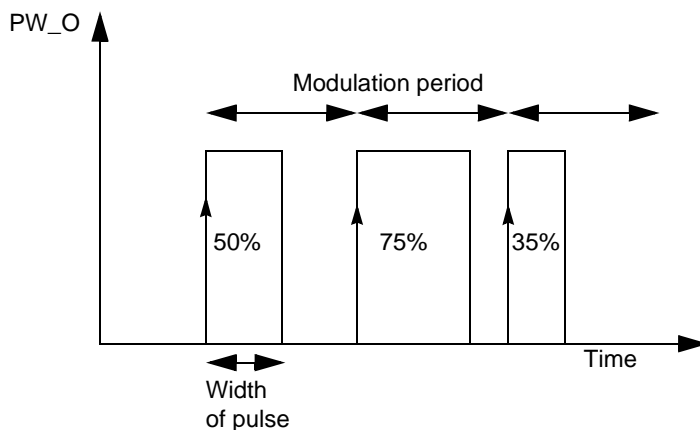


**Pulse widths**

To each TOP of the T\_MOD modulation period, the activation period in  $10^{-3}$  second of the PW\_O output is calculated according to the following formula:

State 1 of the gap (shown in  $10^{-2}$  seconds) =  $INP * T\_MOD / 1000$

The following timing diagram illustrates this formula:

**Practical rules**

$T\_MOD = TS$  (where TS is the sampling period of the upstream PID),

The period of the current task (expressed in  $10^{-3}$  second) is equal to:

**(Required resolution) \* 10 \* T\_MOD.**

The PID is in the MAST task, the MAST's period is  $50 * 10^{-3}$  s,  $TS = 500 * 10^{-2}$  s and the required resolution is 1/50 (a T\_MOD period must contain at least 50 periods of the current task).

$T\_MOD = TS = 500$ .

The period of the task where the PWM is introduced must therefore be less than  $500 * 10 / 50 = 100 * 10^{-3}$  s.

The PWM function can therefore be programmed in the MAST task.  
the resolution will be 1/100.



## Programming the PWM function

### Introduction

The PWM function is a standard PL7 function. As such, it is available from the functions library.  
From the language editors there, it is possible to use the help of a PWM function's input to facilitate its programming.

**Note:** The PWM function's input can be done in any periodic task (MAST or FAST).  
The function does not have to be conditioned.

### Illustration

The illustration below gives you a general idea of the Functions screen in the library supporting the implementation of the PWM function.

EF

DFB

Information Functions: Parameters

Family	Lib.V	App.V	Name	Comment
Orphee Function	2.10	-	PID	Mixed PID regulator
Stalling functions	2.00	-	PID_MMI	Management of operator dialogue dedicated to CCX17 from
GRAFCET	1.00	-	PWM	Pulse width modulation of a numerical size
Single precision reals	2.22	-	SERVO	PID output stage for open/close valve order
Process Control	2.01	2.01		
Integer tables	2.10	-		

Call format

Parameters of the FUNCTION:

Name	Type	Kind	Comment	Entry field
INP	WORD	IN	Numerical size to modulate	%MW11
PW_Q	EBOOL	OUT	Discrete output cyclic report equal to the LESS than value	%Q6.3
PARA	AR_W	IN/OUT	PWM parameters (5 word table)	%MW90:5

Call display

PWM ( %MW11,%Q6.3,%MW90:5 )

### Syntax

The PWM function's call syntax is:

**PWM(INP,PW\_0,PARA)**



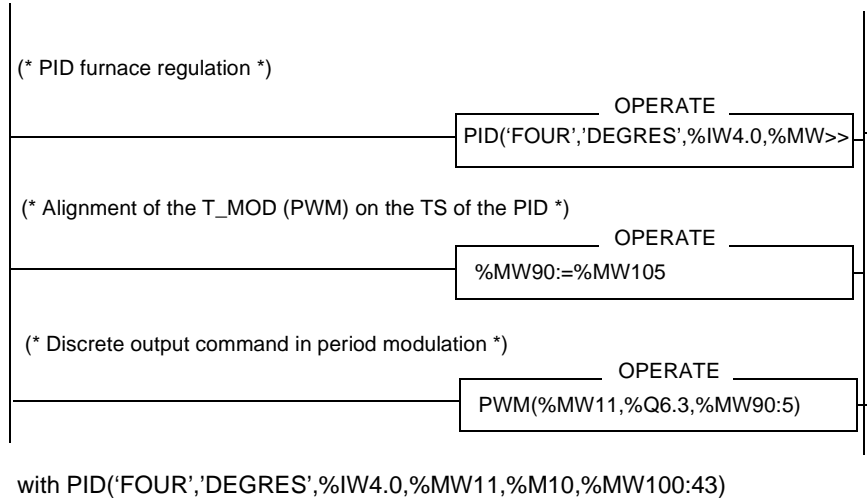
**Parameters of the PWM function**

The table below presents the different parameters of the PWM function.

Parameter	Type	Kind IN = Input OUT = Output	Description
INP	%MWi	IN	Analog value to be modulated in the pulse width (format [0 – 10000])
PW_0	%Qxy.i ou %Mi	OUT	Logic output (TOR) whose aspect ratio is the image of the INP input
PARAM	%MWi:5	IN / OUT	Modulation period shown in 1/100ths of seconds (between 0 and 32767). T_MOD must be more than or equal to the current task's period and it is adjusted by the system to be a whole multiple of this. Table of 5 words whose first word corresponds to the T_MOD parameter. The following are used internally by the function and must never be modified by the application

**Examples**

The example offered below is done in Ladder language.





## SERVO Function

### General

The SERVO function supports regulation with a motor type actuator steered by 2 alternating actions (UP and DOWN).

**Note:** It must be connected in tandem with the analog output of a PID. It cannot be used alone.

When there is a copy of a position, the valve's position is locked via the INP (setpoint) and POT (position measurement) inputs.

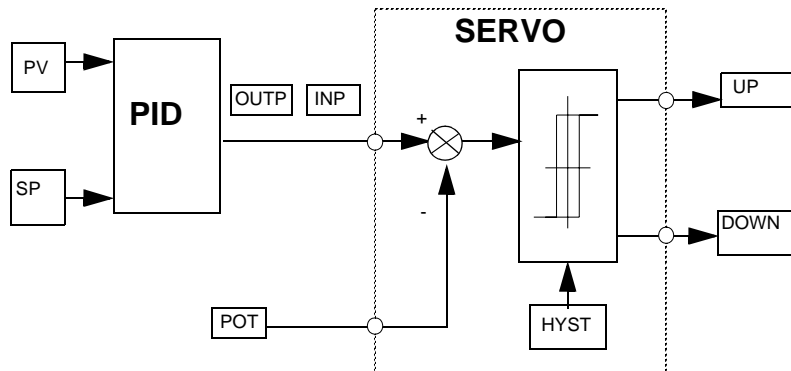
When the copy does not physically exist, the algorithm no longer uses the PID's absolute output but the output's variation. The UP output (or DOWN, according to the variation sign) is put at 1 for a length of time proportional to the actuator opening time and to the variation of the value. Also the notion of minimum pulse time is introduced.

### Principle of operating with a position copy

The SERVO function locks the motor's position according to a setpoint of the INP position from a PID's output on the [0-10000] format, and to a POT position measurement.

The locking algorithm is a relay with hysteresis.

In this case, the PID, T\_MOTOR and T\_MINI parameters are not used.



**Note:** The description of the parameters used is presented in the (See Programming the SERVO function, p. 386) module.



**Principle of  
operating  
without a  
position copy  
(POT= -10000)**

In this case the SERVO function synchronizes itself with the upstream PID via the bias of the PID parameters table, passed in parameter to the SERVO function.

The algorithm receives in input the PID's variation output and converts it into pulse period, according to the following formula:

$$T\_IMP \text{ (shown in } 10^{-3} \text{ s)} = OUT \times T\_MOTOR / 1000$$

The acquired period adds itself to the remaining period of the preceding cycles: in fact, what is not "consumed" in a cycle is memorized for the following cycles. This ensures a smooth operation especially with sudden variations of the (ex: PID setpoint level) command and in manual mode.

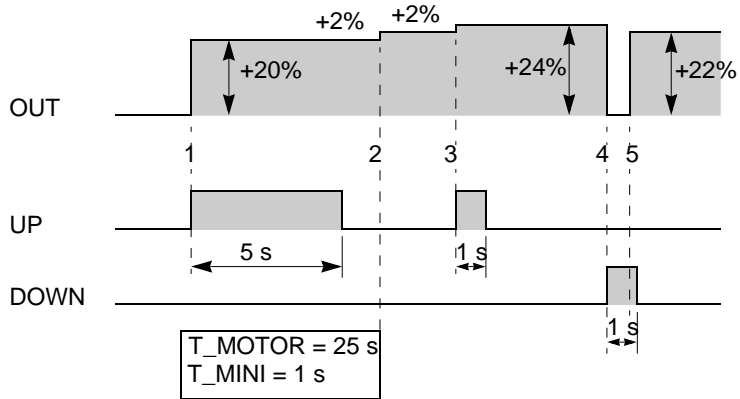
**Note:** The description of the parameters used is presented in the (See Programming the SERVO function, p. 386) module.

