

# TCWIN

## Software Manual

ABB reserves the right to change the information contained in this document without notice. The information represents no obligation on the part of the company.

All products referred to are covered by the appropriate trademark and/or copyright legislation.

---

## Before you read

Each reader of this manual can tailor the way its contents are read to his or her own preferred way of learning.

If you want to get to know first all the functions the TC offers you and then experiment by creating a project, you need simply follow the order of the chapters.

Alternatively, if you want to start creating a demonstration project immediately, analyzing the functions as they are met with, you need to proceed to “Chapter 3 -> TCWIN menus“ and then jump directly to “Chapter 6 -> Using TCWIN“.

---



# Contents

<b>Foreword</b>	The manual .....	F-1
	How can it help you? .....	F-1
	Conventions .....	F-1
<b>Introduction</b>	What is TCWIN? .....	I-1
	Requirements for displaying the Help on Line .....	I-1
	File architecture .....	I-1
	What is a project? .....	I-2
	Files generated by a project.....	I-2
<b>Installing TCWIN</b>	HW requirements: minimum specifications .....	1-1
	HW requirements: ideal specifications .....	1-1
	Installation procedure.....	1-1
<b>Programming functions</b>	Field and Variable .....	2-1
	Relationship between fields and variables.....	2-1
	List of functions .....	2-1
<b>TCWIN menus</b>	List of menus.....	3-1
<b>The functions in detail</b>	Page.....	4-1
	Multilanguage label .....	4-1
	Multilanguage text .....	4-2
	Numerical field .....	4-2
	ASCII field .....	4-8
	Dynamic Text Field .....	4-10
	Variables .....	4-13
	Page sequences .....	4-16
	Information Messages.....	4-18
	Direct Commands .....	4-19
	Text Lists.....	4-21
<b>The menus in detail</b>	File .....	5-1
	Tools .....	5-2
	Object.....	5-2
	Fields .....	5-2
	Edit.....	5-3
	Page.....	5-3
	Configure .....	5-4
	Windows .....	5-14
	TCWIN language .....	5-14
	? .....	5-14
<b>Using TCWIN</b>	Terminology used .....	6-1
	Forms assumed by the mouse pointer.....	6-1
	Meaning of menu icons.....	6-1

---

<b>Creating a project with TCWIN</b>	Creating the project.....	7-2
	Project information .....	7-4
	Setting project languages.....	7-5
	Project setup .....	7-7
	Inserting variables .....	7-8
	Inserting pages.....	7-10
	Inserting sequences .....	7-17
	Inserting direct commands .....	7-18
	Data exchange area.....	7-22
Information messages.....	7-25	
<b>Compiling and transferring a project</b>	Compiling a project .....	8-1
	Trasferring the project.....	8-2
<b>Creating and printing documentation</b>	Importance of documentation.....	9-1
	Print the project.....	9-1
<b>Creating a back-up of the project</b>	Importance of a Back-up .....	10-1
	How to create a Back-up.....	10-1
<b>Defining the fonts</b>	Meaning of the icons used in the menus.....	11-2
	Personalizing a Font .....	11-3
<b>Appendix A</b>	Variables .....	AA-1
	List of pages.....	AA-1
	Screen sequence .....	AA-1
	Information messages.....	AA-2
	Direct commands .....	AA-2
	Translations (Part 1 of 2).....	AA-2

---

---

## Foreword

**The manual** The programming manual is the tool that allows the user to create his or her own application packages for the VIDEO TERMINALS (TC) by using the TCWIN programming package.

**How can it help you?** The manual contains all the functions, instructions, concepts and examples necessary for the user to learn quickly and easily.

**Conventions** Below is a list of representational devices used in this manual together with their respective meanings:

*File > Open* This style is used to indicate a menu option. It represents the complete path necessary for reaching the option required.

*Label* This style is used to indicate a data input field.

[] The contents are appear on the display.

⌘ Indicates that this input field must be used.

□ Identifies a key.

📖 Identifies an option within a window.

📁 Identifies a folder.

⚠ Draws attention to essential points.

---



# Introduction

## What is TCWIN?

TCWIN is a program that allows the user to create the application package required to work on the TC. It is easy to use and simple to understand.

TCWIN will only work in a **Windows 95/98** or **Windows NT** environment.

## Requirements for displaying the Help on Line

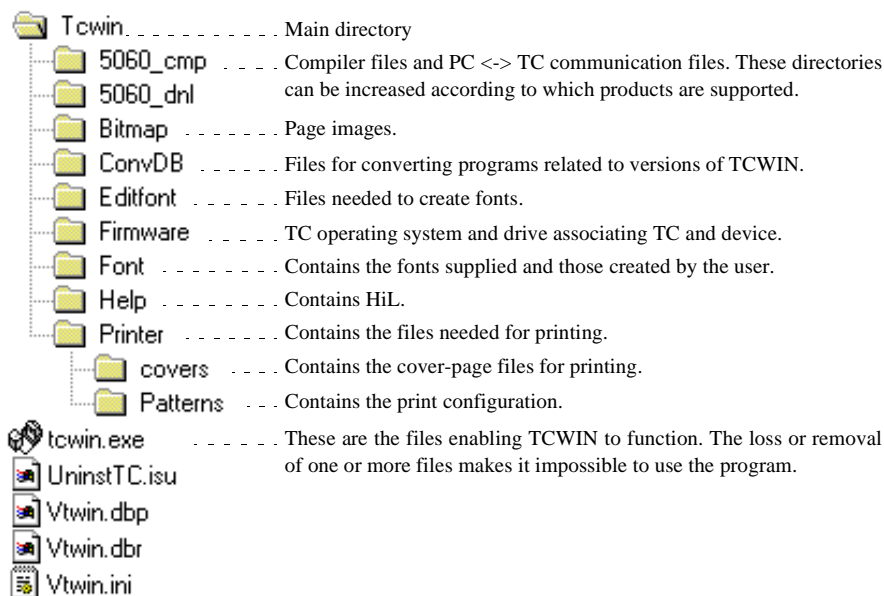
TCWIN contains an exhaustive Help on Line (HoL) clarifying the vast majority of doubts that a user might have. To have the HoL on screen a Browser needs to have been installed on the Personal Computer (PC) to display pages in HTML format. (Typical Browsers: Internet Explorer, Netscape Communicator or equivalents.)

**(This type of HoL will be available with the forthcoming versions).**

Currently TCWIN contains a HoL that does not require any special programs for displaying it.

## File architecture

TCWIN, after it has been installed, creates a structure which we show below together with the contents of the various files.



## What is a project?

A PROJECT may be defined as a set of screens (defined later as PAGES) having the same dimensions as the display of the TC being used. All pages can be freely configured by the user, so as to be able to contain texts and/or the display/setting of process variables. The various pages configured in this way can be freely interconnected to create the best way for the user to navigate between them. Finally, every project can enable the user to create appropriate diagnostic comments in order to signal the occurrence of anomalous events in the process.

To sum up: a PROJECT can be considered as a more or less complex system of pages whose aim is to enable the handling and/or display of a productive process.

## Files generated by a project

Table 0.1 lists the extensions of the files generated by the project.

Table 0.1: Significance of the files

Extension	Location	Significance
.MDB	Project directory	Project file -- all the files needed for the project are obtained from this file. <b>Loss of this file will cause the project to be lost.</b>
.OBJ	Project directory	This file is obtained by compiling the project file (.mdb) containing the text part of the project. This file is generated every time the project is compiled. It does not need to be kept in the archive.
.BIN	Main directory of TCWIN	This file is transferred to the TC. It is obtained after giving a command to transfer the project from the PC to the TC. It groups together the information contained in the .obj and .objg files. It is not necessary to keep this file in the archive.
.PRJ	Main directory of TCWIN	Temporary project file. A numerical file may be found with this extension if TCWIN has been closed incorrectly. <b>The file can be removed once TCWIN has been closed.</b>
.BMP	Main directory of TCWIN	Temporary project file. A numerical file may be found with this extension if TCWIN has been closed incorrectly. <b>The file can be removed once TCWIN has been closed. (Before removing check that the file has not been created on purpose by the user.)</b>
.LBD	Project directory	Temporary information file on database record block. A numerical file may be found with this extension if TCWIN has been closed incorrectly. <b>The file can be removed once TCWIN has been closed.</b>



The removal of the .MDB file will cause an irrevocable loss of the project.

---

## Chapter 1      Installing TCWIN

### **HW requirements: minimum specifications**

For TCWIN to work properly the machine must be configured as follows:

- Processor:            486 DX/2
- Operative system: Windows 95 / 98 / NT 3.51 or later
- RAM memory:        8 Mbytes

### **HW requirements: ideal specifications**

For TCWIN to work at its best the machine must be configured as follows configured as follows:

- Processor:            PENTIUM 133Mhz or later
- Operative system: Windows 95 / 98 / NT 3.51 or later
- RAM memory:        16 Mbytes

### **Installation procedure**

Insert the relevant medium in the appropriate drive and click on *Avvio > Esegui...*

Digit a:\setup.exe and confirm by pressing OK.



**If the drive to be used is not “a:”, put in the appropriate letter.**

Follow the instructions on screen.

---



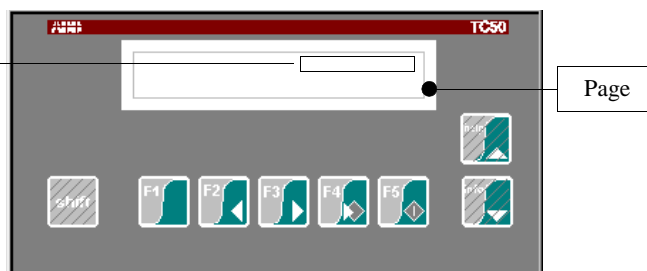
## Chapter 2 Programming functions

### Field and Variable

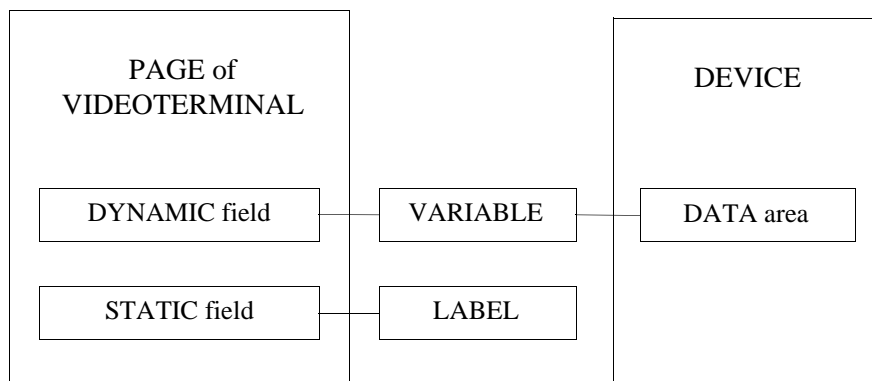
Before dealing with the programming functions available, it is essential that certain fundamental concepts be defined.

The programming packages often contain the terms FIELD and VARIABLE.

We use the word FIELD to signify an area of the page that can take on certain meanings. A field can be either STATIC or DYNAMIC. By static field we mean a field that does not change the display status in the page; by dynamic field we mean a field that changes the display status in the page as a function of the VARIABLE assigned to the device connected.



### Relationship between fields and variables

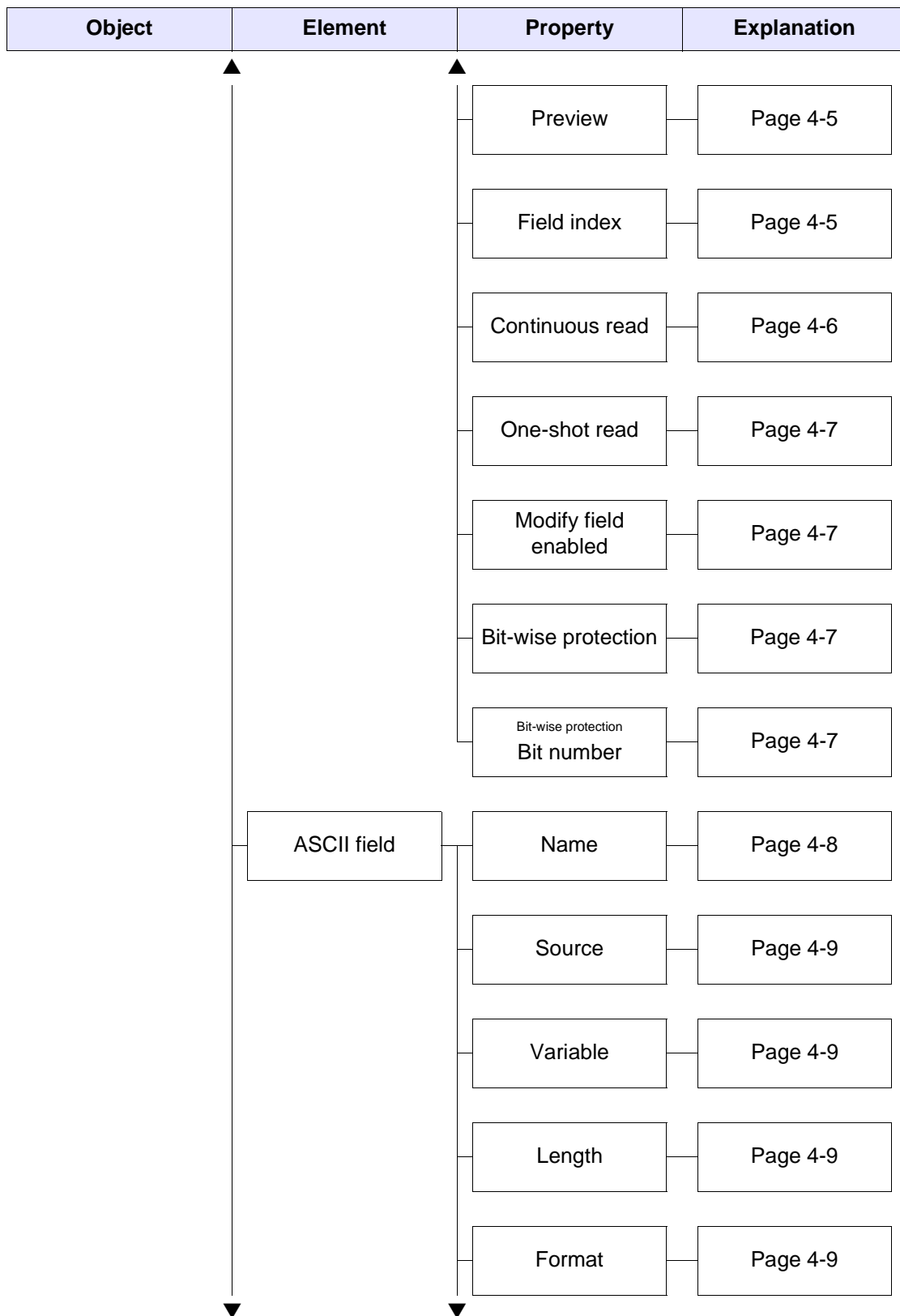


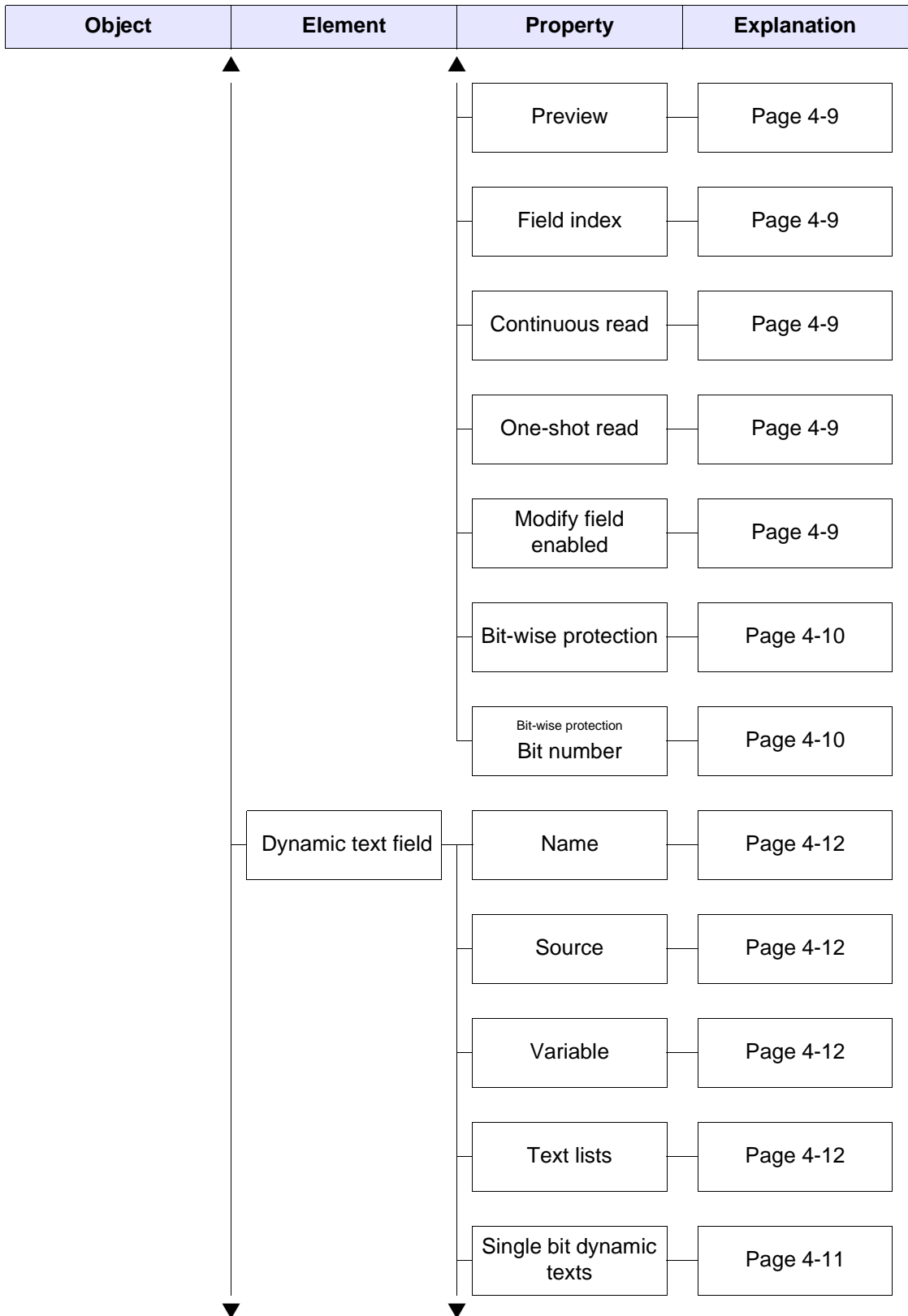
The VARIABLE enables the user to assign a data in the device connected to a field.