---



**Example**

The example offered below is done in Ladder language.

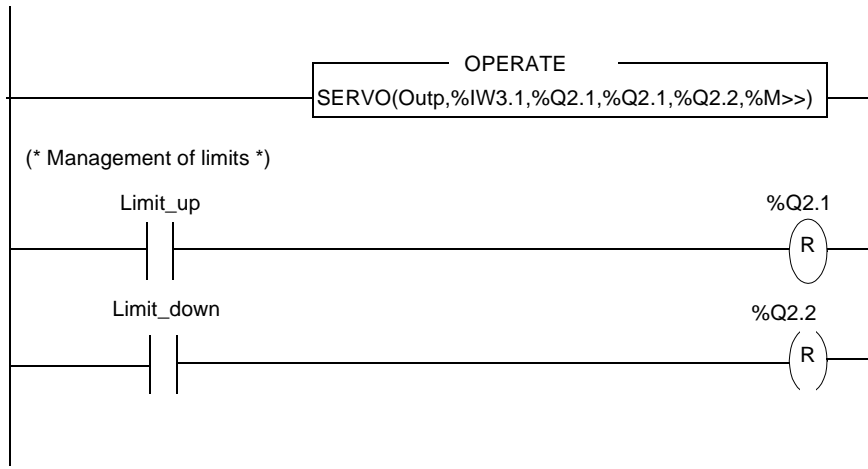
**Caption:**

1. The PID variation output is of +20% ( $T\_MOTOR$  pulse = 25 s for a 100% variation), in this case the pulse affects the UP output for a period of 5 s,
2. The PID variation is of +2%, which would correspond to a pulse of 0.5 s. This pulse is less than  $T\_MINI$  ( $=1\text{ s}$ ), and it does not affect the outputs,
3. A second variation of +2% appears and the function holds this variation concurrently with the preceding one (which corresponded to a variation less than the minimal value) for its calculation, which corresponds to a positive global variation of +4%, and therefore to a pulse of 1 s on the UP output,
4. A variation of -24% appears and the activated pulse is therefore of 6 s on the DOWN output,
5. Before the following second, another variation of +22% brings the system back to a global variation of 2% < to the variation of  $T\_MINI$  (4%). The function finishes carrying out the minimal pulse of 1 s.

**Note 1:** The SERVO function does not manage the position limits, these must be managed by the application. If a limit is detected, you must force the output corresponding to 0 (UP for the high limit, DOWN for the low limit).



**Example:** (done in Ladder language)



**Note 2:** Passing to the operating mode with copy to the mode without copy is possible (for example: when the copy has an error, go to mode without copy).



## Programming the SERVO function

### Introduction

The SERVO function is a standard PL7 function. As such, it is available from the functions library.  
From the language editors there, it is possible to use the help of a SERVO function's input to facilitate its programming.

**Note:** The SERVO function's input can be done in any periodic task (MAST or FAST). The function does not have to be conditioned.

### Illustration

The illustration below gives you a general idea of the Functions screen in the library supporting the implementation of the SERVO function.

EF

DFB

Information Functions: Parameters

Family	Lib.V	App.V	Name	Comment
Single precision reals	2.22	-	PID	Mixed PID regulator
Process Control	2.01	-	PID_MMI	Management of operator dialogue dedicated to CCX17 from PID
Integer tables	2.00	-	PWM	Modulated in the pulse width of a numerical size
Bit tables	2.00	-	SERVO	PID output stage for open/close valve order
Integer tables	2.10	-		
Integer tables	2.00	-		

Call format

Parameters of the PROCEDURE:

Name	Type	Kind	Comment	Entry field
INP	WORD	IN	Position setpoint, format [0;10000] (to conn>>	OUTP
POT	WORD	IN	Position copy, format [0;10000] [-10000>>	-10000
UP	EBOOL	OUT	Alternating output, UP operating direction	%Q2.1
DOWN	EBOOL	OUT	Alternating output, DOWN operating direction	%MW100:43

Call display

SERVO ( OUTP,-10000,%Q2.1,%MW100:43,%MW180:10 )

### Syntax

The SERVO function's call syntax is:

**SERVO(INP,POT,UP,DOWN,PID,PARA)**



### Parameters of the SERVO function

The table below presents the different parameters of the SERVO function.

Parameter	Type	Kind IN = Input OUT = Output	Description
INP	%MWi	IN	Position setpoint ([0 – 10000] format) that has to be connected to the PID output.
POT	%MWi or direct	IN	Position copy, ([0 - 10000] format) 0 : closed valve; 10000: open valve. If the copy does not exist. POT must be initialized at –10000. This particular value indicates "no copy".
UP	%Qxy.i or %Mi	OUT	Output signal for the motor's UP operating direction.
DOWN	bit type %Q or %M	OUT	Output signal for the motor's DOWN operating direction.
PID	%MWi:43	IN / OUT	The PARA parameter table of the upstream PID. Used if there are no copy words for the synchronization with the upstream PID. See Parameters of the PID function, p. 374.
PARA	%MWi:10	IN / OUT	(See the table below for the breakdown of the PARA table).

The table below presents the different parameters of the PARA table:

Parameter	Position	Function
T_MOTOR	%MWi	Valve opening time shown in $10^{-2}$ s. Used if the copy does not exist (POT = -10000).
T_MINI	%MW(i+1)	Minimal pulse time shown in $10^{-2}$ s. Used if the copy does not exist (POT = -10000).
HYST	%MW(i+2)	Value of the hysteresis on the [0 – 10000] format. Used if the copy does not exist (POT: [0 - 10000]).

#### Note:

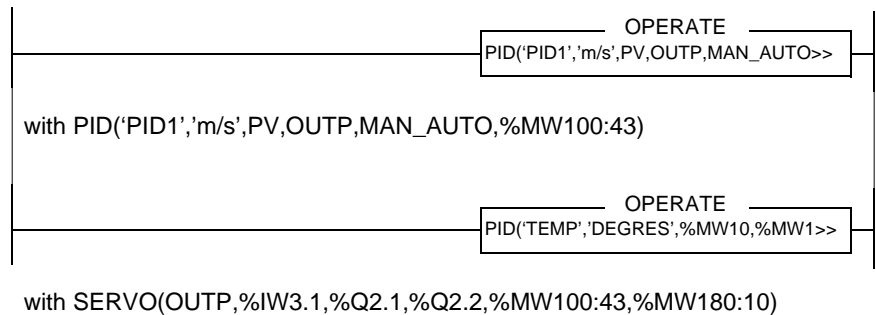
- The other parameters that are used by the function's internal management must never be modified by the application.
- All the parameters are obligatory, regardless of the operating mode.



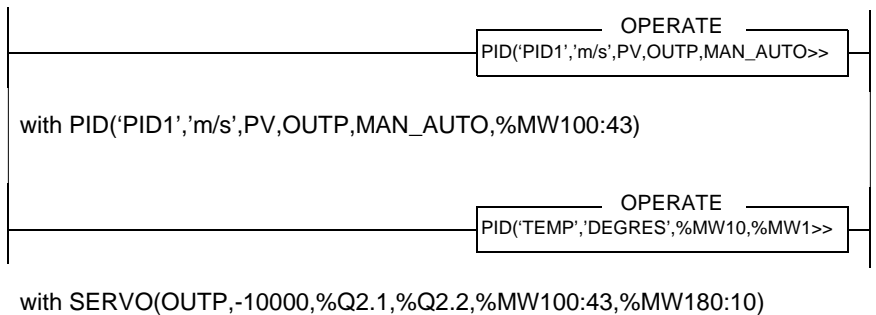
**Examples**

The examples offered below are done in Ladder language.

Case with a position copy.



Case without a position copy.





## Performance of the functions in the operating mode

---

### Introduction

This paragraph describes the performance of the functions in different starting cases:

- cold start (new application, change of cartridge...),
  - warm restart (power return without changing the application context),
  - first execution after adding a function via modification in connected mode.
- 

### Cold start

This type of start occurs for a new application or a change of cartridge.

On a cold start, the PLC can start automatically in RUN (according to the application's configuration). The function correctors have a security performance: manual mode, outputs at 0. In addition, this supports the switching of the PLC into RUN mode without carrying out the PID adjustment, then its debugging with the CCX 17 (the adjustment can only be done in RUN).

---

### Warm restart

This type of restart occurs for a power return, without changing the application context.