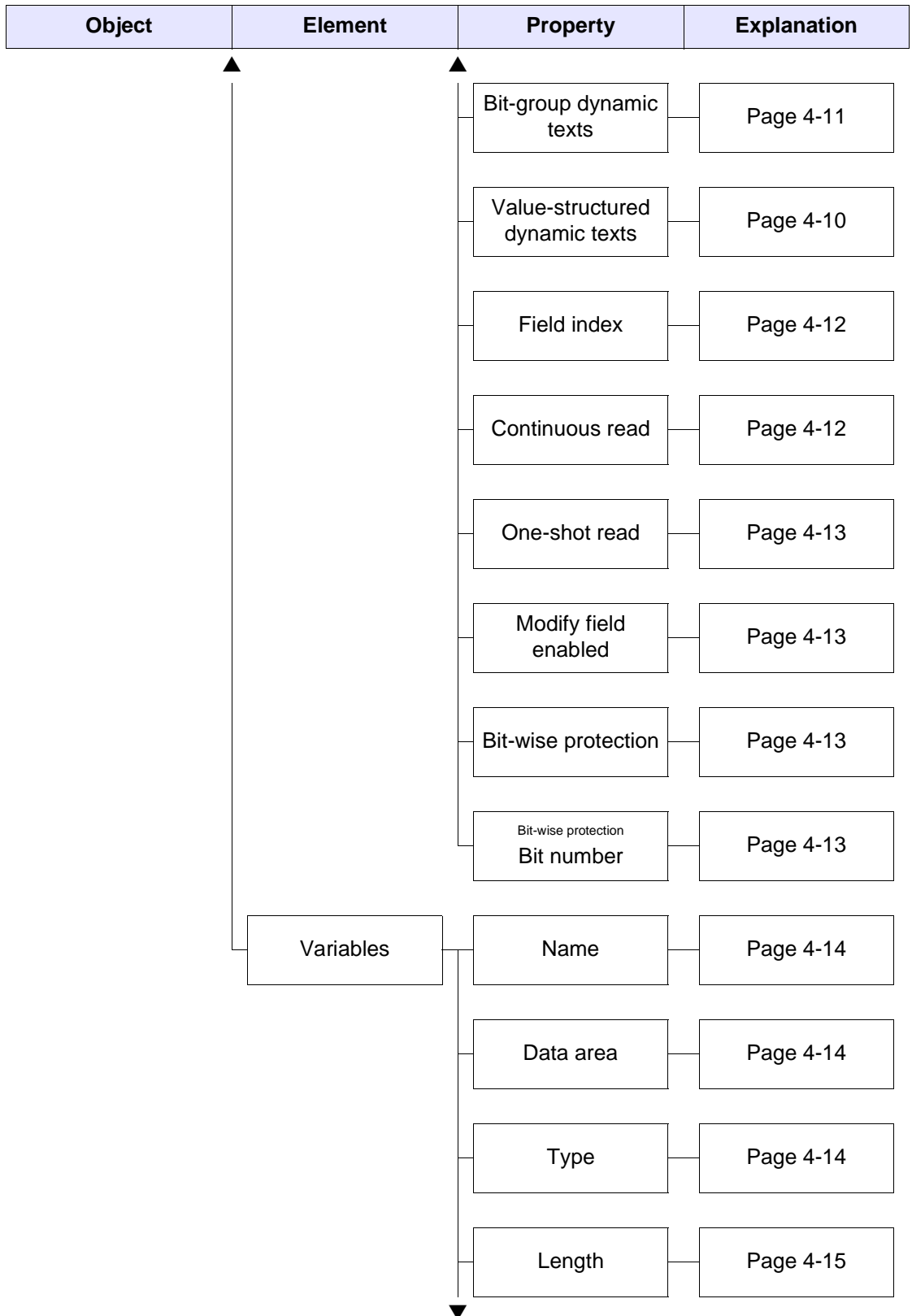
### List of functions

The following pages illustrate by means of a tree the relationship between the principal objects, the elements associated with them respectively and the properties of these elements. In addition there is a page reference to where the reader can find the most useful explanation of their meaning.

Object	Element	Property	Explanation
Page		Number	Page 4-1
		Name	Page 4-1
		Refresh time	Page 4-1
		Help	Page 4-1
	Multi-language label	Multi-language text	Page 4-2
	Numerical field	Name	Page 4-3
		Source	Page 4-3
		Variable	Page 4-3
		Leading zeros	Page 4-3
		Visible digits	Page 4-4
Truncated digits		Page 4-4	
Format		Page 4-5	







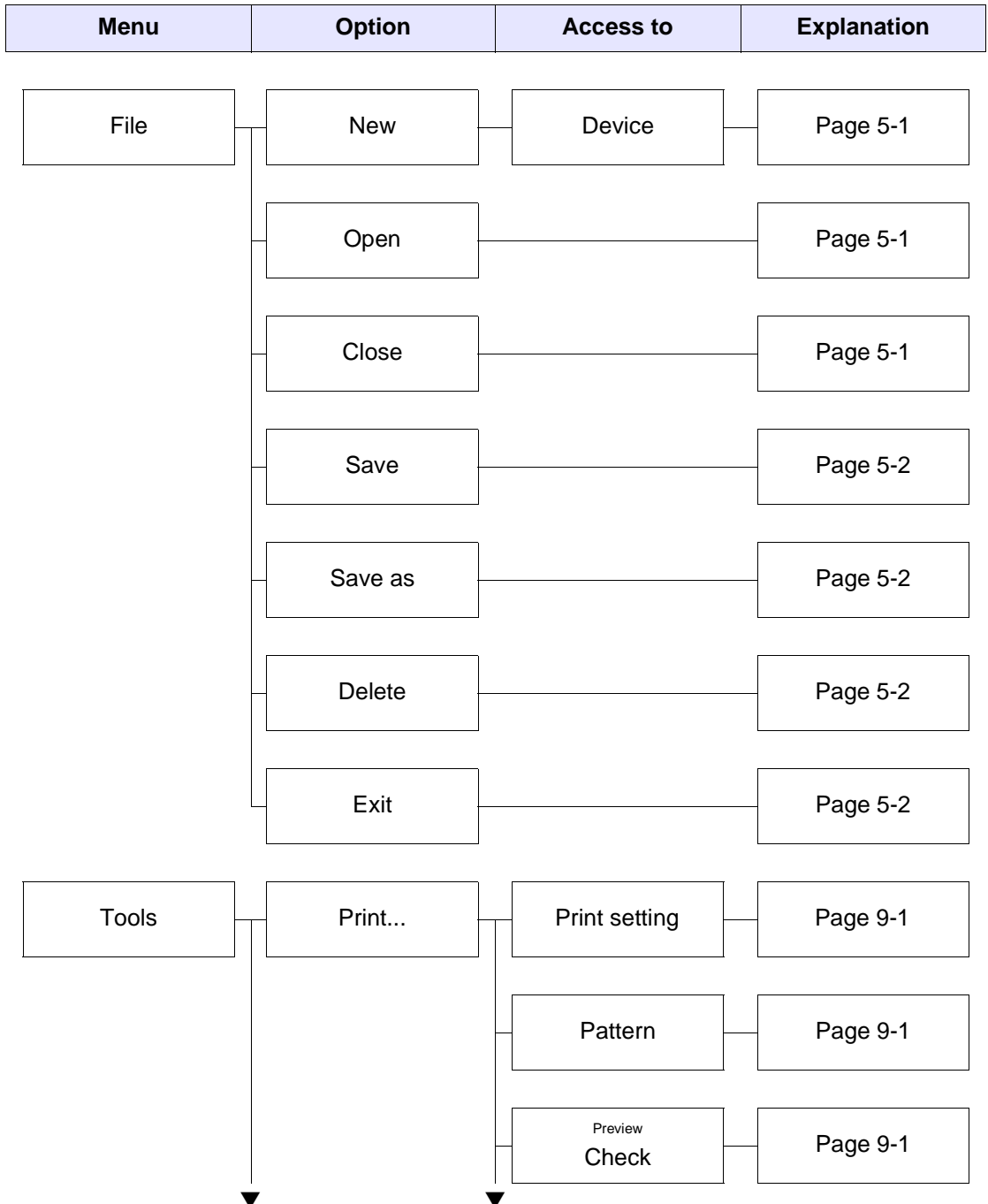
Object	Element	Property	Explanation
		Signed	Page 4-15
		BCD	Page 4-15
		Address	Page 4-15
		Input limits	Page 4-15
		Linear scale	Page 4-15
Screen sequences		Number	Page 4-18
		Name	Page 4-18
		Start/Stop sequence	Page 4-18
		Start/Stop sequence Start page	Page 4-18
		Start/Stop sequence Stop page	Page 4-18
		Random sequence	Page 4-18
		Random sequence Selected page	Page 4-18

Object	Element	Property	Explanation
Information messages		Bit number	Page 4-19
		Name	Page 4-19
		<small>Message</small> Message	Page 4-19
		<small>Message</small> Preview	Page 4-19
		<small>Help message</small> Help	Page 4-19
		<small>Help message</small> Preview	Page 4-19
Direct commands		Name	Page 4-20
		Variable	Page 4-20
		<small>Bit command</small> Bit number	Page 4-21
		<small>Value-structured command</small> Value	Page 4-21
Text lists		Name	Page 4-21
		Texts	Page 4-21


Object	Element	Property	Explanation
		▲ List of selected texts	Page 4-21

## Chapter 3 TCWIN menus

**List of menus** The following schemata show how the TCWIN menus fit one into the other and where the user can find the relevant explanations so as to interpret their meaning more easily.



Menu	Option	Access to	Explanation
		Preview View	Page 9-2
	Print Change	Name	Page 9-2
		Optional sections Available	Page 9-2
		Optional sections Selected	Page 9-2
		Global settings Cover included	Page 9-2
		Global settings Index	Page 9-2
		Global settings Info-project	Page 9-3
		Global settings Comment sections	Page 9-3
		Page settings Margins	Page 9-3
		Page settings Setting	Page 9-3
		Page settings Footer	Page 9-3
		Page settings Page numbers	Page 9-3

Menu	Option	Access to	Explanation
	Project compilation	Stop first error	Page 8-1
		No stop	Page 8-1
		Stop after N. errors	Page 8-1
		Display warnings	Page 8-1
		Outcome	Page 8-1
	Project transfer	Communication ports	Page 8-2
		Baud rate	Page 8-2
		Fw update	Page 8-2
	Font definition	Font	Page 11-1
		Dimension	Page 11-1
		Character overview	Page 11-1
		Character management	Page 11-1

Menu	Option	Access to	Explanation
Object	None		Page 6-3
	Library		Page 6-2
Fields	Label		Page 6-3
	Numerical		Page 6-3
	Ascii		Page 6-3
	Dynamic		Page 6-3
Edit	Delete		Page 6-2
	Erase all		Page 6-2
	Cut		Page 6-2
	Copy		Page 6-2
	Paste		Page 6-2
	Duplicate		Page 6-2

Menu	Option	Access to	Explanation
	Undo		Page 6-2
	Redo		Page 6-2
	Create library		Page 6-2
	Zoom up		Page 6-2
	Zoom down		Page 6-2
	Settings		Page 5-3
	Multilanguage definition		Page 5-3
Page	Internal keys definition		Page 5-3
Configuration	Page data	Order	Page 5-4
	Global internal keys definition	Keys summary	Page 5-4
	Exchange area Term./Device	Device	Page 5-9
		Settings Enabled	Page 5-9

Menu	Option	Access to	Explanation
		Settings Name	Page 5-9
		Settings Area type	Page 5-9
		Settings Start address	Page 5-9
		Settings No. words	Page 5-9
		Settings Refresh time	Page 5-10
	Project information	Created on	Page 5-10
		Edited on	Page 5-10
		Project name	Page 5-10
		Project version	Page 5-10
		Author	Page 5-10
		Company	Page 5-10
		Comment	Page 5-10

Menu	Option	Access to	Explanation
		Last date of compilation	Page 5-10
		Created with TCWIN version	Page 5-10
		Firmware version necessary	Page 5-10
	Project language	List of languages	Page 5-10
		Choice of language	Page 5-13
		Font	Page 5-13
	Project set-up	Edit mode idle timeout	Page 5-13
		Start-up sequence	Page 5-13
	Hardware	List of VTs	Page 5-13
		List of ports	Page 5-13
		Name of device	Page 5-13
		Make	Page 5-13

Menu	Option	Access to	Explanation
		▲	
		Model	Page 5-13
		Communication parameters	Page 5-13
Windows	Tile horizontal		Page 5-14
	Tile vertical		Page 5-14
TCWIN language	Languages available		Page 5-14
?	Index		Page 5-14
	Search for help on...		Page 5-14
	Information on TCWIN	System information	Page 5-14
		Check installation	Page 5-14

## Chapter 4 The functions in detail

**Page** By *Page* we mean an ensemble of data and labels items that make up the visual aspect of the screen as defined by the user and displayed on the TC.

Pages only allows the use of alphanumeric characters and symbols.

Each page has the following Attributes:

- Page number  
A progressive number identifying a page in the list.
- Name of page  
A name indicating the function of a page so that it can be easily recognized.
- Refresh time  
This is the time which elapses between one read of the information by the device and the next.
- Page help  
Supplementary information of help to the user and visible on the TC.

A list follows of the elements that can be inserted in a page:

- Multilanguage label
- Numeric field
- ASCII field
- Dynamic text field

**▲ These elements have been listed in the same order as they appear in the TCWIN menu.**

### Multilanguage label




A *Multilanguage label* is a series of characters referred to as a String, whose definition together with the textual information contained in it is in the language that has been selected.

A *Multilanguage label* field can be displayed in Reverse (With a background color different from the page background and character color different from the rest of the characters).

The field *Multilanguage Label* can have assigned to it any of the fonts in that language (see “Chapter 5 -> Project language”).

More simply, the *Multilanguage label* can be defined as a text that can be displayed automatically in the language selected in the project.

When the  F12 is pressed while in the edit phase a chart appears showing

the characters belonging to the font in use.

### Multilanguage text

All textual information contained in a project has, for each language configured (see “Chapter 5 -> Project language“) a string of characters that defines how such information should be represented.

From now on all textual information of this type is called a Multilanguage text, while the string of characters is called the *Translation*.

For each Multilanguage text you have to define a number of *Translations* equal to the number of languages configured using the project. (Below we show how to introduce these translations).

Example.

In a page dedicated to plant pressure control a multilanguage label has been defined that says what the page deals with and which functions as the title of the page.

In Italian this label corresponds to the text “PRESSIONE”, while in English it corresponds to the text “PRESSURE”.

Project language  
ITALIAN.



Project language  
ENGLISH.



### Numerical field



A *Numerical Field* is defined as one permitting the representation of a variable in a numerical format.

*Numerical Fields* are dynamic fields relating to a numerical variable.

*Numerical Fields* can be represented in binary, decimal, hexadecimal and floating point formats.

A *Numerical Field* can be displayed in Reverse (With a background color different from the page background and character color different from the rest of the characters).

Numerical fields have various parameters that have to be compiled; some of these are obligatory (**☒**), while others depend on what the user needs to have represented. The parameters are as follows:

Name:

Name defining the field. It is advisable to assign a name that helps the programmer recognize it and its contents.

Comment:

A comment can be assigned. If possible it should be the full explanation of the function of the field and its contents, but it can also be an alphanumeric character sequence.

Source:

The origin can be determined of a variable on the device or a data memory variable or another kind of variable. (See "Chapter 4 -> Variables").

Variable (**☒**):

This is the Leading zeros variable to which the field relates.

Leading zeros:

This parameter determines whether to display always the number of digits defined or not to display the significant digits if their value is equal to Example.zero.

Example.

Number of digits set = 6, value of data 100.

- |                                     |               |                 |        |
|-------------------------------------|---------------|-----------------|--------|
| <input checked="" type="checkbox"/> | Leading zeros | Display format: | 000100 |
| <input type="checkbox"/>            | Leading zeros | Display format: | 100    |

### Visible digits:

This is the number of digits to be displayed. Usually the number of digits is chosen on the basis of the value that the variable can assume the value.

Example.

If the variable reaches a value of up to 9999, just set the number of digits visible at 4; if a lower number of characters (3) is set, the left-most digit is not displayed.

Let us assume that the value is 2450; with the Visible digits parameter set at 4, the number displayed will be 2450; if, on the other hand, the Visible digits parameter is set at 3, the number displayed will be 450.

### Numerical format:

You can determine whether to display the field in binary, decimal or hexadecimal.

Example.

The value of the data in binary format is 100011. The screen will show:

Binary	->	100011
Decimal	->	35
Hexadecimal	->	23

### Truncated digits:

You can declare how many digits will not be displayed on the right side of the field (less significant digits).

Example.

The value of the data in the device is 200. Depending on the number of digits truncated, the screen will show:

0 truncated digits->	200
1 truncated digit ->	20
2 truncated digits->	2



**In the case of write data the comprehensive value of the truncated digits is sent to the device.**

---

truncated digits: 1  
 Value set in TC: 30  
 Value transferred to device: 300

Format:

This defines the way a field is represented. One or more separating characters can be inserted between the digits; all characters are accepted but only one type of character for any given format.

Example.

The value of the data is 25467; the value displayed is as follows:

Format	Display
#####	25467
###.##	254.67
#:##:##	2:54:67

Preview:

Shows how the field will appear on the terminal.

Field Index:

For the sequence followed by the cursor positioning itself on the settable data. The positioning follows an ascending order, that is, from the lowest index to the highest.

The key to the order is Index - Row - Column.

Example.

We enter 4 read/write data items from DATA 1 to DATA4, and assign indices as follows:

Data	Index
1	0
2	1
3	0
4	2

Arrange data as in figure.



The cursor positions will be in the following order:

Data 1-3      (Index 0)  
 Data 2        (Index 1)  
 Data 4        (Index 2)

Continuous read:

This parameter must be selected when you need to display the real value of a given magnitude moment by moment.

When this option is chosen, the variable assigned to the field is continuously read and the field thereby continuously updated.

**⚠ Remember that the continuous read mode means that the TC is continuously engaged in sending requests to the device attached.**

The interval between one request and another depends on the value set for the *Refresh time* (See Page 4-1) and is the same for all the fields in the page.

Example.

You need to control a plant with magnitudes that vary continually: temperature measurements, pressure measurements, numbers of products, the position of a trolley etc. To have the information displayed correctly you need to select continuous read.

Instant	Device	Display TC
t0	123	123
t1	124	124
t2	125	125

### One-shot read:

This parameter must be chosen only when there is no need to show the real value of a given measurement moment by moment.

When this option is chosen, the variable assigned to the field is read only once; the read occurs when the page containing the field assigned to this variable is displayed.

### Example.

If a page contains fields that are not conditionable by the plant process (see set-point settings, timer settings, etc.) the “One-shot read” mode must be used.

Instant	Device	Display TC
t0	1123	1123
t1	2344	1123
t2	1266	1123

Where t0 is the moment the page is first displayed.

### Modify field enabled:

This parameter determines whether the field should be read only or read/write. With a read/write field a device variable can be set using the TC.

### Bit-wise protection:

This function is valid only for settable fields, that is, for read/write fields. Using this parameter it can be established whether the field is write-protected, that is, whether or not its value can be varied using the TC. Usually this facility is used to protect important data in the device connected from the risk of overwriting it with wrong values introduced by unauthorized personnel, or to stop the value being changed as a result of a particular situation within the production process. The protection mechanism functions by setting the bit assigned to the value 0 to make it possible to change the data and to the value 1 if the data is to remain unmodifiable. It is the job of the device connected to manage the protection bits using the command area. (See “Chapter 5 -> Exchange area Terminal <-> Device“).

### Bit number:

It is possible to decide which bit will function as field protection.

Example.

Let us take 4 fields, for the sake of simplicity numbered from 1 to 4; we assign protection bit number 0 to fields 1 and 2, bit number 1 to field 3 and no protection to field 4.

Bit number	Status of bit	Field	Setting
Bit 0	1	1 - 2	not possible
Bit 1	1	3	not possible
Bit n	x	x	x

First case:  
No field can be modified

Bit number	Status of bit	Field	Setting
Bit 0	0	1 - 2	possible
Bit 1	1	3	not possible
Bit n	x	x	x

Second case:  
Fields 1 and 2 can be modified, field 3 cannot

Bit number	Status of bit	Field	Setting
Bit 0	1	1 - 2	not possible
Bit 1	0	3	possible
Bit n	x	x	x

Third case:  
Field 3 can be modified, fields 1 and 2 cannot

Field 4 is always modifiable as it is never subject to protection.

### ASCII field



An *ASCII Field* is defined as one permitting the representation of a variable in alphanumeric format.

*ASCII Fields* are dynamic fields relating to a string variable.

*ASCII Fields* can be represented only in ASCII format.

An *ASCII Field* can be displayed in Reverse (With a background color different from the page background and character color different from the rest of the characters).

The *ASCII Field* can have assigned to it any of the fonts in that language (see "Chapter 5 -> Project language").

*ASCII Fields* have various parameters that have to be compiled; some of these are obligatory (☞), while others depend on the representation needs of the user. The parameters are as follows.

Name:

See Numerical Field Page 4-3.

---

Comment:

See Numerical Field Page 4-3.

Source:

See Numerical Field Page 4-3.

Variable (⌘):

See Numerical Field Page 4-3.

Length:

The length of the string or, more simply, the number of characters in the field can be determined.

Format:

The format corresponding to the Length is shown in characters.

Example.

Length	Format
10	\$\$\$\$\$\$\$\$\$\$

Preview:

See Numerical Field Page 4-5.

Field Index:

See Numerical Field Page 4-5.

Continuous read:

See Numerical Field Page 4-6.

One-shot read:

See Numerical Field Page 4-7.

Modify field enabled:

See Numerical Field Page 4-7.

---

Bitwise protection:

See Numerical Field Page 4-7.

Bit number:

See Numerical Field Page 4-7.

### Dynamic Text Field



A *Dynamic Text field* is that field which permits the representation of binary data in a text format.

A *Dynamic Text field* is a dynamic field that relates to a numerical variable.

Text is displayed by interpreting the value of a variable or the state of one or more of its bits corresponding to a text list. (See Page 4-21).

The text list corresponding to the variable could contain more elements than the variable itself can represent.

If the value of the variable corresponding to the text list does not identify a valid text, a series of [ ! ] characters appears on the display.

A *Dynamic Text field* can be displayed in Reverse (With a background color different from the page background and character color different from the rest of the characters).

A *Dynamic Text field* can be linked to a list in three different ways:

- Assigned to the numerical value of a given variable
- Assigned to a single bit of a given variable
- Assigned to a group of bits of a given variable

Assigned to the numerical value of a given variable:

The value (in binary or BCD) of the variable assigned to the text list is used to determine which text to display. The value 0 is **inadmissible**.

Example.

Take a list of 8 texts (from Text 1 to Text 8). If the value of the variable assigned to the list is 5, then Text 5 will appear on the display; if the variable has a value of 8, then Text 8 will appear, while if the variable has a value over 8 the display will show [ !!!!! ]. In the case of a read/write Dynamic Text field, then setting Text 3 would


---

---

write the value 3 to the variable.

Assigned to a single bit of a given variable:

Only one bit of the variable assigned to the text list is used to determine which text to display. If the field is settable, updating the bit within the variable **modifies also changes the** state of the bits **not** involved.

 **It is advisable to use different variables for each dynamic text within the same page.**

Example.

Take a list of 8 texts (from Text 1 to Text 8) and relate to it bit 0 of the assigned variable; when the state of the bit is 0 the display shows Testo 1, when the state of the bit is 1 Text 2 is displayed. The texts from Text 3 to Text 8 are not handled. In the case of a read/write dynamic field, setting Text 1 would reset the bit assigned within the variable; if Text 2 is set the bit assigned within the variable is set. **All the other non-involved bits are reset !!!**

Assigned to the bit group of a given variable:

A group of bits of the variable assigned to the text is used to determine which text to display. The variable must have just one bit at 1 and all the others at 0 (with more than one bit at 1 the text assigned to the highest bit is displayed; with all bits at 0 a series of characters [ ! ] is displayed). With a settable dynamic field, the selection and successive confirmation of a text causes the assigned bit to change from status 0 to status 1 and the remaining bits of the variable to be automatically reset. This type of dynamic field can be compared to a rotating selector with a certain number of positions, where the number of positions is the number of bits selected.

Example.

Take a list of 8 texts (from Text 1 to Text 8) and assign to it the group of bits from bit 4 to bit 11 of the variable assigned. When the status of bit 4 is 1 Text 1 appears on the display, when the status of 5 is 1 Text 2 appears and so on for all the other bits of the group. If all the bits are at 0 the display shows [ !!!!!! ]. If, on the other hand, all the bits are at 1, the text corresponding to the value of the highest bit (Text 8) is displayed. In the case of a read/write dynamic field, the selection of Text 1 causes bit 4 of the variable to pass to logical status 1, while the choice of Text 3 would cause bit 7 to be set. **All other bits not involved are set at 0 !!!**

---

A *Dynamic Text field* has assigned to it various parameters that have to be compiled; some are mandatory (☞), others depend on the representation needs of the user. The parameters are as follows.

Name:

See Numerical Field Page 4-3.

Comment:

See Numerical Field Page 4-3.

Source:

See Numerical Field Page 4-3.

Variable:

See Numerical Field Page 4-3.

Text lists (☞):

It is possible to select which text list to assign to a variable.

Type:

It is possible to choose the mode of assigning a dynamic text.

First bit (☞):

Indicates the first bit assigned to the list of texts in Bit Group of Single Bit mode.

Last bit:

Indicates the last bit assigned to the text list in Bit Group mode. Within the variable, this bit must be more significant than the First Bit.

Field Index:

See Numerical Field Page 4-5.

Continuous read:

See Numerical Field Page 4-6.

---

One-shot read:

See Numerical Field Page 4-7.

Modify field enabled:

See Numerical Field Page 4-7.

bitwise protection:

See Numerical Field Page 4-7.

Bit number:

See Numerical Field Page 4-7.

## Variables

A *Variable* is an object allowing you to assign to a dynamic a data contained in the device connected.

There are two types of variable: one a string variable (generally used for exchanging textual information with the device) and a numerical variable (that can be of the type “fixed point” or “floating point” and is used for exchanging values).

String variables:

This type of variable allows the user to display a set of alphanumerical characters.

Remember that an Ascii character occupies 8 bits (1 Byte) of a register, so it is necessary to bear in mind the number of registers necessary to display the number of characters set.

Example.

A string variable of 8 characters is defined.

Given that each character is composed of 8 bits, 64 bits (8 bits x 8 characters) will be needed. If the register in the device connected is formed of only 16 bits, 4 registers will be necessary; if, on the other hand, the device contains 32 bit registers, 2 registers will be necessary.



**TCWIN will not check for overlaps of the addresses of the devices used for defining the variables.**

### Fixed Point or Entire Variables:

This type of variable, as its name implies, can be represented either with the decimal point in a pre-fixed position or without a decimal point, irrespective of the value displayed.

Example.

Let us suppose we want to insert a variable having the format of 2 digits before the decimal point and 2 digits after it: ##.##

The values are displayed as follows:

Value	Displayed form
4567	45.67
23567	35.67
1000	10.00
53	00.53

Name:

Name that defines the variable. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned which, if possible, should be a complete explanation of the function of the variable and its meaning, but it can also be an alphanumerical character sequence.



**The comment is not given in the duplication phase of the variable.**

Source:

This determines to which device to assign the variable.

Data Area:

This determines which area of the device is assigned to the variable (e.g.: Input word, Output word Etc.). The list of data areas depends on the type of device selected.


Type:

This selection determines the display mode of the data area:Byte, Word, Dword. The display mode depends on the device selected.

---

Length:

This defines the number of characters making up the string and thus determines the number of bytes of the variable

 **TCWIN does not check for congruence between the Length of the field and the Length of the string.**

With sign:


This parameter defines whether a minus sign will be shown for negative values in the display or not. Plus signs are not shown.

Example.

Four-digit variable with sign (5 digits in total): for the value 1234 the display will show 1234; for the value -1234 it will show -1234.

BCD:

Allows the content of the variable to be shown in BCD format.

Address ():

This field determines the address of the data chosen. The address type depends on the type of device connected.

Example.

Data chosen	-> Register
Address allowed by the device	-> 0-100
Address chosen	-> 25

Input limits:

Used to assign to the variable whether there will be an input limit or not; in the affirmative case the value of the minimum and maximum admissible limits must be defined.

Linear scaling:

Using this parameter a display value can be attributed that is different from the value actually contained in the device.

Example.

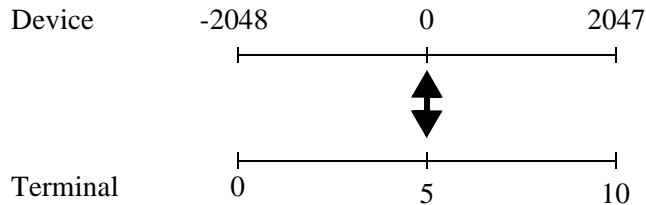
Suppose we have a variable bearing the value of an analogical input

connected to a pressure transducer: the value possible run from -2048 to 2047. It is awkward to display this value because in reality the pressure read by the manometer runs from 0 to 10Bars, making it impossible for the user to establish the correct value without carrying out conversion calculations. To avoid these calculations, just set the required parameters.

In the example used the following settings have been made:

Minimum on terminal (on display) = 0  
 Maximum on terminal (on display) = 10  
 Minimum in device = -2048  
 Maximum in device = 2047

If these parameters are inserted the terminal can calculate a linear interpolation between the values registered by the device and those to be displayed on the terminal.



It follows from the diagram above that the value 0 registered by the device will be displayed on the terminal as 5.

Linear scaling will be active in two directions if the “Input Enabled” parameter has been selected. To set the value 2 using the terminal means writing the value 819 to the device.



In addition, Linear scaling functions as a result of extrapolation: in the example in question the value 4095 read from the device will be displayed as 20 by terminal.

## Page sequences

Video pages in non-Touch Screen products must be input in a sequence to be able to be used.



**If the pages are not input in the right sequence the display must be managed by the device connected, using the command area.**

A *Page Sequence* is defined as one or more interrelated pages. The pages are grouped logically; the purpose of sequences is to be able to display topics distributed over different pages by means of   change page.

For a project to make sense there needs to be at least one page sequence defined as Start-up Sequence.

---

There are three ways of calling up a sequence: by assigning the command to a  $\square$ , by using a command from the device connected or, alternatively, as a start-up sequence when the TC is switched on.

*Page Sequences* on video can be classified as Start/Stop and as Random Sequences.

Start/Stop Sequences:

This type of sequence must have the Start and Stop Pages indicated. The page number of the start page must be lower than that of the stop page; not all the pages in the intervall between start and stop need to be present, but at least one must be for this type of sequence to make sense. Entering the sequence, the first page displayed is that identified as the Start page, then, when a Change Page request is made, the page displayed is the one with the nearest page number. The display order is cyclical, that is, when the last page is reached it starts from the first again and vice versa.

Example.

Imagine a Start/Stop Sequence 1-7, with pages 1 3 4 7 defined, and assuming the currently displayed page to be 4, when the Change Page request is made in one direction (up) page 7 will be displayed, in the orther direction (down) the page will be 3.

Random Sequences:

In this type of sequence pages can be put in any order. There must be at least one page for this type of sequence to make sense. Entering the sequence, the first page displayed is the first page in the list, irrespective of the value of the number. The page displayed when Change Page is requested is the nearest in the page list. The display order is cyclical, that is, when the last page is reached it starts from the first again and vice versa.

Example.

Imagine a Random Sequence 9-1-5-7, and assuming the currently displayed page to be 4, when the Page Up request is made page 5 will be displayed; if the Page Dn request is made the page will be 9.

*Sequence of Pages* has assigned to it various parameters that must be compiled; some are mandatory (**☞**), others depend on the representation needs of the user. The parameters are listed below.

---

Number:

This is the identifying number of the sequence.

Name:

Name that defines the sequence. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned which, if possible, should be a complete explanation of the function of the sequence and its meaning, but it can also be an alphanumeric character sequence.

Start/Stop Sequence:

Allows this type of sequence to be selected.

Random Sequence:

Allows this type of sequence to be selected.

Start Page (☹):

Active only if Start/Stop Sequence has been selected: allows the start page of the sequence to be specified.

Stop Page (☹):

Active only if Start/Stop Sequence has been selected: allows the stop page of the sequence to be specified.

Page Selected:

Active only if Random Sequence has been selected: allows the page to be inserted in the sequence to be specified.

### **Information Messages**

*Information Messages* are texts displayed when the device registers an event and communicates it to the TC using the message input area (See “Chapter 5 -> Message area:”). The TC prepares a display context appropriate for messages.

*Information Messages* can be displayed using any of the project fonts. When required, the messages can be displayed in rotation automatically (By using Command Area, see “Chapter 5 -> Command area:”), otherwise

---

the scrolling requires the appropriate  (See relevant Hardware Manual); the display order of the messages is chronological, that is, in order of arrival.

*Information Messages* have assigned to them various parameters that must be compiled; some are mandatory (☞), others depend on the representation needs of the user. The parameters are listed below.

Bit number (☞):

Indicates the bit to which the informational message should be related. (The message is activated when the bit specified is put at status 1).

(See “Chapter 5 -> Exchange area Terminal <-> Device“).

Name:

Name defining the message. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned which, if possible, should be a complete explanation of the function of the sequence and its meaning, but it can also be an alphanumerical character sequence.

Message (☞):

The message to be displayed is edited.

Preview:

Shows what the message will look like on the display of the TC.

Help Message:

Use this to edit the text of the help page.

Preview:

Shows what the help page will look like on the display of the TC.