With a power return after an outage (regardless of how long it lasted) and if the application context is not lost or modified, the functions go back to their state before the outage. If the user wants to use another performance, it is his responsibility to test the %S1 system bit and to associate the required processing to it (forcing in manual mode...).

**Note:** The PLC's time-and-date stamp allows you find out the duration of the last outage.

---

### Adding a new call in connected mode

Following the addition of a new function regulation call in connected mode, an identical initialization to the case of the cold start is carried out.

**Note:** In order to be seen as a new function, this must use a new parameter table. Therefore, the removal of a PID, followed by adding a PID that uses the same parameter table is not considered as an addition of a new PID. In this case the PID is executed in the same state and with the same parameters as the preceding PID.

---







---

## Introduction

### Subject of this chapter

This chapter presents the operator dialogue on CCX 17.

### What's in this Chapter?

This Chapter contains the following Maps:

Topic	Page
Dialog operator on the CCX 17	392
Selecting a loop	394
Controlling a loop	395
Adjusting a loop	396
PID_MMI Function: programming	397
Performance of the PID_MMI function according to the PLC and CCX 17's operating modes	401



## Dialog operator on the CCX 17

---

### Introduction

The CCX 17 permits viewing and control of all the modifiable parameters of a PID corrector without having to program a specific application.

The dialog operator function integrates a control and adjustment application on the CCX 17 application's PIDs. It provides management of 3 types of screens on the CCX 17 supporting selection of a PID, viewing and controlling of this PID and adjusting the parameters of the PID. It inserts itself easily into any dialog operator application on the CCX 17.

<b>Note:</b> This function is only effective if the PLC is in RUN.
--

### Limitations

**There is no limit to the number of PIDs** in the application. On the other hand, a maximum of 9 PIDs are accessible via the dialog operator function on the CCX 17-30 and on the CCX 17-30.

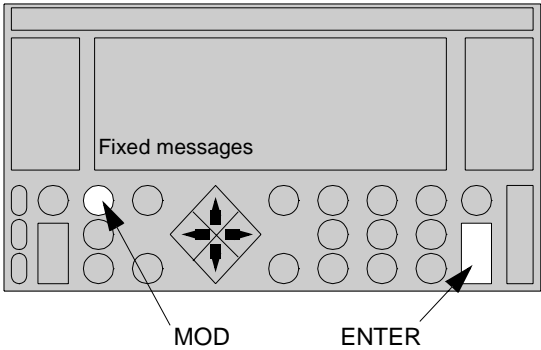
Navigating from one screen to another is done via the CCX's command buttons and within screens via the up and down arrow buttons. The navigation offered is a "vertical" navigation. You must always return to the loop selection screen to gain access to the values of other correctors.

The display is done over 4 lines (8 lines with the CCX 17-30) with the messages in 40 characters.

---



# What the keys do

Position of the keys	Functions
 <p>Fixed messages</p> <p>MOD</p> <p>ENTER</p>	<p>The MOD key supports switching from display mode to input mode (in this case, the selected value starts flashing).</p> <p>On the same screen, the input mode remains active for all the fields and pressing MOD again allows you to exit the input mode (stopping the flashing).</p> <p>In input mode, the modification of a parameter is acknowledged by pressing the ENTER key.</p>

## Set up principle

Setting up the dialog operator is easy:

- the PID\_MMI function(s) are started at each cycle (non-conditioned call),
- a single call to the PID\_MMI function manages all the application's PIDs. However, a call from the PID\_MMI function via CCX\_17, which is connected to the PLC is necessary,
- detection of the application's PIDs via the PID\_MMI function is automatic, including when adding or removing in RUN. Therefore there is no declaration to do,
- Identification of the required corrector is done via the "TAG" parameter from the PID function and its selection depends on the value of the function's "DEVAL\_MMI" parameter. (Only acknowledged via the PID\_MMI function, the PIDs whose DEVAL\_MMI parameter is = 0).

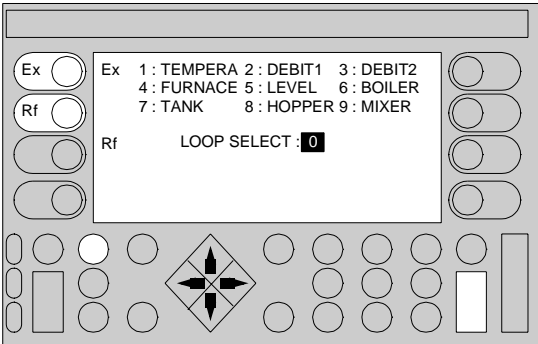


## Selecting a loop

### Introduction

The number of PIDs operated by the CCX 17s is a maximum of 9 loops, regardless of the number of connected CCX 17s.

### Selection screen

Display	Functions
	<p>This screen displays all the labels implemented in the PL7.</p> <p>A figure is associated to each label (from 1 to 9 maximum).</p> <p>To control a loop, the operator must input the corresponding number.</p> <p>After inputting the loop number, the loop controlling screen is displayed.</p> <p>Pressing the Exit (Ex) button allows you to leave the regulation screens.</p> <p>Pressing the Refresh (Rf) button allows you to refresh the screen. This operation must be performed after deleting or adding loops via the PL7 in connected mode.</p>

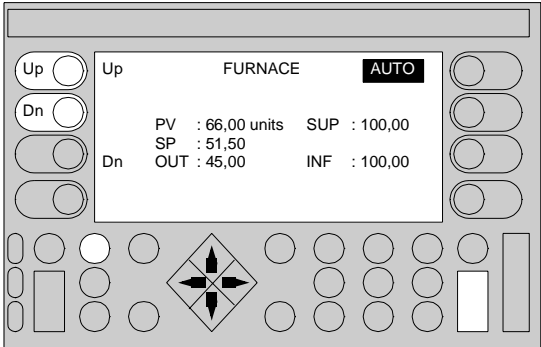
**Note:** If the application has no accessible PID via the CCX 17 (whether there is no PID in the application, or the PID's existing DEVAL\_MMIs are all at 1), the "NO PID" message is displayed. The Exit and Refresh buttons keep their functions.



# Controlling a loop

**Introduction** This screen supports control over the setpoint, command and Manual/Auto mode values. The PV\_INF and PV\_SUP values are also displayed and controllable via this screen and they allow you to set the scale of the measurement in physical units.

## Selection screen

Display	Functions
	<p>The Manual/Auto screen is highlighted. Every time you press the associated command button you switch from one mode to the other.</p> <p>In automatic mode, controlling an output is not authorized.</p> <p>You can switch from one input field to another by using the up and down arrow keys. The operating mode is as follows: as soon as the screen is displayed, it is the <b>SP</b> value which is selected (highlighted), then using the down arrow button, OUT (if manual), LESS and MORE.</p> <p>Pressing MOD supports switching into input mode (press MOD again to exit it).</p> <p>The <b>Dn</b> button gives access to the adjustment screen and to return to the loop selection screen use the <b>UP</b> button. (the PV, SP, OUT, LESS and MORE values are displayed as integers with 2 significant figures after the comma).</p> <p>PV, SP, LESS and MORE are in physical units. OUT is in percentage.</p>

**Note:** When a field is flashing (input mode), the value is not refreshed in case of modification by the application or the PL7.



## Adjusting a loop

**Introduction** This screen supports adjustment of the PID parameters (KP, TI, TD, TS) as well as the output limits OUT\_MIN and OUT\_MAX.

### Selection screen

Display	Functions
	<p>You can switch from one input field to another by using the up and down arrow keys.</p> <p>As soon as the screen is displayed, the KP value is displayed (highlighted).</p> <p>The KP parameter has no unit. TI, TD and TS are in seconds. OUT_MIN and OUT_MAX are in percentage.</p> <p>Pressing the <b>Up</b> button returns you to the loop controlling screen.</p>

**Note:** When a field is flashing (input mode), the value is not refreshed in case of modification by the application or the PL7.



## PID\_MMI Function: programming

### Introduction

The PID\_MMI function supports establishment of the dialog with the PLCs to which the CCX 17 is connected. A PID\_MMI function is necessary via CCX 17 for the steering, the display and the adjustment of the application's PIDs.

The PID\_MMI function is a standard PL7 function. As such, it is available from the functions library.

From the language editors there, it is possible to use the help of a PID\_MMI function's input to facilitate its programming.

**Note:** Inputting a PID\_MMI function must be done in the task with the slowest period containing PIDs (MAST or FAST). The function does not have to be conditioned.

Example: An application with:

- a FAST task at  $10 \cdot 10^{-3}$  s containing PIDs,
  - a MAST task at  $50 \cdot 10^{-3}$  s containing PIDs,
- the PID\_MMI function must therefore be programmed in the MAST task.

### Illustration

The illustration below gives you a general idea of the Functions screen in the library supporting the implementation of the PID\_MMI function.