## Direct Commands

Using a *Direct Command* the value of a variable can be changed the moment the  assigned to the direct command is pressed. A project can have any number *Direct Commands* configured and these can be assigned to a button.

A *Direct Command* is always assigned to a numerical variable. *Direct Commands* are classifiable into Bit-structured Direct Commands and Value-structured Direct Commands.

Bit-structured Direct Commands:

Bit-structured Direct Commands allow you to change a single bit of a numerical variable.

Value-structured Direct Commands:

Value-structured commands allow you to change the value of a numerical variable by means of forcing a constant. Value-structured direct commands affect the entire value of the assigned numerical variable.

Example.

Value of variable 120, value specified 45. After pressing the  $\square$  the value 45 is transferred to the device.

*Direct commands* have assigned to them various parameters that must be compiled; some are mandatory (**☞**), others depend on the representation needs of the user. The parameters are listed below.

Name:

Name defining the direct command. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned which, if possible, should be a complete explanation of the function of the direct command and its meaning, but it can also be an alphanumerical character sequence.

Variable (**☞**):

This is the variable on which the direct command operates.

Bit:

If set, the direct command is bit-structured.

---

---

Bit number (**☞**):

Determines the bit number of the numerical variable specified on which the direct command operates.

Value:

If set, the direct command is value-structured.

Value:

The value of the operand is assigned.

## **Text Lists**

The *Text Lists* function is used in the project to make a symbolic text correspond to the value of a numerical variable.

The text lists serve to construct the *Dynamic Texts*. (See Page 4-10)

Each text list contains status texts that, in general, are used to indicate the operational status of a plant or a component of a plant.

A text list must contain at least two texts. A text may be composed of a series of spaces. A text may appear in any project font.

*Lists of Texts* have assigned to them various parameters that must be compiled; some are mandatory (**☞**), others depend on the representation needs of the user. The parameters are listed below.

Name:

Name defining the text list. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned which, if possible, should be a complete explanation of the function of the text list and its meaning, but it can also be an alphanumerical character sequence.

Texts:

Shows the texts contained in the list.

List of texts selected (**☞**):


Used to edit the texts to be put on the list.


---




## Chapter 5 The menus in detail

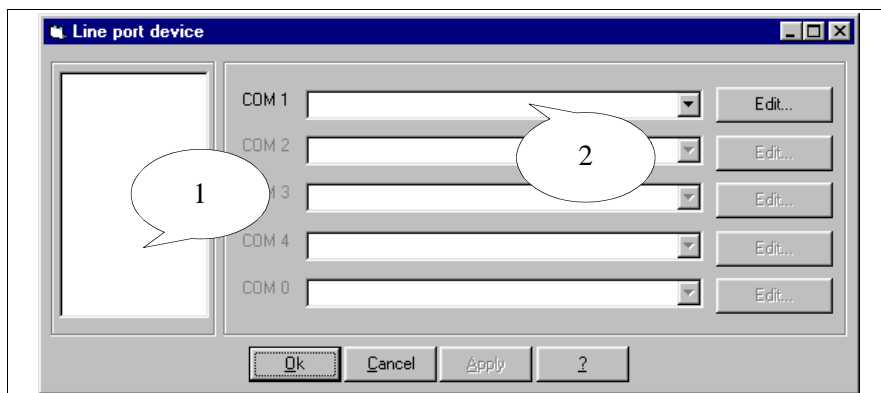
### File *New...*

Use this to create a new project. By choosing this  you automatically open the following mask:

1) A list appears of all the TCs that can be programmed.  
By clicking on TC to be used you can activate  Edit.

Click on  Edit to call up the mask below.


3) You are shown the device connected to the TC terminal.

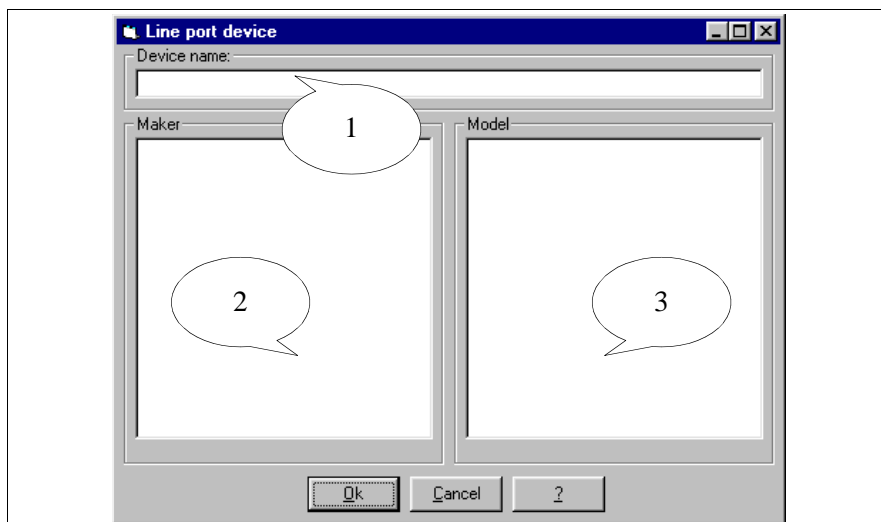


1) The name assigned to the device connected is displayed.

2) All the devices that can be connected to the selected operator terminal are listed.

3) A list appears of the models of the selected device.

Click on  Ok.



### *Open...*

Used to call up an existing project.

### *Close*

Used to abandon the project currently displayed.

***Save***

Used to save the currently displayed project on disk.

***Save as...***

Used to save under a different name the program currently being worked on.

***Delete...***

Used to delete a given project from the archive.



**The project is deleted DEFINITELY.**

***Exit***

Used to abandon TCWIN.

**Tools*****Print...***

See “Chapter 9 -> Creating and printing documentation“

***Compile project***

See “Chapter 8 -> Compiling and transferring a project“

***Download project***

See “Chapter 8 -> Compiling and transferring a project“

***Font editor***

See “Chapter 11 -> Defining the fonts“

**Object**

The content of this menu is explained on “Chapter 6 -> Meaning of menu icons“, where the corresponding TCWIN icons are also shown.

**Fields**

The content of this menu is explained on “Chapter 6 -> Meaning of menu icons“, where the corresponding TCWIN icons are also shown.


---

**Edit*****Setting***




Used to activate changes in the properties of a selected element.


***Multi-language definition***

Used to edit the translations of a selected multi-language label.

The other  of this menu are explained on “Chapter 6 -> Meaning of menu icons“, where the corresponding TCWIN icons are also shown.


**Page*****Keys definition***

Used to define the link between an F  and a function, which is will be valid only for the page being displayed. This association has priority over a global reconfiguration. To define the function double-click on the desired  displayed in the list or else by clicking directly on the  of the page of TCWIN displayed in the foreground.

Functions assignable to F :

The following functions are not assignable to any device variable but perform predefined tasks.

None:

No local function attributed, therefore use global  configuration.

Disable key:

Disables the .

Sequence:

Used to call up the assigned sequence.

Internal Command: Change language

Use this to change the current project language, substituting one of those declared. The new language does not remain active even after a new start-up.

Internal Command: Show project information

This displays the project information page.

**Internal Command: Show sequences directory**

This displays a system page with a list of all the sequences programmed. From this page can be selected the sequence to be called up.

**Internal Command: Quit project**

Allows you to exit from project and then enter the programming page.

**Real time direct command:**

Puts the bit at 1 as long as the  $\square$  is depressed.

**Flip-flop direct command:**

Inverts the status of a bit (from 1 -> 0 and vice versa) each time the  $\square$  key is pressed.

**Value-structured direct command:**

Changes the value of a given variable.

**Configure*****Pages data***

Used to display the cross-reference between fields and pages. The type of order can be chosen: by page or by data. While the first lists all the pages and shows which variable is contained in them, the second lists all the variables and shows which page they are contained in.

***Global keys definition***

Used to define the association between F  $\square$  and function, that will be valid for the entire project irrespective of the page being displayed. This correspondence remains valid so long as the  $\square\square$  are not reconfigured locally page by page, in which case the priority passes to the local reconfiguration. To define the function double-click on the desired  $\square$  shown in the list. (See also "Chapter 5 -> Keys definition").

***Exchange area Terminal <-> Device***

The device exchanges information with the TC by means of variables used separately in different pages or by using the *Exchange Areas*.

The *Exchange areas* are structures containing information relate to the

---

device connected. These areas are exchanged periodically with the device. Conceptually these areas can be divided into read areas and write areas. The read areas are updated with the expiry of a time set by the programmer and are divided into a *Message Area* and a *Command Area*. The write area updates the device connected only when there is a change of status of variables in the TC; this area is called the *Status Area* and is divided into a Terminal status area, an  status area.

**⚠ The detailed significance of the various words and commands depend on the type of TC being used; thus for information not given here, see relevant Hardware manual.**

Message area:

This area is used by the TC to acquire events occurring in the plant and detected by the device (e.g. a photocell has been intercepted, a thermal protection device has intervened).

The message area can be assigned directly either to the device's input area or its data area.

It is the message area that defines the registers for controlling the *Information* (See "Chapter 4 -> Information Messages"); the length in words of the areas depends on the TC being used.

Command area:

This area is used by the device connected to make the TC carry out certain functions and/or commands. This area is composed of 4 fixed words (numbered from 0 to 3). Word 0 defines the command that the TC has to carry out, words 1 to 3 serve as parameter words. The functions and/or commands are contained in the TC and are identified by a numerical code and by parameters.

WORD NUMBER	NAME OF WORD
0	COMAND
1	PARAMETER 1
2	PARAMETER 2
3	PARAMETER 3

To make the TC carry out an action, the device must first prepare the parameters related to the action by writing them in the appropriate word, then write the code for the action in the corresponding word.

**⚠ The parameter words must be written first to ensure that there are no losses of information.**

Seeing that the TC is aware of having to carry out an action when it

finds a value other than 0 in word 0, writing the parameters after the command you risk a situation where the TC reads the 4 words before the device has read all the parameters. The consequence would be that data is lost or a wrong action is carried out.

At this point, finding a value other than 0 in the command word, the TC realizes that the device is making a request and so reads the 4 words, then it interprets the command, carries it out and sets the command word back to 0.

The device must interpret this resetting as meaning that another command can be sent.

The status area of the terminal is used to monitor what is happening between the TC and the device.

An example.

You want to set the language in the TC as Italian (2). First of all you need to determine the command to use on the basis of the list of commands set out in the relevant Hardware Manual: this is command no. 07.

### COMMAND “07”: SET CURRENT LANGUAGE

The command SET CURRENT LANGUAGE has 1 parameter and produces an updating of the TC’s language in line with the data sent by the device. The command does not need a response. The format of the command sent by the device is as follows:

WORD NUMBER	NAME OF WORD
0	Sets current language
1	Language identifier
2	Not used
3	Not used

Where:

Language identifier = The number of the new current language (It depends on the way the project is set. See Page 5-10).

---

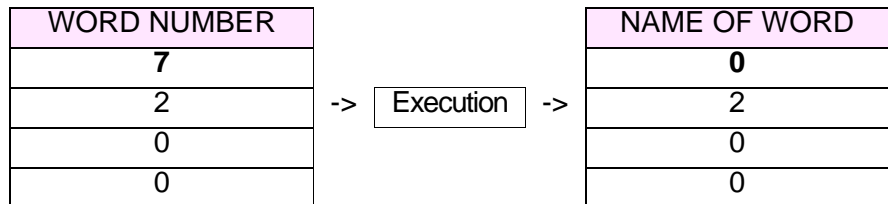
First of all, the necessary parameters are set:

WORD NUMBER	NAME OF WORD
0	0
1	2
2	0
3	0

After entering the parameters we write the command code:

WORD NUMBER	NAME OF WORD
0	7
1	2
2	0
3	0

The TC reads the words, executes the command and puts the command word back to 0 to indicate to the device that the operation has been completed.



Status area:

This area is used by the TC to inform the device of any change that has occurred in the operational status of the TC or in response to a request command coming from the device connected. The TC writes each area the moment there is any change in the information contained in it.

Status area of terminal:

This area consists of 4 fixed words (numbered 0 to 3). Word 0 is coded in binary and defines the status of the TC; word 1 is not used; word 2 contains the page number that appears on the display if the active context is that of Project pages; word 3 is in binary and contains the active context of the TC in the event that Project pages is not active.

WORD NUMBER	NAME OF WORD
0	STATUS WORD
1	N.U.
2	PAGE IDENTIFICATION
3	FIELD IDENTIFIER

Example.

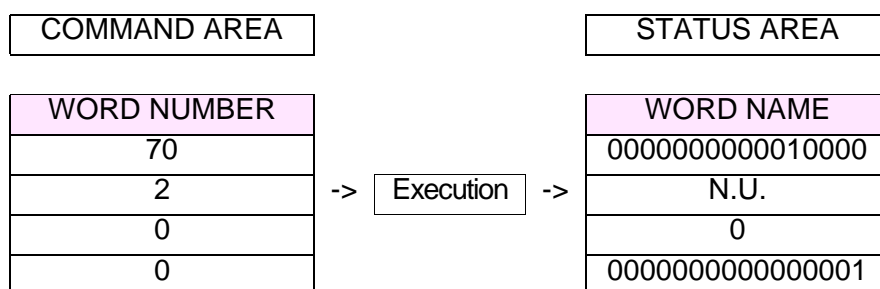
As in the previous case, the user wants to set the TC language as Italian (2). First of all, the user must determine which command to use: this command is 07.

See "Chapter 5 -> COMMAND "07": SET CURRENT LANGUAGE"

Let us suppose that the device writes the wrong command code (for example 70) in the command word.

WORD NUMBER	NAME OF WORD
0	70
1	2
2	0
3	0

The TC reads the words, realizes that the command code is wrong and sets the corresponding words in the status area in the following manner.



Bit 4 of the STATUS WORD is put at 1 to indicate that the command has not been carried out: the device, interpreting this diagnostic data, must conclude that the previous command has not been successfully executed and that the command must be repeated sending the correct code.

Status area for keys:

This area is composed of 1 words. This word is in binary code and define the status of the  pressed. The area is exchanged with the device when a  is pressed.

WORD NUMBER	NAME OF WORD
0	OPERATIVE KEYS STATUS

Each *Exchange area* has associated to it several parameters that must be compiled; some of them are mandatory, others depend on the user's needs. They are as follows:

Device:

Displays which device is connected.

Settings:

Displays the list of the exchange area registers and makes it possible to edit and/or enter them.

Enabled:

Activates the data exchange. (Compiling the registers does not automatically activate the data exchange).

Name:

The name is that which defines the area. It is advisable to assign a name that makes it easier for the programmer to recognize and understand its significance.

Type of data area:

Used to choose which area of the list is to be configured.

Start address:

Defines the address and the type of data area starting from which the words to be exchanged are mapped. The types of data areas available, Registers, Inputs, Outputs etc. depend on the type of device selected.

Word length:

Used to set the number of words to use for the areas to be configured. (The words are 16 bits long, registers are 32 bits and use 2 Words.)

With some areas you cannot modify the number of words which is fixed at 4.


Refresh time:

This determines the time that must elapse between one update of the exchange information and another.

### ***Project information***

Used to input project-related information that can be printed and/or displayed on screen.

- Created on
- Modified on
- Date of last compilation
- Created using TCWIN version
- Version of firmware necessary

The  listed below can be set by the user:

- Name of project
- Version of project
- Author
- Company
- Comment

### ***Project language***

As already mentioned, TCWIN makes it possible to create multilanguage projects. This means that with an appropriate command the display of a given project on screen will change in accordance with the language chosen. The language change occurs provided that the user has defined the languages into which the project must be translated.

The first language in the list is taken as the mother tongue (M.-t.), that is the language in which things are normally displayed. If the project languages are not set, the project is treated as monolingual and it will not be possible to assign any translation and the display will always be in the mother tongue (M.-t.).

The various languages can be displayed using different fonts, that is with different graphical attributes from the characters themselves.

TCWIN, depending on the type of TC being used, creates a list of fonts available with it. Up to 4 different fonts can be chosen that then become the active project fonts to be used for editing texts in various languages. If no font is selected, the font used is that set by the system.

---

Example:

Let us suppose that our intention is to create a project in three languages (Language 1 to Language 3) and that the TC in use allows you to use the 10 fonts listed in the table below (the fonts used in the example have no relation to those really in existence).

Table 5.1: Font.