Family	Lib.V	App.V.	Name	Comment
Orphee Function	2.10	-	PID	Mixed PID regulator
Stalling functions	2.00	-	PID_MMI	Management of operator dialogue dedicated to CCX17 from PID
GRAFCET	1.00	-	PWM	Modulated in the pulse width of a numerical size
Single precision reals	2.22	-	SERVO	PID output stage for open/close valve order
Process Control	2.01	2.01		
Integer tables	2.10	-		

Name	Type	Kind	Comment	Entry field
ADDR	AR_W	IN	Topological address of the destination CCX17 [ta>>	ADR#0.0.4
EN	EBOOL	IN/OUT	Activation of DOP on CCX17	%M1
BUTT	AR_X	IN/OUT	5 bit table linked to the command buttons>>	%M10:5
PARA	AR_Y	IN/OUT	PID_MMI parameters [62 word table]	%MW45:62

Call display

PID\_MMI ( ADR#0.0.4,%M1,%MW10:5,%MW45:62 )



**Syntax**

The PID\_MMI function's call syntax is:

**PID\_MMI (ADDR, EN, BUTT, PARA)**

---

**Parameters of the PID\_MMI function**

The table below presents the different parameters of the PID\_MMI function.

Parameter	Type	Kind IN = Input OUT = Output	Description
ADDR	%MWi:6	IN	CCX 17's address
EN	%Mi	IN / OUT	Activation of the regulation operator dialog. The application puts this bit to 1 and the PID_MMI function puts it back to 0 when you exit the regulation operator dialog (press Ex)
BUTT	%Mi:5	IN / OUT	Bits associated to the CCX 17 buttons. These bits allow you to drive different screens as well as Manual/Auto.
PARA	%MWi:62	IN / OUT	Parameters of the PID_MMI. The first 4 are the words of the communication report.

**Note:** The 4 words of the report are the same as in all the asynchronous communication functions (communication OF, integrated OF DOP and OF PID\_MMI). However the OF PID\_MMI automatically manages these words and the application never has to modify them. They are provided for information. For more information, refer to the Dialog operator (See Operator Dialog functions, p. 281).

**Example of the CCX 17:**

If the CCX 17 is connected directly to the PLC's (UNI-TELWAY) AUX socket, it is at the UNI-TELWAY 4-5 slave addresses.

The coding can be done:

- via immediate value: PID\_MMI(ADR#{0.254}0.0.4,...) or simply:  
PID\_MMI(ADR#0.0.4,...),
- via a table of 6 words: %MW10:6 := ADR#0.0.4 PID\_MMI(%MW10:6,...).

**Synchronization of the dialog operator**

The CCX 17 can be used to show screens other than the regulation screens. The IN bit is there to activate/deactivate the regulation dialog operator.

Setting IN to 1 activates the regulation operator dialog and is displayed on the PIDs' selection screen.

---

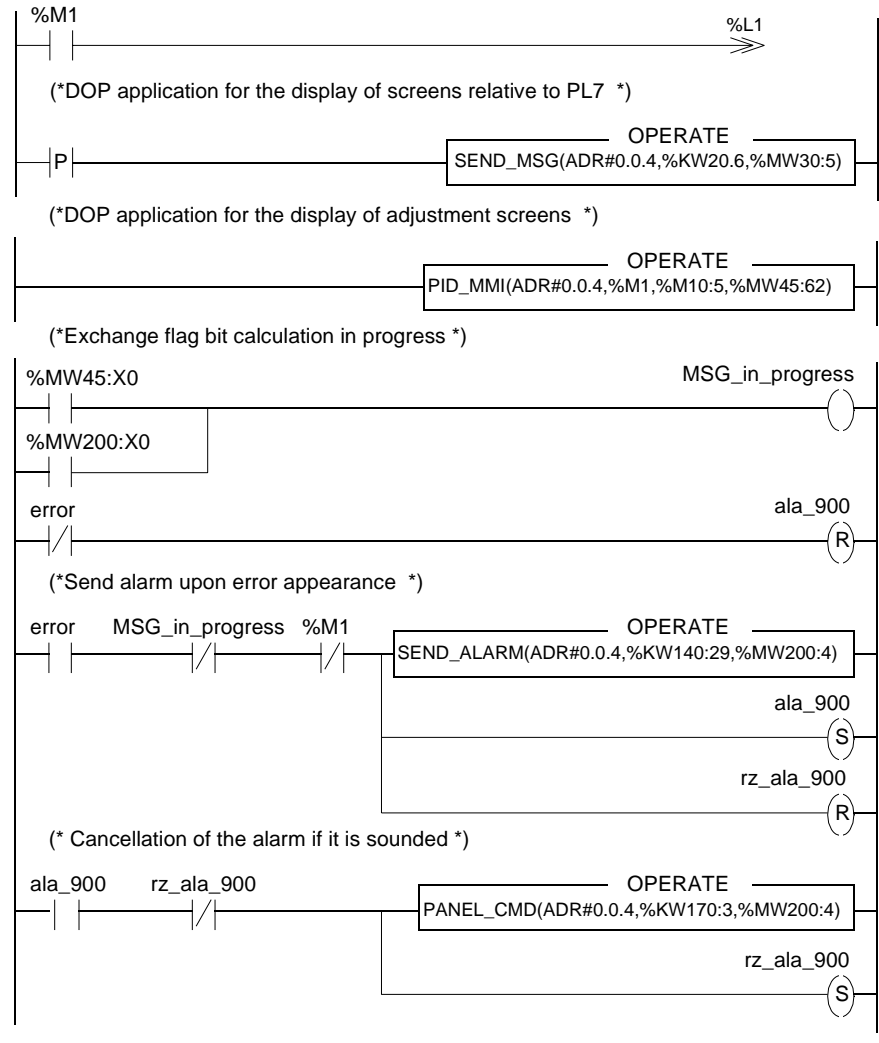


**Examples**

The examples offered below are done in Ladder language.

%MI is associated to the IN bit (display switch on the operator). The alarm management application is always activated, just like the regulation operator dialog.

(\*Management of display communication on the CCX 17\*)





**Management of the command buttons**

When the PID\_MMI is activated (IN at 1), it assigns the CCX 17 command buttons. If the application uses these buttons for anything but regulation, they must be reassigned on the falling edge of IN (using the ASSIGN\_KEYS function described in the DOP (See Operator Dialog functions, p. 281)). On the other hand, if the CCX 17 is only for regulation, it is advised that you carry out a non-conditioned SET of the IN bit in the application.

---

**Selection of the PIDs managed by the PID\_MMI function**

Each PID possesses a bit type DEVAL\_MMI parameter. If this bit is at 1, the PID is not managed by PID\_MMI. It is the only available level of protection. Moreover, if the application has more than 9 PIDs, it is the way to master those that are handled by the PID\_MMI.

---

**Alarm management**

It is up to the user to program his/her own alarm management. This overlays itself on the management of the regulation screens.

If an alarm (coming from the dialog operator's application) goes off while one of the 3 regulation screens is displayed, the CCX\_17 is then dedicated to managing the alarm messages. When you return to the regulation operator dialog, the screen appears incomplete. Up/Dn or Refresh allows you to refresh this screen.

---

**Several PID\_MMI functions**

It is possible to connect several CCX 17 terminals to the same PLC, so it can be useful to have several PID\_MMI in the same application. In this case, the different PID\_MMI must be executed consecutively (no integrated PID call) via the same PL7 task.

---



## Performance of the PID\_MMI function according to the PLC and CCX 17's operating modes

---

**Introduction** This paragraph describes the performance of the PID\_MMI function according to the different operating modes of the PLC and the CCX 17:

- warm restart,
- crossing into Run or Stop,
- reconnecting to the CCX 17.

---

**Warm start** This type of restart occurs for a power return, without changing the application context.  
If a problem such as a micro-outage on the PLC occurs while sending a message, the command is not repeated. It is therefore necessary to reinitialize the dialog by activating the IN bit via the application program.

---

**STOP/RUN and RUN/STOP crossing** In STOP, the PID\_MMI function is no longer active. Nevertheless, you can still output parameters belonging to the displayed screen.  
In STOP/RUN, the function goes back to the state it was in before crossing into STOP.

---

**Power outage or reconnecting to the CCX 17** With a power return or reconnection to the CCX 17, this reinitializes the communication with the PLC. Periodically, the PID\_MMI reassigns the CCX 17's command buttons. So after 20 seconds or more, pressing one of the first 3 buttons will display one of the regulation screens (preferably the Ref or Dn button on the left of the second row).

---

**Note:** Via the application, it is also possible to detect the presence of the CCX 17 by using the language words associated to the communication channels and to manage the dialog reset via the IN bit.

---

**Cold start** It is only on a cold start that the regulation screens are reset.

---







---

Introduction

Subject of this chapter

This chapter presents an example of application.