Fonts available	Display
System font	ABCD abcd 1234
Font 1	<b>ABCD abcd 1234</b>
Font 2	ABCD abcd 1234
Font 3	<i>ABCD abcd 1234</i>
Font 4	<i>ABCD abcd 1234</i>
Font 5	<b>ABCD abcd 1234</b>
Font 6	ABXΔ αβχδ 1234
Font 7	<b>ABCD abcd 1234</b>
Font 8	ABCD abcd 1234
Font 9	<b>ABXΔ αβχδ 1234</b>
Font 10	<i>ABCD abcd 1234</i>

We now assign 4 project fonts

Fonts available		Project fonts
Font 1	->	Font 1
Font 2		
Font 3		
Font 4		
Font 5		
Font 6	->	Font 6
Font 7	->	Font 7
Font 8		
Font 9		
Font 10	->	Font 10

We now assign project fonts to the various languages using a different order each time:

Lang. 1 (M.-t.)	<table border="1"> <tr><td>Project font</td></tr> <tr><td>Font 1</td></tr> <tr><td>Font 6</td></tr> <tr><td>Font 7</td></tr> <tr><td>Font 10</td></tr> </table>	Project font	Font 1	Font 6	Font 7	Font 10	->	<table border="1"> <tr><td>Language font</td></tr> <tr><td>Font 1</td></tr> <tr><td>Font 6</td></tr> <tr><td>Font 7</td></tr> <tr><td>Font 10</td></tr> </table>	Language font	Font 1	Font 6	Font 7	Font 10
Project font													
Font 1													
Font 6													
Font 7													
Font 10													
Language font													
Font 1													
Font 6													
Font 7													
Font 10													
Language 2	<table border="1"> <tr><td>Project font</td></tr> <tr><td>Font 1</td></tr> <tr><td>Font 6</td></tr> <tr><td>Font 7</td></tr> <tr><td>Font 10</td></tr> </table>	Project font	Font 1	Font 6	Font 7	Font 10	->	<table border="1"> <tr><td>Language font</td></tr> <tr><td>Font 7</td></tr> <tr><td>Font 10</td></tr> <tr><td>Font 1</td></tr> <tr><td>Font 6</td></tr> </table>	Language font	Font 7	Font 10	Font 1	Font 6
Project font													
Font 1													
Font 6													
Font 7													
Font 10													
Language font													
Font 7													
Font 10													
Font 1													
Font 6													
Language 3	<table border="1"> <tr><td>Project font</td></tr> <tr><td>Font 1</td></tr> <tr><td>Font 6</td></tr> <tr><td>Font 7</td></tr> <tr><td>Font 10</td></tr> </table>	Project font	Font 1	Font 6	Font 7	Font 10	->	<table border="1"> <tr><td>Language font</td></tr> <tr><td>Font 1</td></tr> <tr><td>Font 10</td></tr> <tr><td>Font 6</td></tr> <tr><td></td></tr> </table>	Language font	Font 1	Font 10	Font 6	
Project font													
Font 1													
Font 6													
Font 7													
Font 10													
Language font													
Font 1													
Font 10													
Font 6													

Note that the order of the font varies with the language: this is very important, because the association between the fonts and the various languages is related to its position in the list.

1° Font Lang. 1 -> Font 1 ABCD abcd 1234	Lang. 2 <b>ABCD abcd 1234</b>	Lang. 3 ABCD abcd 1234
2° Font Lang. 1 -> Font 6 ABXΔ αβχδ 1234	Lang. 2 <b>ABCD abcd 1234</b>	Lang. 3 <b>ABCD abcd 1234</b>
3° Font Lang. 1 -> Font 7 <b>ABCD abcd 1234</b>	Lang. 2 ABCD abcd 1234	Lang. 3 ABXΔ αβχδ 1234
4° Font Lang. 1 -> Font 10 <b>ABCD abcd 1234</b>	Lang. 2 ABXΔ αβχδ 1234	Lang. 3 ABCD abcd 1234 < System font

The mother tongue can be changed at any time by moving one of the languages to the head of the list.

Lang. (M.-t.)	<-	Lang. 1
		Lang. 2
		Lang. 3
Lang. (M.-t.)	<-	Lang. 3
		Lang. 2
		Lang. 1

The languages have associated to them various parameters that must be compiled; some are mandatory, others depend on the user's needs. The parameters are those listed below.

List of languages:

Displays the languages in which the project can be displayed. the first on the list, as already mentioned, is considered to be the mother tongue.

Language selection:

Used to insert a language in the list.

Font:

Used to assign the font to the language to be displayed.

### ***Project settings***

The general project settings listed below can be entered in this menu.

Edit-mode idle timeout:

This indicates the time the terminal will remain idle in edit-mode before returning to display mode.

Start up sequence:

This indicates the first sequence to be displayed on switching on.

### ***Device***

Used to change the type of TC being used, any related parameters and the type of device connected to the TC. This operation can be carried out at any

stage of the project. (See “Chapter 5 -> New...“.)



**Changing the type of TC or the type of device connected can lead to a loss of data and graphical information.**

## Windows

### *Horizontal arrangement*

With this the active windows can be displayed horizontally.

### *Vertical arrangement*

With this the active windows can be displayed vertically.

## TCWIN language

### *Languages available*

With this you see a mask containing the various languages in which TCWIN can be displayed.

## ?

### *Index*

With this you can call up the index of all the topics dealt with in the Help on Line.

### *Search for help on....*

With this you can call up a mask for looking for a particular topic.

### *Information about TCWIN...*

With this you can call up a mask where you can get *System information* e *Installation control*, the former allowing you to have information on the machine where TCWIN is installed, the second allowing you to get information on TCWIN installation.

---

---

## Chapter 6 Using TCWIN

### Terminology used

We offer below an explanation of the operational terms used in the document.

- Click:** Press a key of the mouse once and then release it. (If not otherwise stated, this is the left key of the mouse.)
- Double click:** Press the left key of the mouse twice in rapid succession. (If not otherwise stated, this is the left key of the mouse)
- Select:** Move the pointer of the mouse so that it is over an object and click.
- Drag:** Select an object, press the left key of the mouse, keep it pressed down and move the object to the point desired, then release the key.

### Forms assumed by the mouse pointer

The pointer of the mouse assumes various forms which depend on the operation being carried out.



Normal form of the pointer.



Pointer on "hold": operation still being carried out.



Operation in background: more than one operation going on at the same time.

















### Meaning of menu icons

The table shows all the menu icons together with their respective meanings.

Some of the functions listed below only affect a field that has been selected.








---

Table 6.1: List of TCWIN icons, their respective menus and meanings (Part 1 of 2)

Tools Bar	Pulldown Menu	Action	Selection Required
	<i>File &gt; New</i>	Creates a new project.	--
	<i>File &gt; Open</i>	Opens an existing project.	--
	<i>File &gt; Save</i>	Saves a project on disk.	--
	<i>Tools &gt; Print...</i>	Prints the project.	--
	<i>Edit &gt; Cut</i>	Saves a selection in the notes and deletes the object selected from the page.	Yes
	<i>Edit &gt; Copy</i>	Saves a selection in the notes.	Yes
	<i>Edit &gt; Paste</i>	Inserts a selection from the notes into the page .	No
	<i>Edit &gt; Delete</i>	Deletes the object selected from the page.	Yes
	<i>Edit &gt; Erase all</i>	Deletes all the objects in the page.	No
	<i>Edit &gt; Duplicate</i>	Duplicates the object selected.	Yes
	<i>Edit &gt; Build library</i>	Saves a selection in a file on disk.	Yes
	<i>Object &gt; library</i>	Inserts a library in the page.	No
	<i>Edit &gt; Undo</i>	Each keystroke undoes the last action performed.	No
	<i>Edit &gt; Redo</i>	Each keystroke restores the action undone.	No
	<i>Edit &gt; Zoom up</i>	Increases the degree of enlargement of the page displayed.	No
	<i>Edit &gt; Zoom down</i>	Decreases the degree of enlargement of the page displayed.	No

-- Option not valid for this menu

Table 6.1: List of TCWIN icons, their respective menus and meanings (Part 2 of 2)

Tools Bar	Pulldown Menu	Action	Selection Required
	? > <i>Contents</i>	Calls up the help on line.	--
	? > <i>Search for help on...</i>	Activates the "search for help" function of the help on line.	--
	<i>Object &gt; None</i>	Puts the pointer in readiness mode.	--
	<i>Fields &gt; Label</i>	Allows the insertion of a multilanguage label.	--
	<i>Fields &gt; Numeric</i>	Allows the insertion of a numerical field.	--
	<i>Fields &gt; Ascii</i>	Allows the insertion of an ASCII field.	--
	<i>Fields &gt; Dynamic</i>	Allows the insertion of a dynamic field.	--

-- Option not valid for this menu



---

## Creating a project with TCWIN

Before seeing how to create a project, it is worth spending a few moments to understand what creating a project means and what the necessary elements are.

The first thing is to understand which functions the TC puts at the disposal of the user. At this point it is not necessary to know in depth how this works: it is enough to know that these function exist.

It is extremely important is to exploit the potential of the panel to the utmost, trying to avoid using the device to manage what the panel manages automatically (messages, Start up sequence, etc.).

Although this seems obvious, often for a variety of reasons it is forgotten and what happens is that the operation of the TC is adapted to the project to be created, which is the worst thing that can happen.

A project must be structured and adapted to fit the TC.

This being clear, we can move on to creating the project. It is necessary to establish the graphical structure of the project, by which we mean the look of the pages and their contents, to know which variables to use and which messages, which data exchange area to use (if necessary). It is necessary too to have defined the sequences and, in general, to have thought of all the elements to be included in the project.

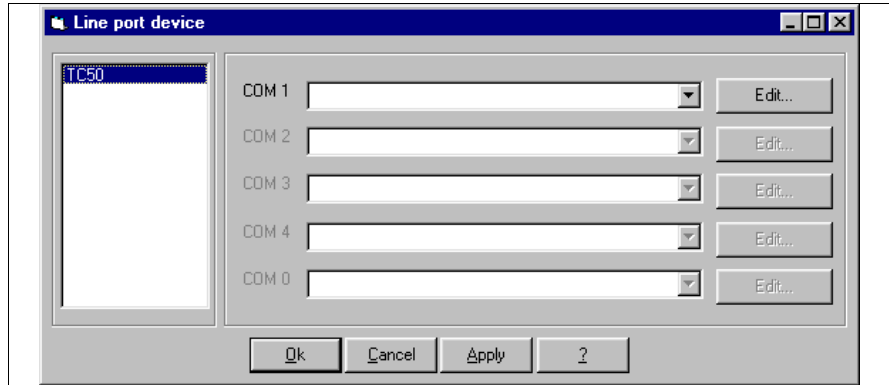
Let is imagine that we wish to create a supervisory project for a wine producer. Using the example of this plant, we will see how to control the temperature of bottling plant. The plant will be monitored by means of messages.

---

## Creating the project

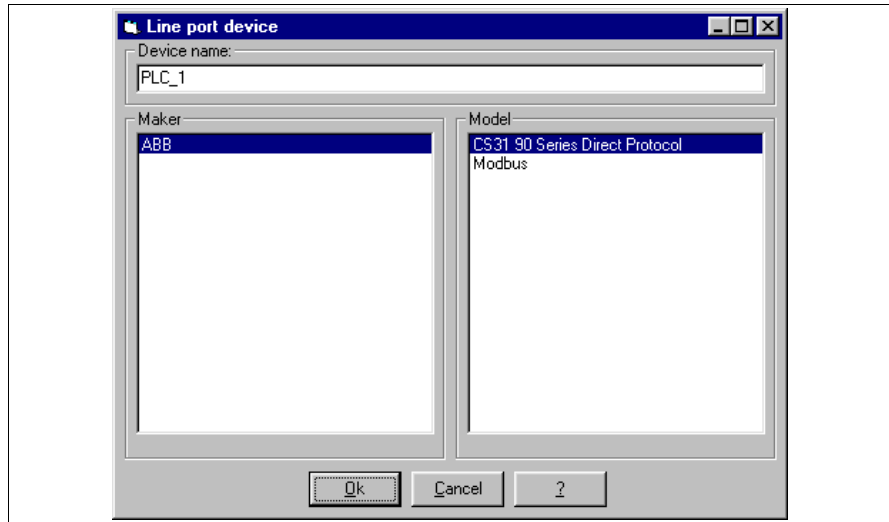
Click on **File > New** (See “Chapter 5 -> New...”)

Select TC50, select PLC and click on Edit



Select the **ABB** in the ABB, CS31 list.

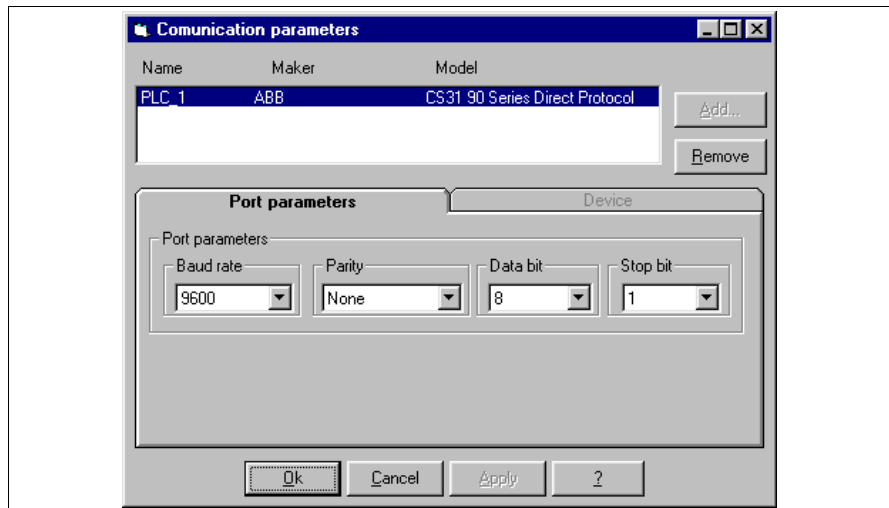
Click on **Ok**.

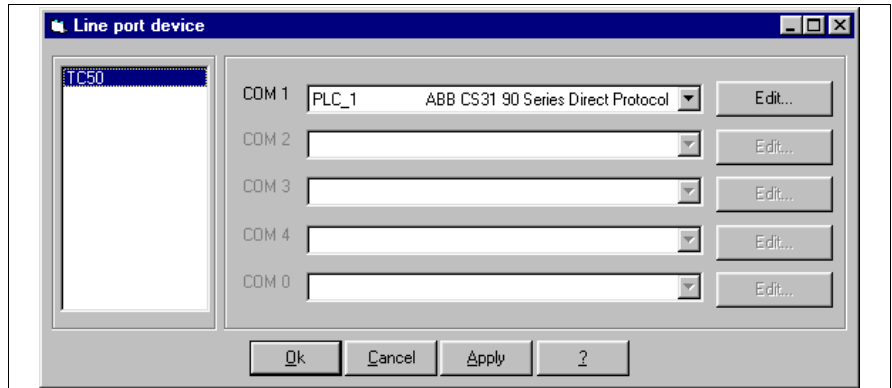



Set communication parameters for serial port of the TC.

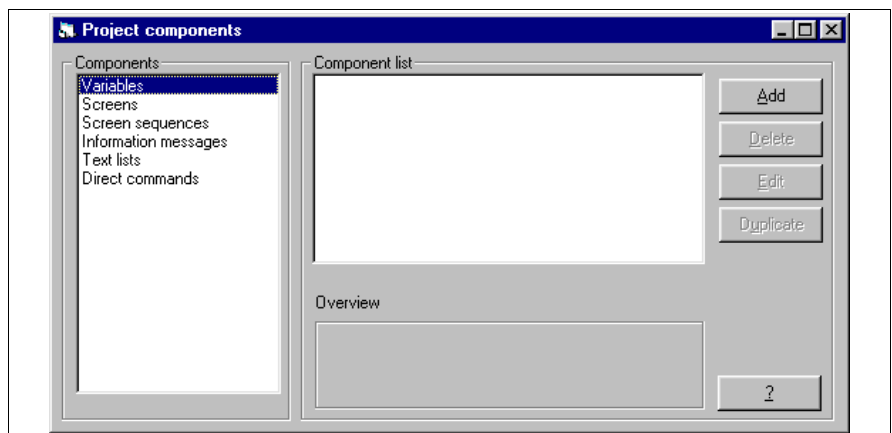
**These must be the same as for the device.**

Click on **Ok**.





Click on  Ok.



The project has been opened.

**Project  
information**

Click on *Configure > Project information* (See “Chapter 5 -> Project information“)

**Project information**

Author

Created on 03/05/99 17.06.54

Modified on 03/05/99 17.08.39

About project

Name SPUMCS0 Version 0.0

Author Ballabio R. Company

Comment Bottling plant


About compilation

Last compilation date 03/05/99 17.08.50

Created by TCWIN version 1.0114

Firmware version needed 0

Ok Cancel Apply ?

Compile the  required.

Click on OK.

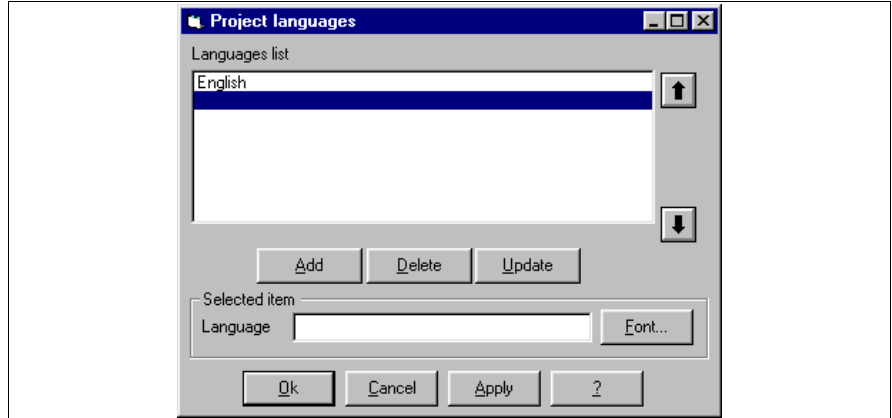
## Setting project languages

Define the languages for displaying the project on the TC panel; in this case the languages chosen are Italian (mother tongue) and English.