What's in this Chapter?

This Chapter contains the following Maps:

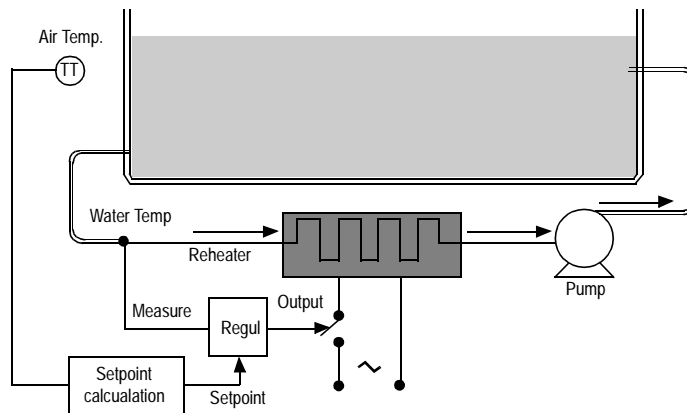
Topic	Page
Description of the application example	404
Configuration of the example	406
Programming the example	409



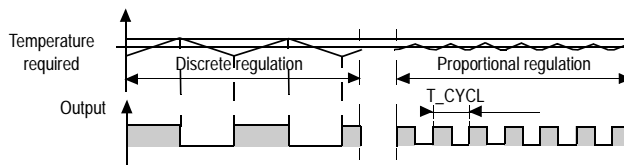
## Description of the application example

### Context

This is about maintaining the water temperature of an open air swimming pool at a required value. This value itself is determined according to the ambient air temperature.



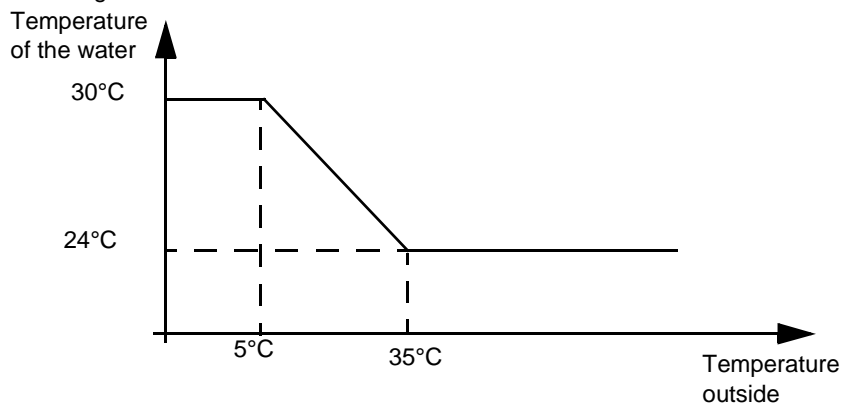
A discrete regulation is generally used in this type of installation. It is suggested that in this example, you substitute it for a proportional regulation with a modulated output, which should support reduction in the size of the temperature oscillations around the required value.



Measuring the water temperature as well as the ambient temperature is done using a Pt 100 resistance thermometer.



The setpoint of the water temperature depends on the exterior temperature according to the below law:



- A HIGH TEMPERATURE alarm will go off if the water temperature exceeds 32°C,
- A LOW TEMPERATURE alarm will go off if it falls below 22°C,
- A REGULATION ERROR alarm will go off if the SETPOINT/MEASUREMENT gap exceeds 2°C in either direction,
- The regulation will become inoperative (output at 0) in case the pump stops.



## Configuration of the example

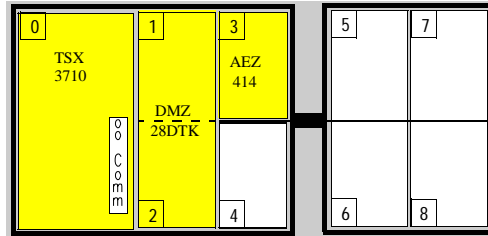
---

### Hardware configuration

This application uses:

- a TSX 37-10 PLC
- a TSX DMZ 28DTK discrete I/O module.
- a TSX AEZ 414 analog input module,

The configuration is thus as follows:



### Assignment

The discrete output %Q2.0 is assigned to the control the reheater.

The discrete output %Q2.1 is assigned to the control the pump.

The discrete outputs %Q2.2, %Q2.3 and %Q2.4 are assigned to alarms.

The bit %M0 is used to select the process controller operating mode AUTO/MANU.

The discrete inputs %I1.1 and %I1.2 are used to modify the setpoint value in AUTO mode and the output value in MANU mode according to the following algorithm:

- %I1.1 = 1 increase of 0.1 % per cycle,
- %I1.2 = 1 decrease of 0.1 % per cycle.

The input %I1.3 supplies information on the state of the pump.

The input %I1.4 is used to control the pump.

The input %I1.6 controls the display on the CCX 17.

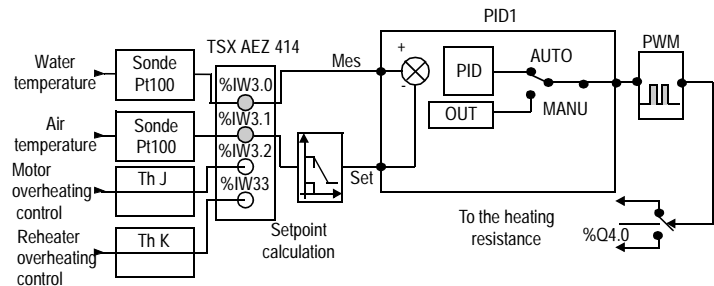
%IW3.0 to %IW3.3 are the values of the analog inputs.

---



Diagram of the process control loop

PID process controller action is in the INVERSE direction (an increase in the process value must correspond to a decrease in the output value).



Configuration

Rack configuration

Slot	Family	Product reference
0	Processors	TSX 3710
1	Discrete Inputs	TSX DMZ 28DTK
2	Discrete Outputs	
3	Analog	TSX AEZ 414

Channel parameters for TSX DMZ 28DTK module

Channel	Type of channel	Address	Symbol	Task
0	Input	%I1.0	-	MAST
1	Input	%I1.1	CONSIG_INCREM	MAST
2	Input	%I1.2	CONSIG_DECREM	MAST
3	Input	%I1.3	ETAT_POMPE	MAST
4	Input	%I1.4	ACT_POMPE	MAST
5	Input	%I1.5	-	MAST
6	Input	%I1.6	VALID_DOP_REG	MAST
7 to 16	Inputs not used			
0	Output	%Q2.0	COMM_RECHAUF	MAST
1	Output	%Q2.1	COMM_POMPE	MAST
2	Output	%Q2.2	ALARM_TEMP_HAUT	MAST
3	Output	%Q2.3	ALARM_TEMP_BAS	MAST
4	Output	%Q2.4	ALARM_DEF_REG	MAST
5 to 12	Outputs not used			



---

**Channel parameters for TSX AEZ 414 module**

Channel	Address	Symbol	Range	Scale	Min.	Max.	Unit	Filtering	Task
0	%IW3.0	TEMP_EAU	Pt100	%..	0	500	°C	0	MAST
1	%IW3.1	TEMP_AIR	Pt100	%..	-200	800	°C	0	MAST
2	%IW3.2	SURCHAUF_MOTEUR	Thermo J	%..	0	1000	°C	0	MAST
3	%IW3.3	SURCHAUF_RECHAUF	Thermo K	%..	0	1000	°C	0	MAST

**Configuration of bits, words and function blocks**

Bit	Words	Function blocks
Internal (%M): 256 System (%S): 128	Internal (%MB,%MW,%MD,%MF): 512 System (%SW,%SD): 128 Common (%NW): 0 Constant (%KB,%KW,%KD,%KF): 128	Series 7 timer(s) (%T): 0 Timer(s) (%TM): 64 Monostable(s) (%MN): 4 Counter(s) (%C): 32 Register(s) (%R): 4 Drum(s) (%DR): 2

---



---

## Programming the example

---

### Proposed processing method

The block PID1 is assigned to temperature process control. The temperature setpoint is calculated on the basis of the ambient temperature.

On reconnection to the mains, the process control operating mode is selected and the pump is started.

The state of the controller is conditioned by the operating state of the pump. If the pump is faulty the PID switches to MANU and the output is forced to 0.

The status word bits (process value high threshold, process value low threshold, deviation high threshold and deviation low threshold) are used to generate alarms.

The PID loop coefficients will be initialized to:

- $K_P = 600$
- $T_I = 300$
- $T_D = 50$

The display on the CCX is as follows:

- $K_P = 6$
- $T_I = 30$
- $T_D = 5$

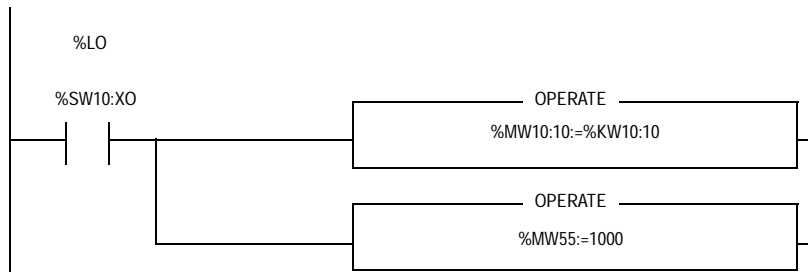
These values can of course be fine-tuned during a subsequent adjustment phase.

---



## MAST-MAIN

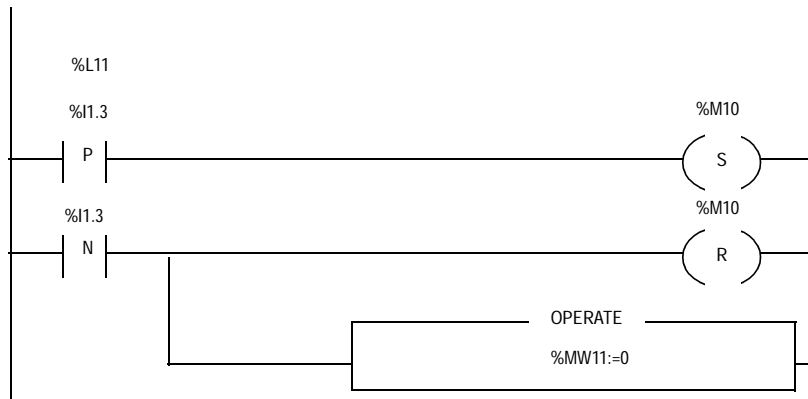
(\*Initialization of constants on cold restart-> PID loop buffer and initialization of PWM period to 10 s\*)



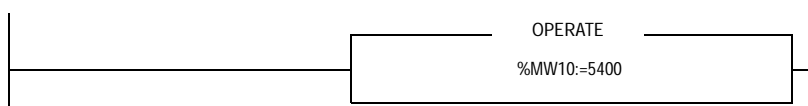
(\*Pump activation \*)



(\* PID loop controller operating mode management. This program allows the CCX17 to modify the A/M bit \*)

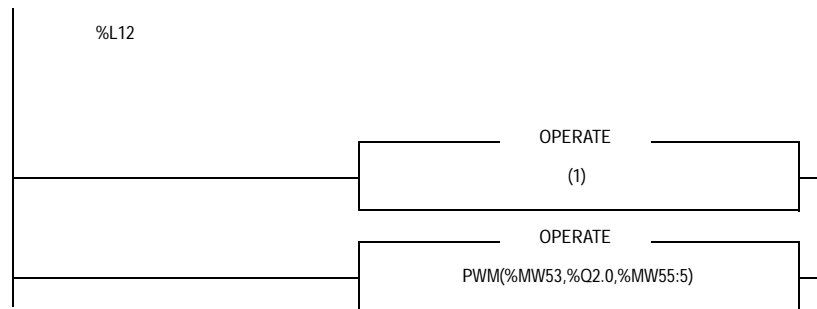


(\* Initialization of water temperature setpoint to 27 °C \*)

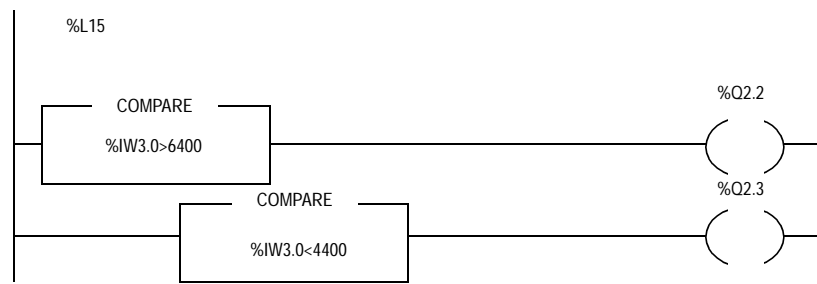




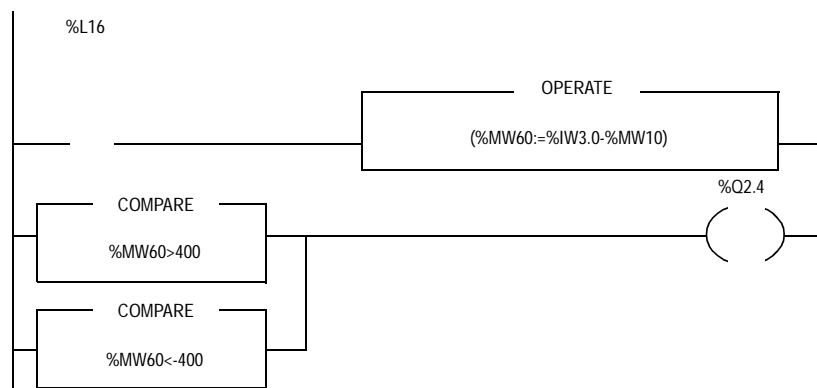
(\* Execution of the temperature process control loop \*)



(\* Management of process value alarms \*)

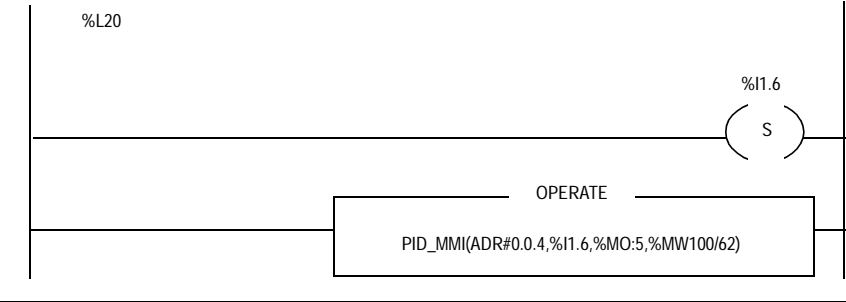


(\* Management of deviation alarms \*)





(\* PID loop controller on CCX17 \*)





---

At a Glance

**Overview** This Chapter reviews several aspects of the process control application.

**What's in this Chapter?** This Chapter contains the following Maps:

Topic	Page
PID parameter adjustment method	414
Role and influence of PID parameters	417



## PID parameter adjustment method

### Introduction

Numerous methods to adjust the PID parameters exist, we suggest Ziegler and Nichols which have two variants:

- closed loop adjustment,
- open loop adjustment.

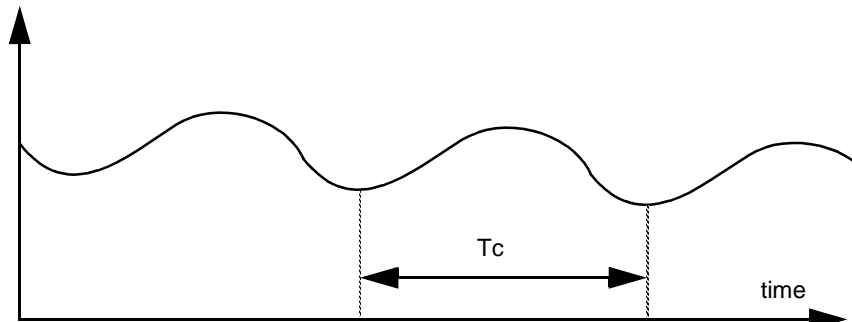
Before implementing one of these methods, you must set the PID action direction:

- if an increase in the OUT output causes an increase in the PV measurement, make the PID inverted ( $K_P > 0$ ),
- on the other hand, if this causes a PV reduction, make the PID direct ( $K_P < 0$ ).

### Closed loop adjustment

This principal consists of using a proportional command ( $T_i = 0$ ,  $T_d = 0$ ) to start the process by increasing production until it starts to oscillate again after having applied a level to the PID corrector setpoint. All that is required is to raise the critical production level ( $K_{pc}$ ) which has caused the non damped oscillation and the oscillation period ( $T_c$ ) to reduce the values giving an optimal regulation of the regulator.