Click on *Configure > Project languages* (See “Chapter 5 -> Project language“)

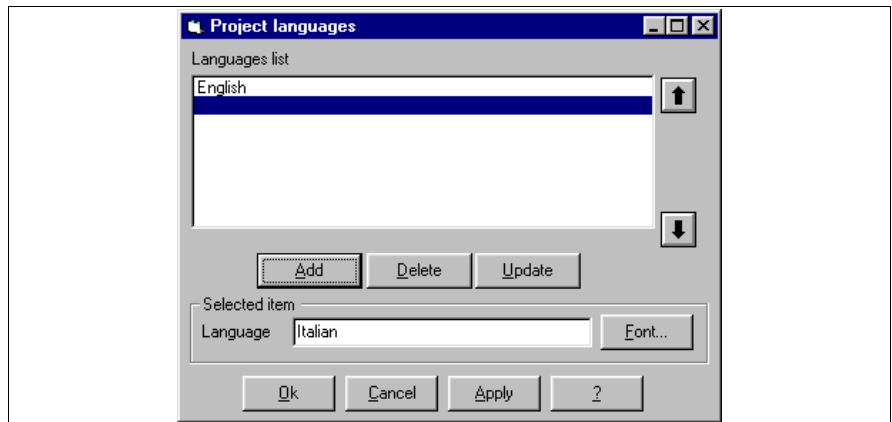
Select the box Language, and insert the mother tongue of the project; digit Italian.

Click on  Add.



Insert the language of translation; digit English.

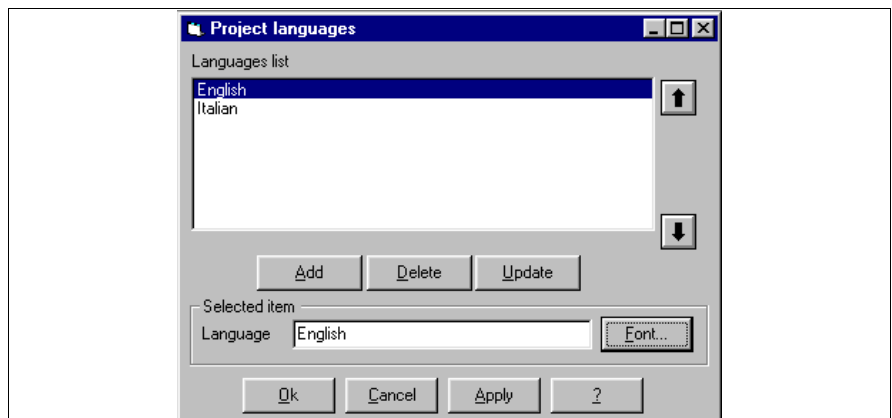
Click on  Add.




At this point the fonts of the languages are set.

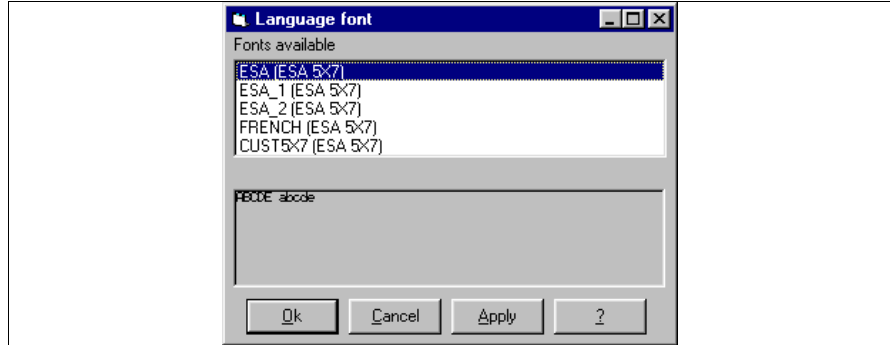
Select  English.

Click on  Font...




Select   
ESA(ESA5X7) in the  
Fonts available list.

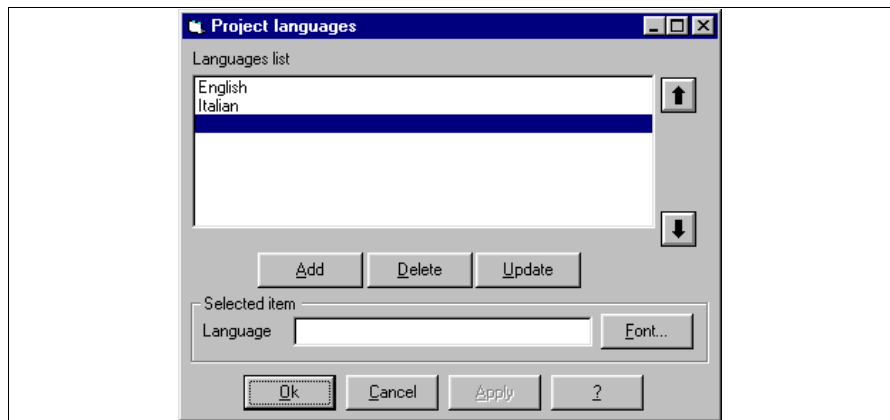
Click on  Ok.



Repeat this procedure selecting  ESA\_1 (ESA5X7). Click on the  Ok.

To activate the settings  
for the language.

Click on  Ok.




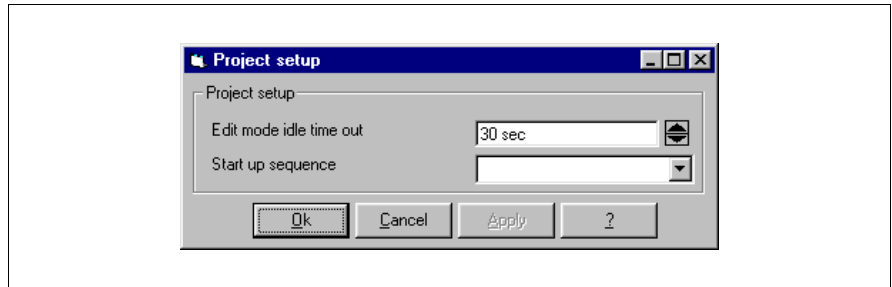
The project now contains information on the language. From now on all the masks containing comments or editable texts will be requested in translation.

**Project setup** Click on *Configure > Project setup* (See “Chapter 5 -> Project settings“)

*Fix the Edit mode idle timeout at 30 secs; the Startup sequence will be set at 1, but first it must be generated, so we will return later to the subject of this box.*

*Compile as illustrated.*

*Click on  Ok.*

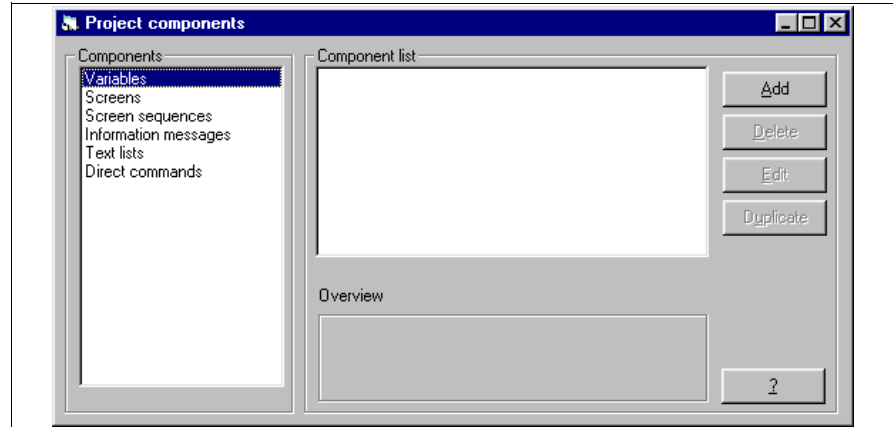


In this way all parts of the project is parametrized; now we can start to insert the elements that make up the project.

## Inserting variables

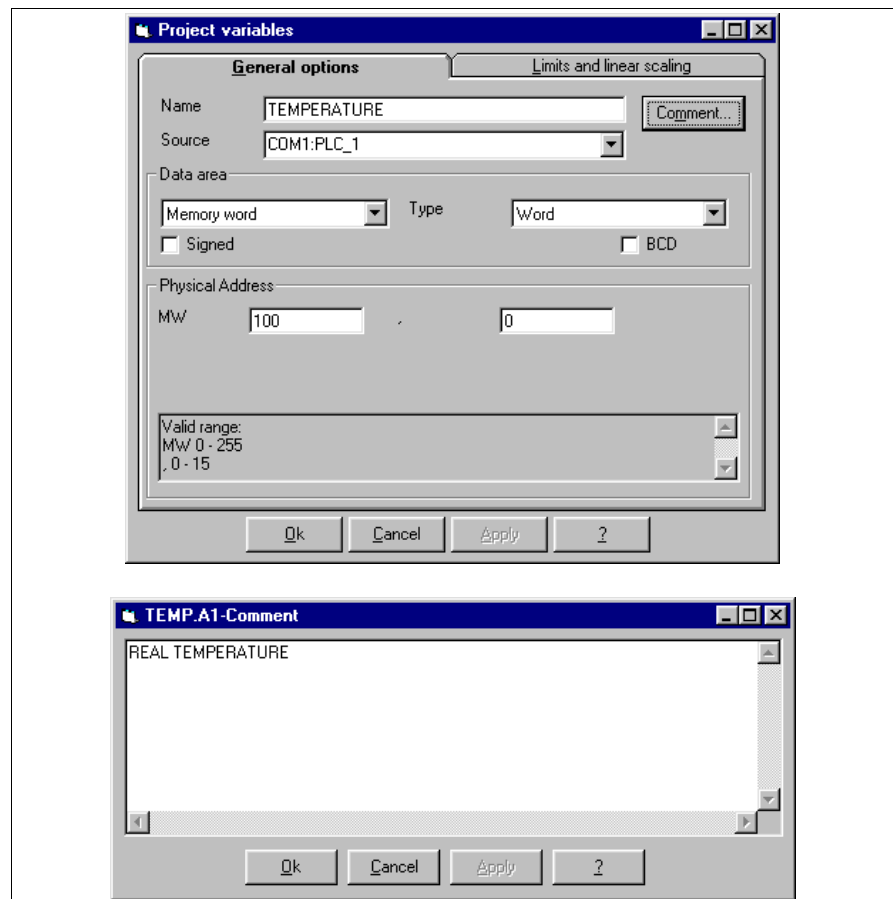
You can proceed in two ways, inserting all the elements like variables, direct commands etc. and then inserting them in the pages, or inserting the pages and step by step creating whatever is needed.

We will choose a mixed procedure (See “Chapter 4 -> Variables“).



Select **la** Variables.

Click on **Add**.



Assign a name to the variable so as to be able to recognize it easily in the list: **TEMP.A1** Add a comment to the variable by clicking on **Comment**.

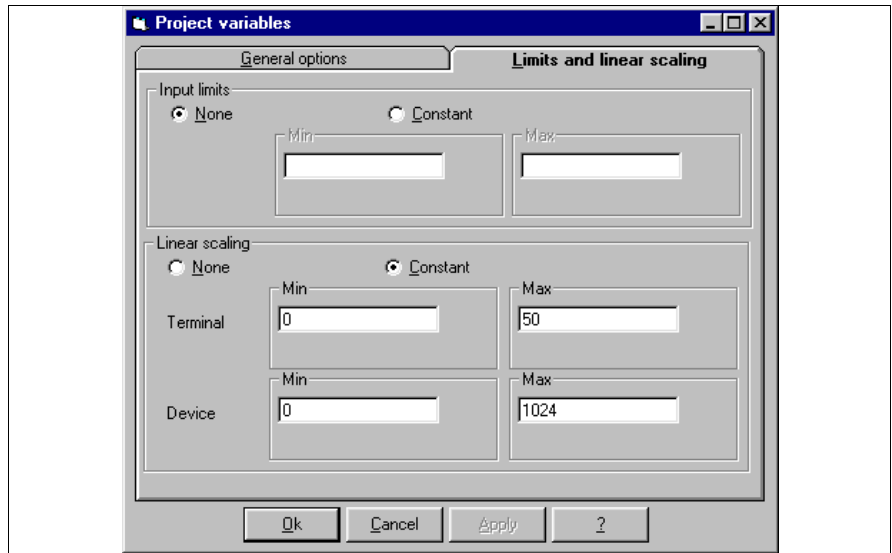
Click on **OK**, then parametrize as illustrated.

The comment should be as exhaustive as possible.

Click on **Limits and Linear correction**.

*Fix the input limits relating to the TC, set the linear correction so as to be able to display the temperature correctly, converting it automatically from the value actually read.*

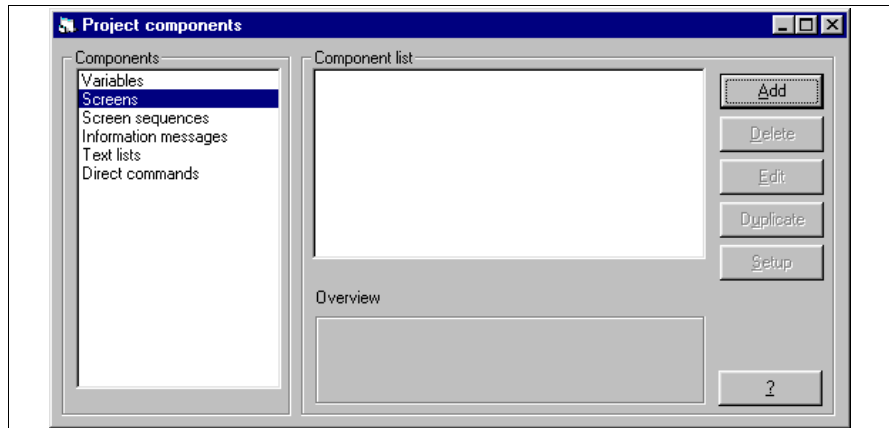
*Click on Ok.*




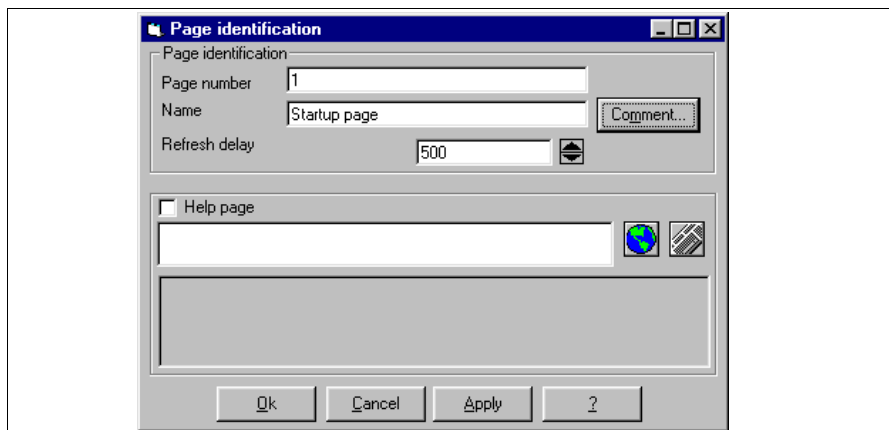
Repeat the operations described above to insert all the variables necessary (See "Table A.1.; Appendix A -> Variables").

## Inserting pages


Select the  Screens (See “Chapter 4 -> Page“).

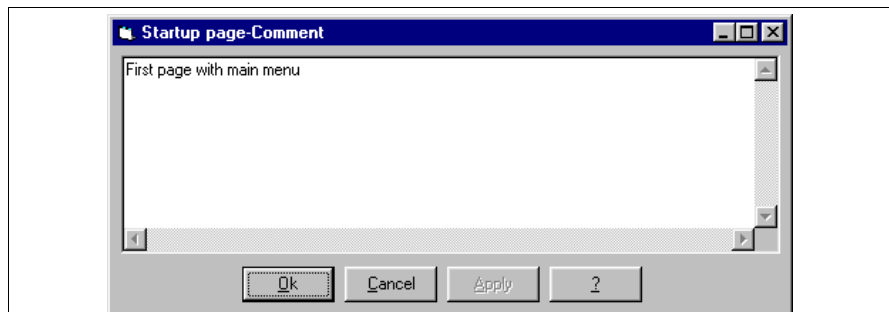


Click on  Add.



Assign the number and name to the page, and set the refresh delay.

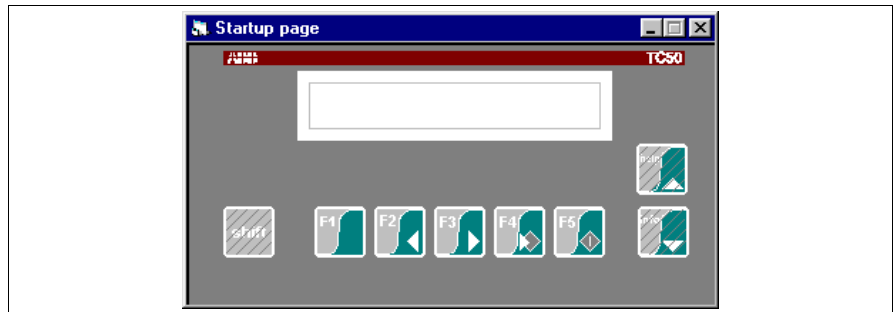
Add a comment to the page by clicking on  Comment.



Once the comment has been edited click on *Ok*. (A comment only in the mother tongue is envisaged).

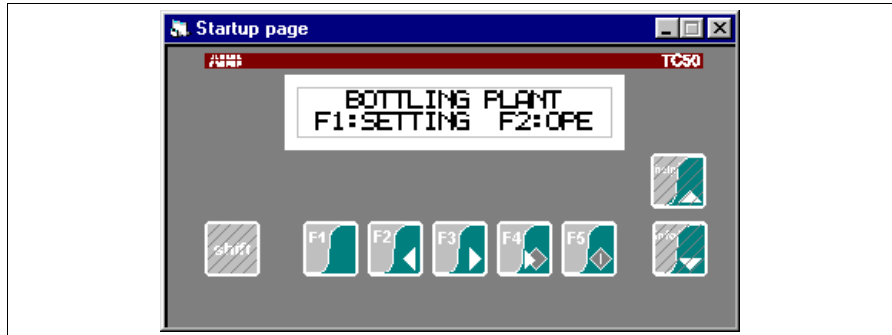
Given the simplicity of the function on this page no Help page exists, so all data is accepted by clicking on OK.