Measure



According to the kind of (PID or PI) regulator, the adjustment of the coefficients is executed with the following values:

-	$K_p$	$T_i$	$T_d$
PID	$K_{pc}/1,7$	$T_c/2$	$T_c/8$
PI	$K_{pc}/2,22$	$0,83 \times T_c$	-

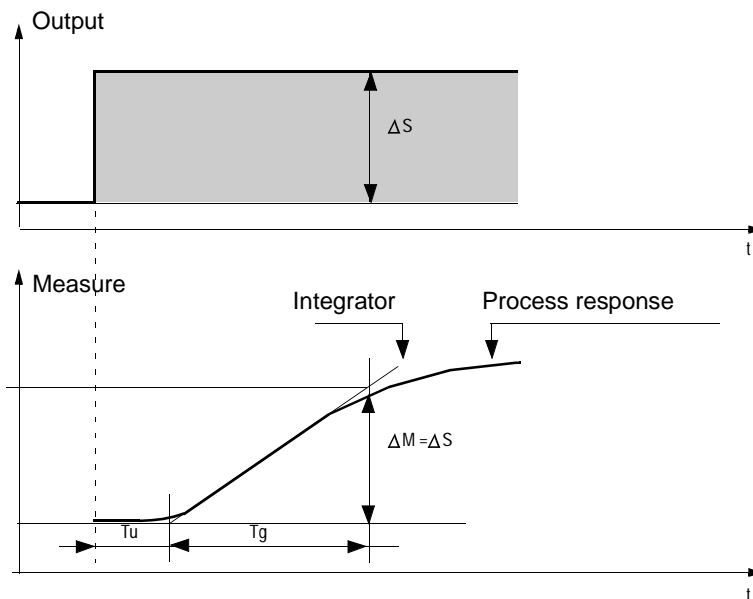
where  $K_p$  = proportional production,  $T_i$  = integration time and  $T_d$  = diversion time.

**Note:** This adjustment method provides a very dynamic command which can express itself through unwanted overshoots during the change of setpoint pulses. In this case, lower the production value until you get the required behaviour.



## Open loop adjustment

As the regulator is in manual mode, you apply a level to the output and make the procedure response start the same as an integrator with pure delay time. .



The intersection point on the right hand side which is representative of the integrator with the time axes, determines the time  $T_u$ . Next,  $T_g$  time is defined as the time necessary for the controlled variable (measurement) to have the same variation size (% of the scale) as the regulator output.

According to the kind of (PID or PI) regulator, the adjustment of the coefficients is executed with the following values:

-	$K_p$	$T_i$	$T_d$
PID	$-1,2 T_g/T_u$	$2 \times T_u$	$0,5 \times T_u$
PI	$-0,9 T_g/T_u$	$3,3 \times T_u$	-

where  $K_p$  = proportional production,  $T_i$  = integration time and  $T_d$  = diversion time.

**Note:** Attention to the units. If the adjustment is carried out in PL7, multiply the value obtained for  $K_P$  by 100.



This adjustment method also provides a very dynamic command, which can express itself through unwanted overshoots during the change of setpoints' pulses. In this case, lower the production value until you get the required behavior. The method is interesting because it does not require any assumptions about the nature and the order of the procedure. You can apply it just as well to the stable procedures as to real integrating procedures. It is really interesting in the case of slow procedures (glass industry,...) because the user only requires the beginning of the response to regulate the coefficients  $K_p$ ,  $T_i$  and  $T_d$ .

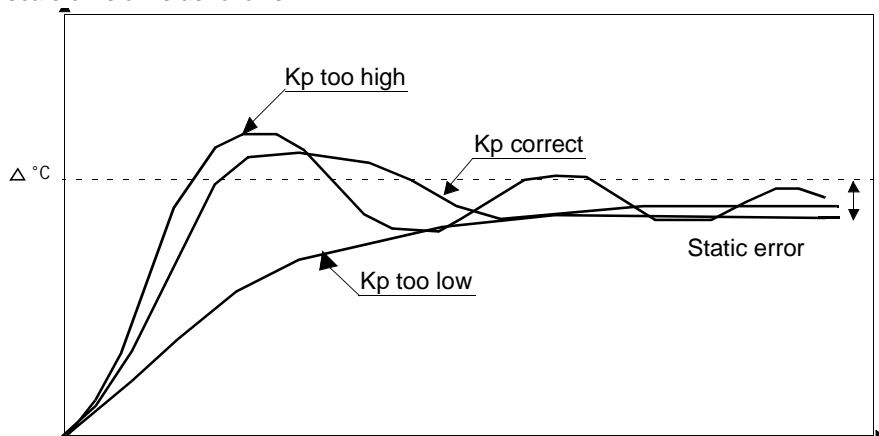
---



## Role and influence of PID parameters

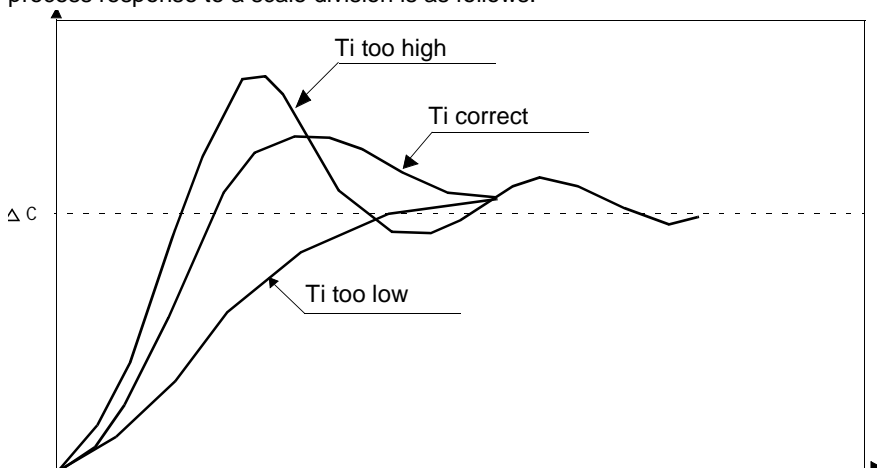
### Influence of proportional action

Proportional action is used to influence the process response speed. The higher the gain, the faster the response, and the lower the static error (in direct proportion), though the more stability deteriorates. A suitable compromise between speed and stability must be found. The influence of integral action on process response to a scale division is as follows:



### Influence of integral action

Integral action is used to cancel out static error (deviation between the process value and the setpoint). The higher the level of integral action (low  $T_i$ ), the faster the response and the more stability deteriorates. It is also necessary to find a suitable compromise between speed and stability. The influence of integral action on process response to a scale division is as follows:



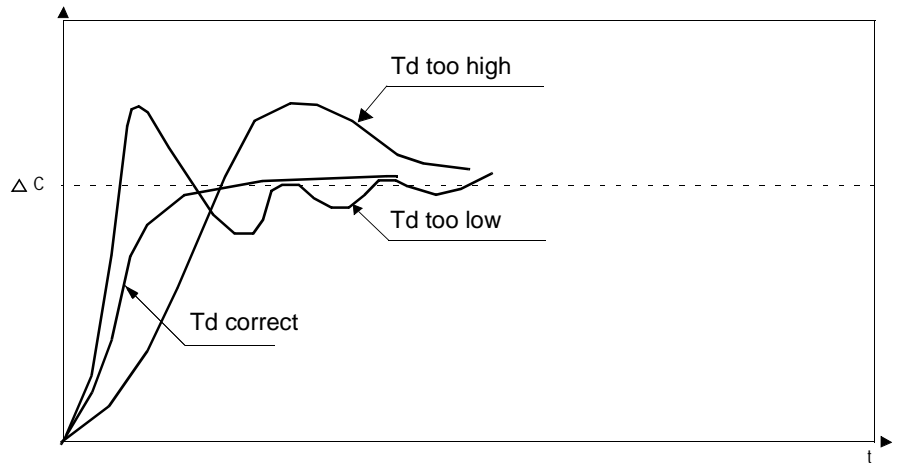


**Note:** A low  $T_i$  means a high level of integral action.

where  $K_p$  = proportional gain,  $T_i$  = integration time and  $T_d$  = derivative time.

### Influence of derivative action

Derivative action is anticipatory. In practice, it adds a term which takes account of the speed of variation in the deviation, which makes it possible to anticipate changes by accelerating process response times when the deviation increases and by slowing them down when the deviation decreases. The higher the level of derivative action (high  $T_d$ ), the faster the response. A suitable compromise between speed and stability must be found. The influence of derivative action on process response to a scale division is as follows:





## Limits of the PID control loop

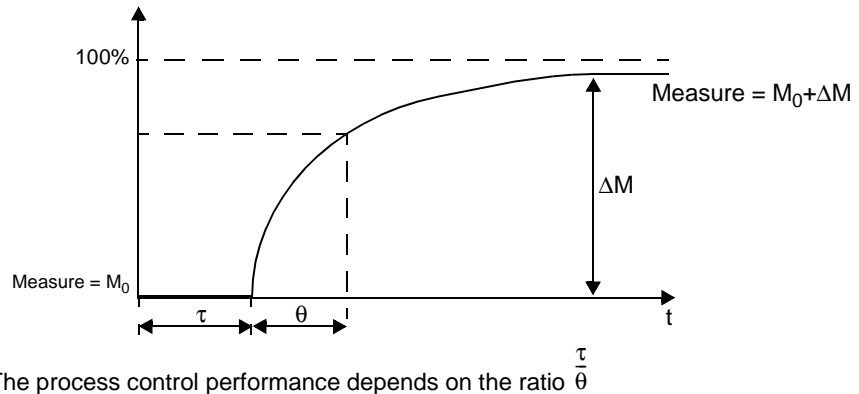
If the process is assimilated to a pure delay first order with a transfer function:

$$(H(p)) = K \frac{(e^{(-\tau)p})}{(1 + \theta p)}$$

where:

$\tau$  = model delay,

$\theta$  = model time constant,



The process control performance depends on the ratio  $\frac{\tau}{\theta}$

The suitable PID process control is attained in the following domain:  $2 - \frac{\tau}{\theta} - 20$

For  $\frac{\tau}{\theta} < 2$ , in other words for fast control loops (low  $\theta$ ) or for processes with a large delay (high  $\tau$ ) the PID process control is no longer suitable. In such cases more complex algorithms should be used.