A blank page appears:




The various elements can be inserted at this point.

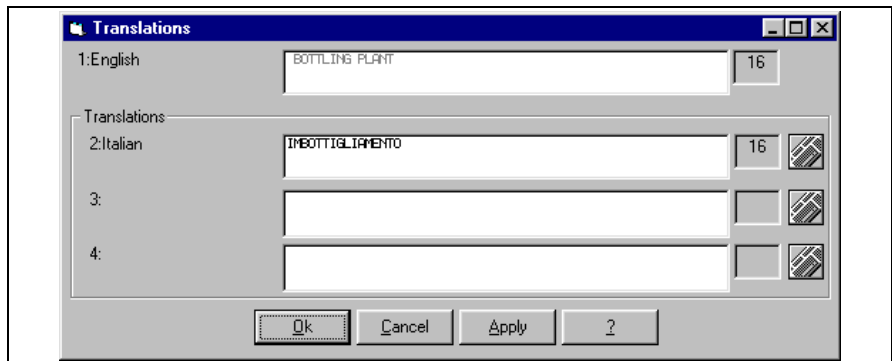
## PAGE 1 - Startup page



Multilanguage labels:  
 BOTTLING PLANT  
 F1:SETTING  
 F2:OPE

This page is seen when the TC is switched on and enables the user to call up the functions indicated by the buttons.

To begin with we insert the multilanguage label (See “Chapter 4 -> Multilanguage label“). Click on  and position the cursor on the page; edit the text and confirm,

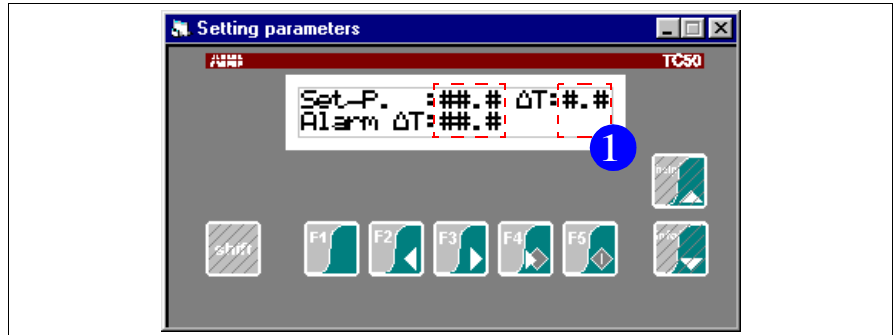


The text of the translations cannot be longer than the mother tongue text. If the translation should require more characters the mother tongue can be lengthened by adding spaces.

Inserting texts that need translation, this needs to be borne in mind.


Insert all text and click on Ok to confirm.

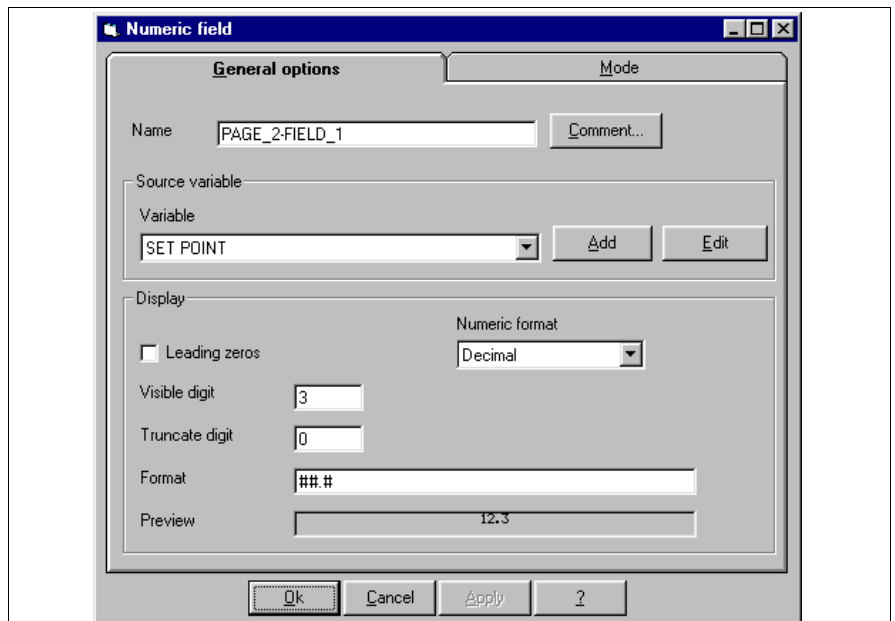
## PAGE 2 -> Setting parameter 1




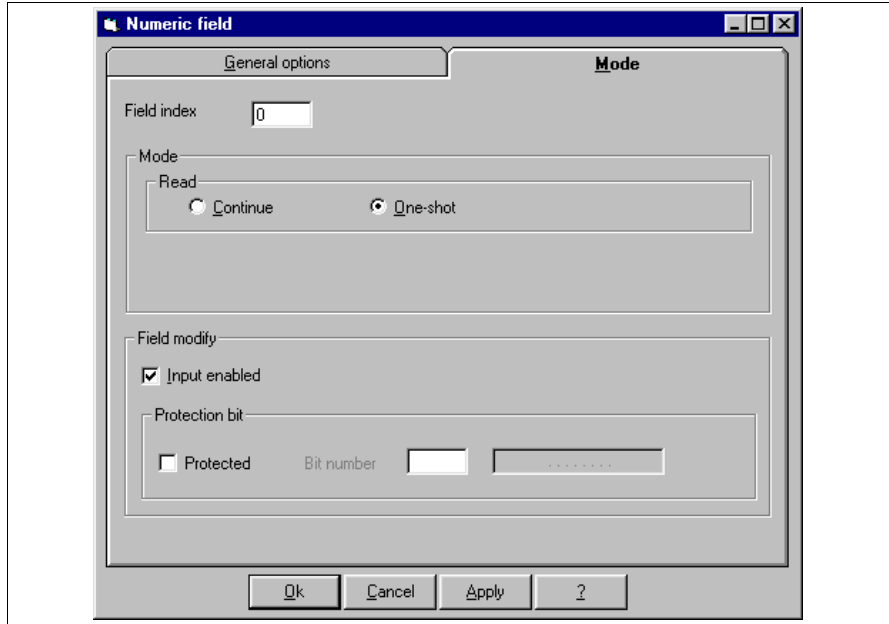
Numerical data field (1):

Variable:  
 TEMPERATURE  
 SET POINT  
 dT  
 ALARM TEMPERATURE

Insert the numerical data. Click on , move the pointer onto the area of the display where the data is to appear. Click.



*Compile as illustrated;  
 there is no comment;  
 then browse the  by  
 clicking on Mode..*



*Compile the fields as illustrated to obtain a continuous read data, which is settable using the TC.*

*Click on OK*

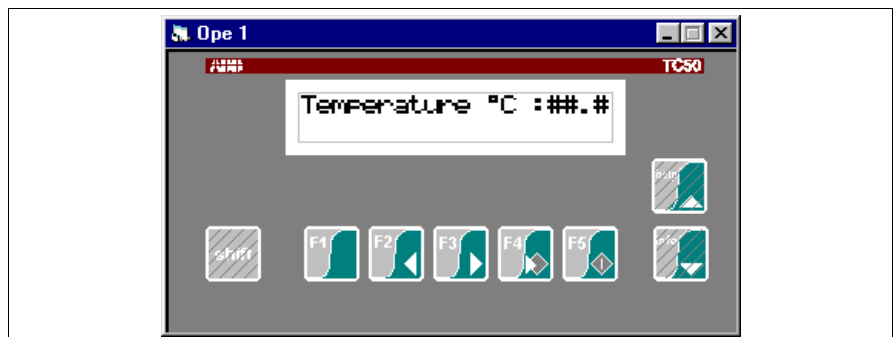
Assign a Multilanguage label comment for SET-P.; repeat this procedure for the other elements.

Insert the pages listed below.

## PAGE 3 -> Setting parameters 2

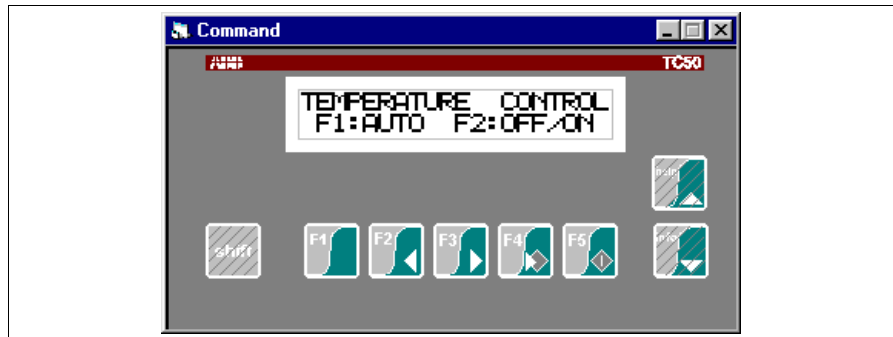


## PAGE 4 -> Ope 1




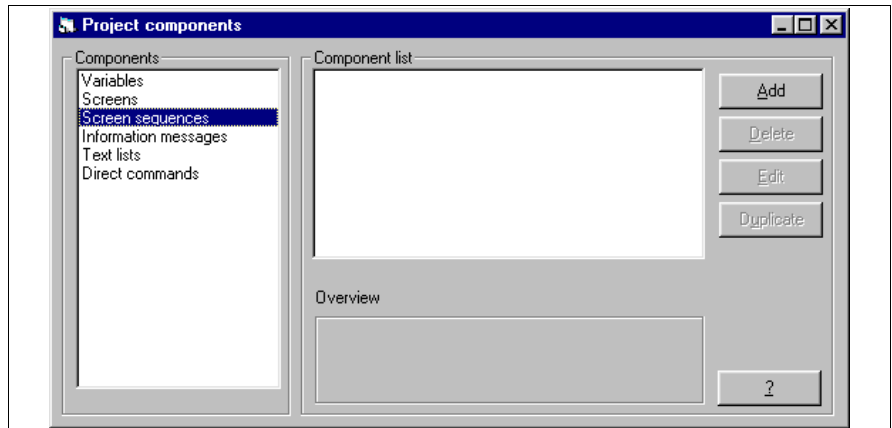
## PAGE 5 -> Ope 2



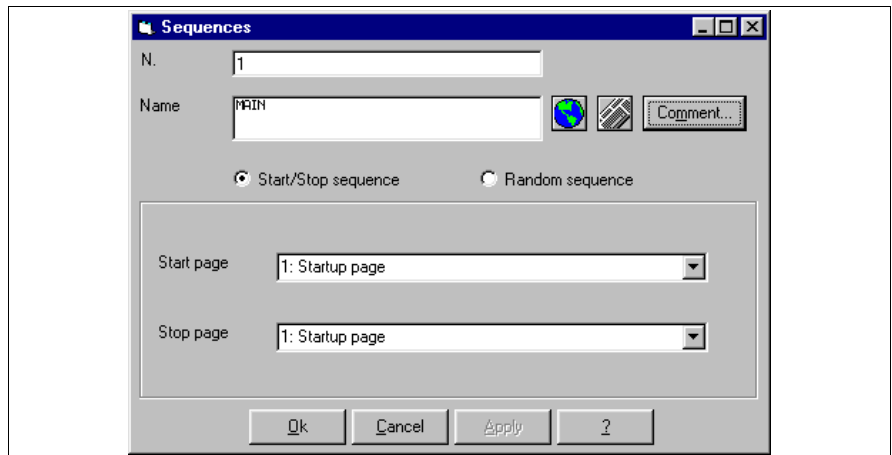
**PAGE 6 -> Command**

## Inserting sequences

Select the  Screen sequences (See “Chapter 4 -> Page sequences“).



Click on  Add.




Assign a name to sequence.

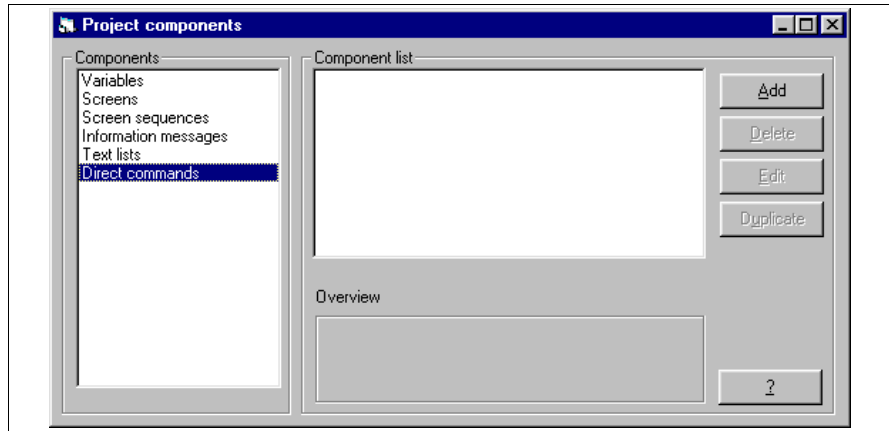
Compile as illustrated.


Click on OK.

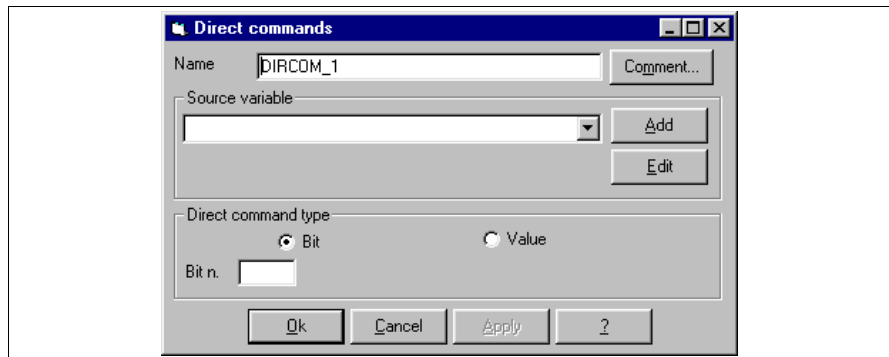
Inserting the rest of screens sequences (See “Table A.3.; Appendix A -> Screen sequence” ).

**Inserting  
direct  
commands**

Select the  Direct commands (See “Chapter 4 -> Direct Commands”).

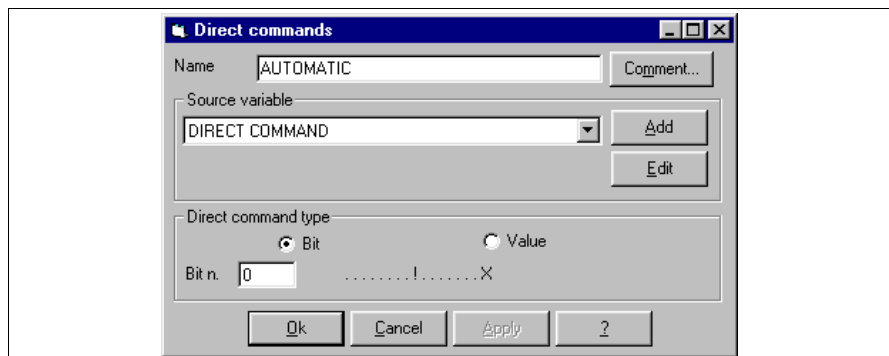


Click on  Add.



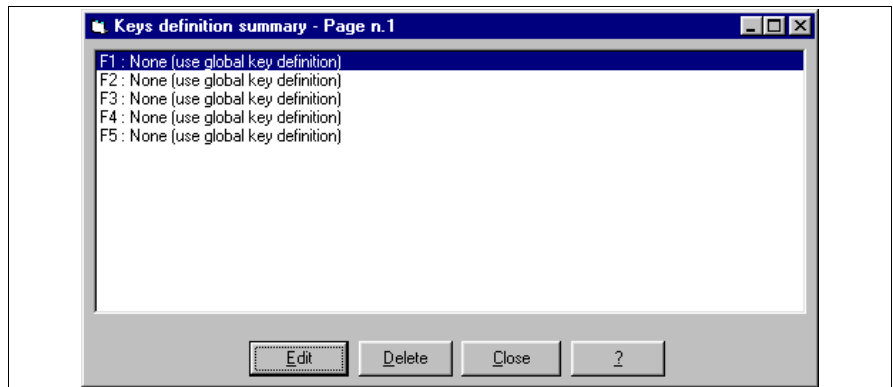
Assign the name and the variable as illustrated.


Click on OK.

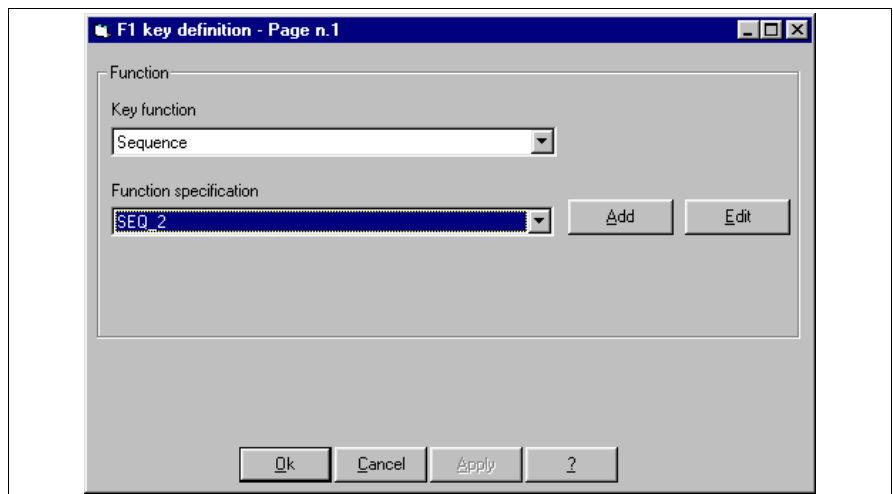


Insert all the Direct commands (See “Table A.5.; Appendix A -> Direct commands”).