For  $\frac{\tau}{\theta} > 20$ , a process control using a threshold plus hysteresis is sufficient.

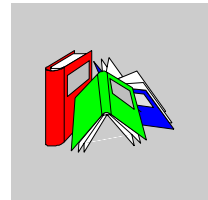






---

# Glossary



---

## A

<b>AS-interface</b>	Actuator Sensor interface
---------------------	---------------------------

---

## C

<b>CCX17</b>	Family of Schneider Automation Human-Machine Interface terminal
--------------	---

<b>Channel group</b>	Channels of the same type with common parameters. This notion concerns certain application-specific modules such as discrete modules.
----------------------	---

<b>CPU</b>	Central Processing Unit: generic name used for Schneider Automation processors
------------	--

---

## D

<b>Discrete</b>	Discrete I/Os
-----------------	---------------

---



**F**

**FIPIO** Field bus used to connect sensor or actuator type devices.

---

**I**

**IP67** Family of Schneider Automation hardware products consisting of sealed I/O modules which connect to the FIPIO field bus, used to produce automated systems with distributed I/Os.

---

**M**

**Momentum** I/O modules using several open standard communication networks.

---

**P**

**PL7** Schneider Automation PLC programming software.

**PV** Identifier indicating the product version.

---

**T**

**TBX** I/O modules remoted on the FIPIO bus.

**TSX/PMX/PCX57** Families of Schneider Automation hardware products.

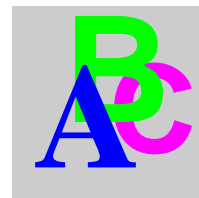
---



---

# Index

---



## Symbols

%CHy.i, 35  
%IWx.i, 274  
%KW, 277  
%MWx, 275  
%QWx.i, 274

## A

Accessing the configuration editor, 56  
    AS-i Bus, 124  
    discrete Micro, 72  
Adding a profile  
    AS-I Bus, 133  
Adding a slave  
    AS-i Bus, 128  
Addressing  
    AS-i Bus, 151  
    AS-i bus, 30  
    built-in interfaces, 26  
    In-rack modules, 28  
Addresssing  
    Discrete, 107  
Adjustment, 21  
    AS-i Bus, 144  
Adjustment objects  
    AS-i Bus, 160  
Advanced operation  
    AS-i Bus, 168  
Alarm output  
    discrete, 93  
Analog, 173

analog, 171  
Application-specific function, 65  
Application-specific instructions, 52  
AS-i, 115  
AS-i Bus, 118  
AS-i slave  
    AS-i bus, 122  
ASSIGN\_KEYS, 326  
Automatic slave addressing  
    AS-i bus, 135

## B

Built-in application-specific interfaces, 26

## C

Closed loop adjustment, 414  
Command objects  
    AS-i Bus, 159  
Communication Interface, 26  
Configuration, 20  
Configuration screen, 70  
    AS-i Bus, 126  
Confirming the configuration, 60, 61  
Connecting  
    AS-i bus, 138  
CONTROL\_LEDS, 323  
Controlling a loop, 395  
Copy/paste, 74



## D

- Debug, 22
  - AS-i bus, 138
- Debug function
  - Discrete, 96
- Debug mode, 95
- Debug screen
  - AS-i Bus, 139
- Declaration of an I/O module, 59
- Derivative action, 418
- Device connection
  - AS-I Bus, 128
- Diag
  - discrete, 100
- Diagnostics, 22
  - Bus AS-i, 141
- Diagnostics mode
  - AS-i Bus, 143
  - Bus AS-i, 141
- Dialog operator, 392
- Discrete, 63, 65, 70
- DISPLAY\_GRP, 315
- DISPLAY\_MSG, 314

## E

- Error bits, 274
- Event input, 84
- Event on a falling edge
  - Discrete, 84
- Event on a rising edge
  - discrete, 84
- Event processing
  - discrete, 84
- Evti
  - Discrete, 84
- Example of application, 403
- Exchange management, 110
- Exchange report, 111
- Explicit exchange
  - Discrete, 113
  - discrete, 114
- Explicit exchanges, 35
  - AS-i Bus, 157

- Explicit objects, 112, 155
  - AS-i Bus, 156, 161
- External errors
  - AS-i Bus, 141
- External supply error, 114
- External supply fault, 83

## F

- Fallback mode
  - discrete, 89
- Fallback to 0
  - AS-i bus, 136
  - discrete, 89
- Family
  - AS-i Bus, 123
- Fault processing by program, 62
- Faulty slave
  - AS-i bus, 147
- Filtering time
  - discrete, 88
- Forcing, 101
  - AS-i Bus, 145
- Function
  - Adjustment, 21
  - Configuration, 20
  - Debug, 22
- Functions
  - discrete, 84

## G

- GET\_MSG, 306
- GET\_VALUE, 321

## I

- Implicit exchange
  - Discrete, 109
- Implicit exchanges, 32
  - AS-i Bus, 153
- Integral action, 417
- Internal errors
  - AS-i Bus, 141



## L

- Language objects, 110, 111, 112, 154, 155
  - AS-i Bus, 156, 161
- Latching
  - discrete, 84
- Library, 52
- limitations, 236

## M

- Maintain
  - AS-i bus, 136
- Maintain outputs
  - discrete, 104
- Maintain state
  - discrete, 89
- Managing exchanges, 154
- Master/Slave
  - AS-i Bus, 120
- Micro discrete inputs, 77
- Micro discrete outputs, 79
- Missing module, 100
- Modifying parameters, 74
- Module diagnostics
  - discrete, 100
- Module fault
  - discrete, 100
- Module selection
  - AS-i Bus, 123
- Multiple selection, 74

## N

- Network frequency
  - discrete, 87

## O

- Offline mode
  - AS-i Bus, 168
- Open loop adjustment, 415
- Operating mode
  - AS-i Bus, 164
- Operating modes, 389
- Operating modes of the dialog operator, 401

## P

- Parameters, 77, 79
  - Discrete, 82, 83
- Performance
  - AS-i bus, 169
- PID function, 371
- PID\_MMI function, 397
- PL7 status bar, 58
- PL7 Toolbar, 57
- Presymbolization, 49, 50
- Process control functions, 363
- Profile
  - AS-i Bus, 120
- Programming rules, 370
- Proportional action, 417
- Protected mode
  - AS-i bus, 166
- PWM function, 378

## R

- Reactivation of outputs
  - Discrete, 90
  - discrete, 103
- READ\_PARAM, 40
- READ\_STS, 37
- Reading adjustment parameters, 40
- Reading status words, 37
- Regulation functions, 369
- RESET, 102
  - AS-i Bus, 146
- RESTORE\_PARAM, 44
- Restoring adjustment parameters, 44
- RUN/STOP input
  - discrete, 91

## S

- Save program and % MW input
  - Discrete, 92
- SAVE\_PARAM, 42
- Saving adjustment parameters, 42
- Selecting a loop, 394
- Selection of modules
  - Discrete, 59



SEND\_ALARM, 311  
SEND\_MSG, 304  
SERVO function, 382  
SET, 102

AS-i Bus, 146

Sink input, 86

Slave configuration

AS-i Bus, 126

Slave number

AS-i bus, 129

Slaves

AS-i Bus, 143

Slaves' status

AS-i Bus, 143

Software installation

General, 18

Source input, 86

Status objects

AS-i Bus, 153, 157

Structure of an AS-i slave

AS-i bus, 122

WRITE\_CMD, 39

WRITE\_PARAM, 41

Writing adjustment parameters, 41

Writing command words, 39

## T

Task

Discrete, 82

Terminal Port, 26

Tripped output

discrete, 100, 103

TSX SAZ 10

AS-i Bus, 120

TSX SAZ 10 module

AS-i Bus, 120

Type of inputs

discrete, 86

## U

Unforcing, 101

As-i Bus, 145

## W

Write command, 102

Write to 0, 102

Write to 1, 102