Now, click on **Page > Key definition** (See “Chapter 5 -> Keys definition“).

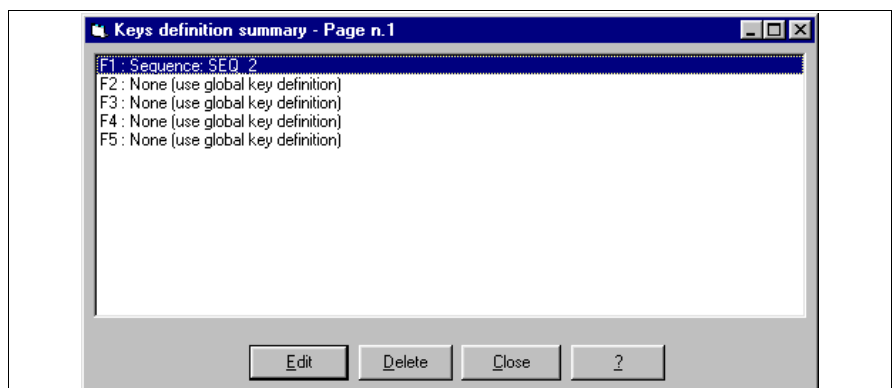



Select as illustrated and click on  Edit.



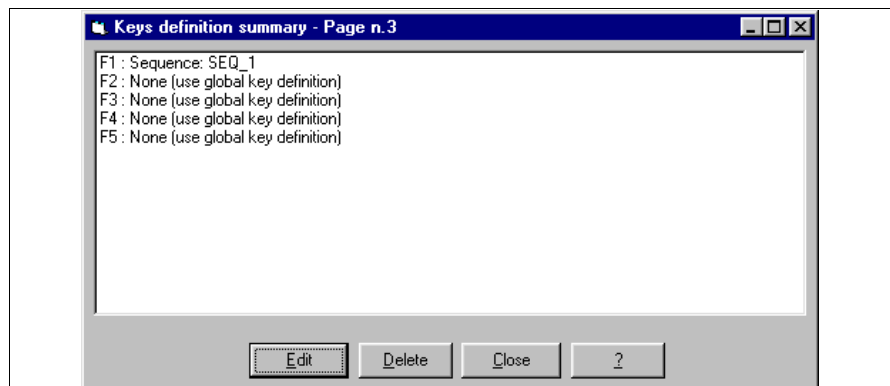
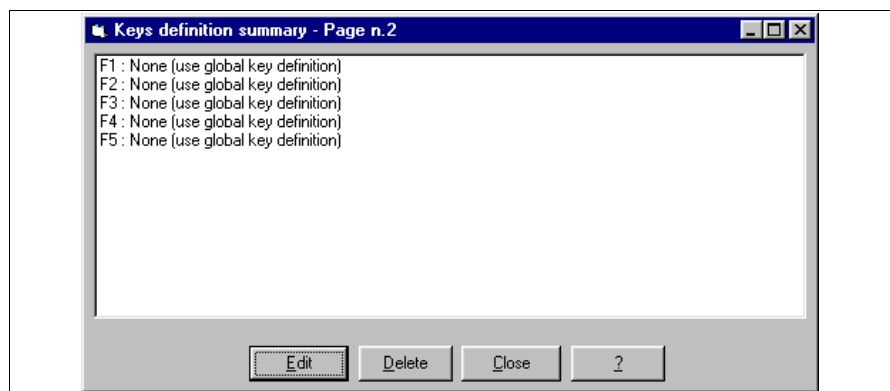
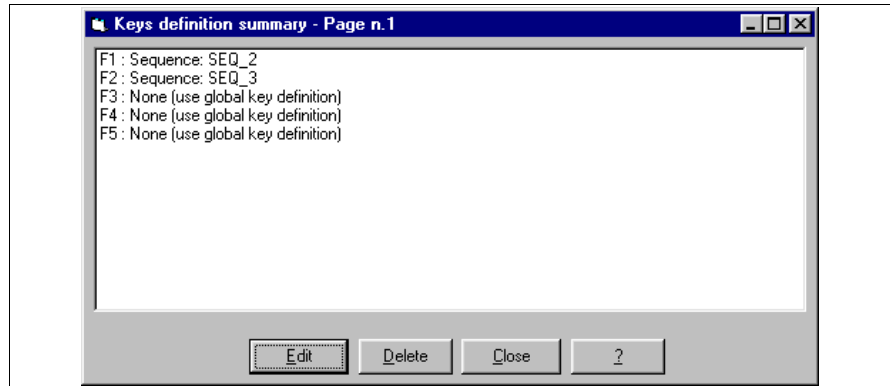
Insert the definition of the function.

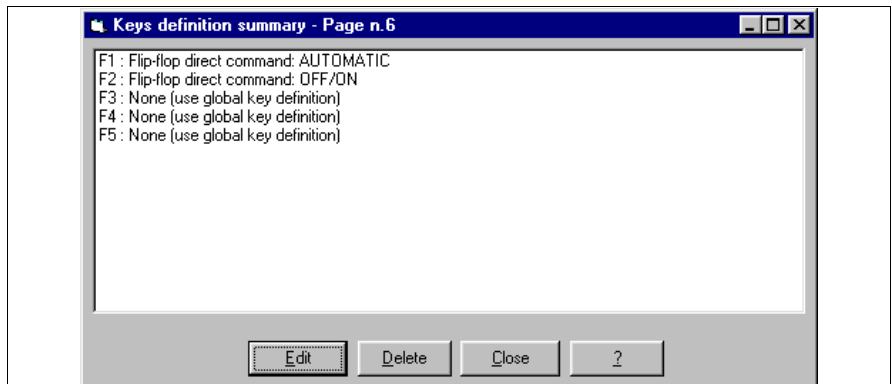
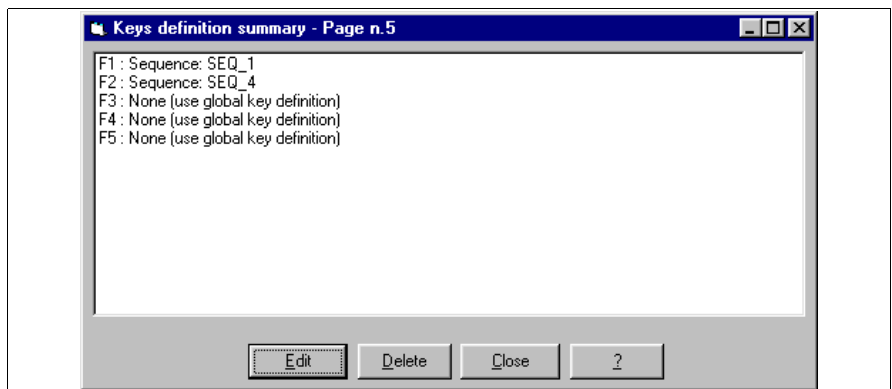
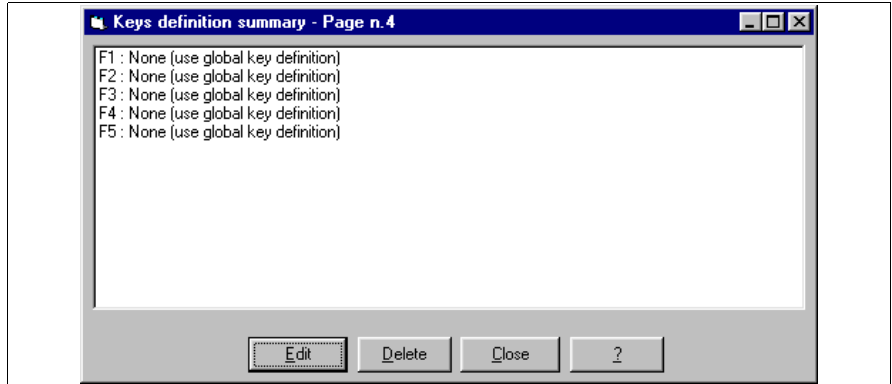
Click on Ok.



Select as illustrated and click on  Edit.

Using the same procedure edit all the Key fonction.

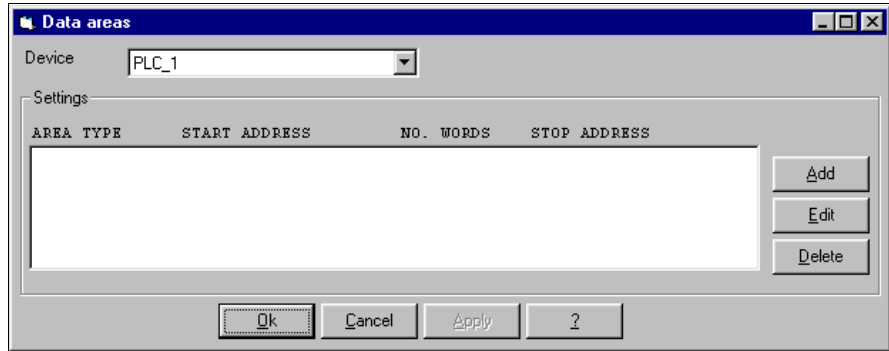




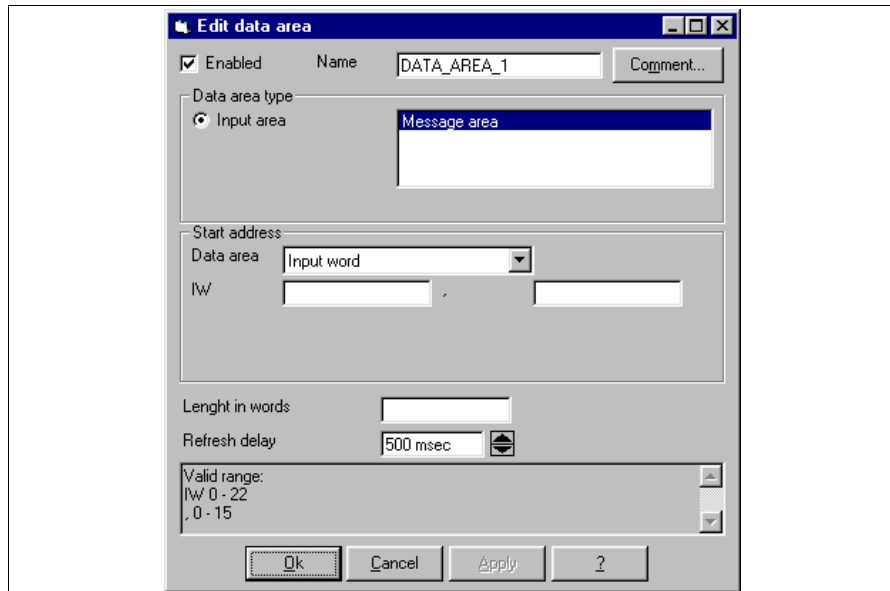
**Data exchange area**

Given the need to define the information messages and the commands to be exchanged with the connected device, it is essential that this area be defined. Click on

*Configure > Area for terminal <---> device exchange* (See “Chapter 5 -> Exchange area Terminal <-> Device“).



Click on Add.



**Edit data area**

Enabled    Name: DATA\_AREA\_1    Comment...

Data area type

Input area    Message area

Status area

Command area

Start address

Data area: Memory word

MW: 0    10

Length in words: 1

Refresh delay: 500 msec

Valid range:  
MW 0 - 255  
0 - 15

OK    Cancel    Apply    ?

Set the parameters as illustrated.

Click on OK.

**Data areas**

Device: PLC\_1

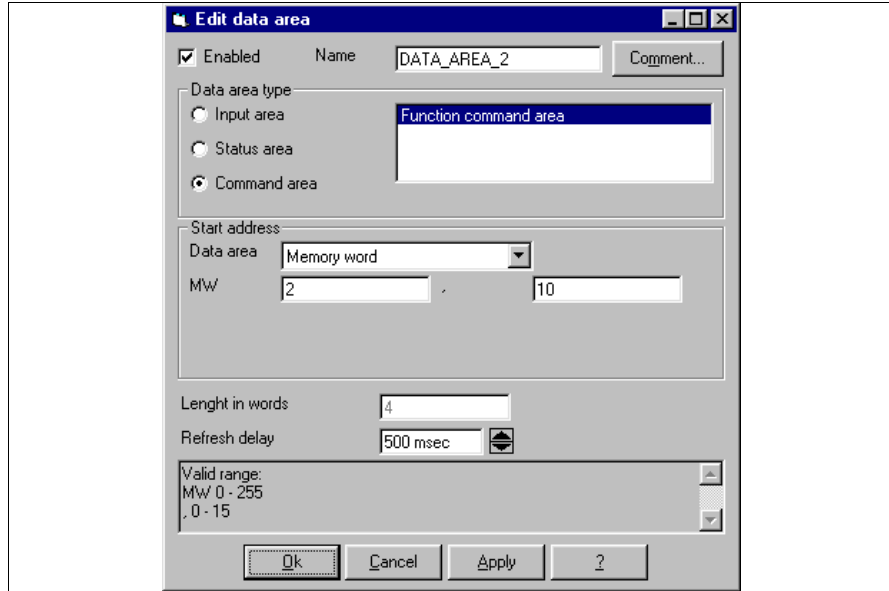
Settings

AREA TYPE	START ADDRESS	NO. WORDS	STOP ADDRESS
MESSAGE AREA	MW 0 , 10	1	MW 0 , 10

Add    Edit    Delete

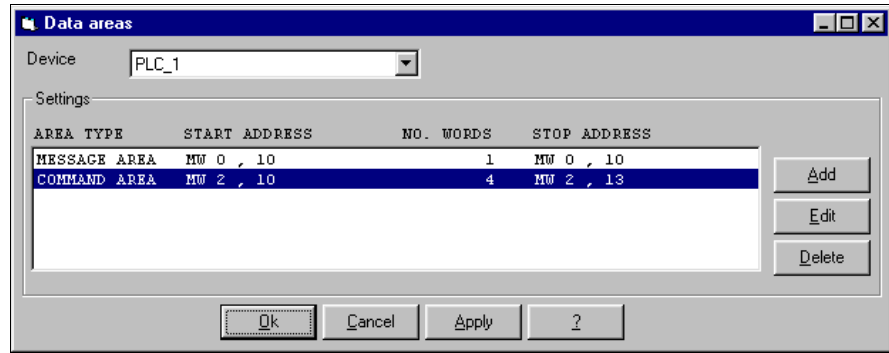
Ok    Cancel    Apply    ?

Click on Add.



Set the parameters as illustrated.


Click on OK.



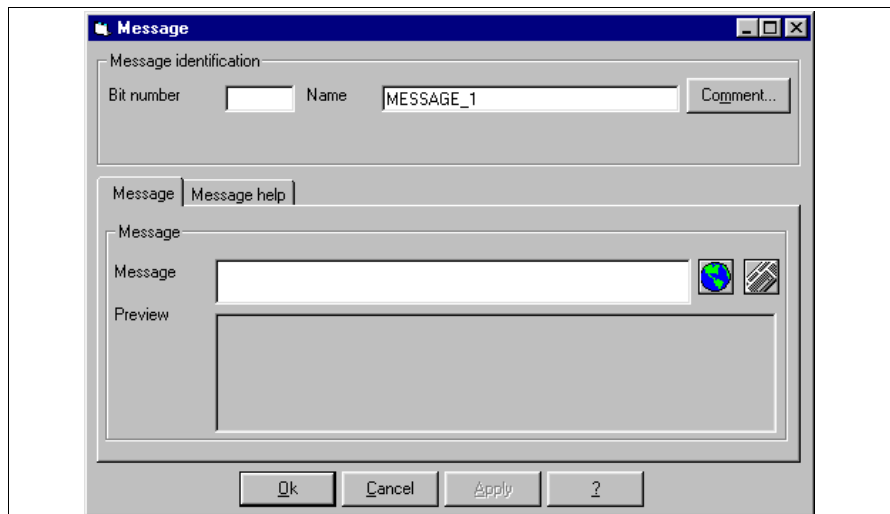
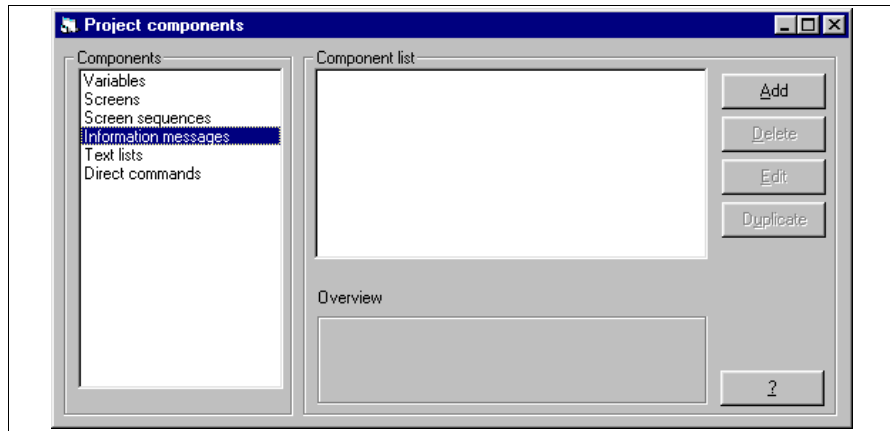
Click on OK to confirm.

**Information messages**

Proceed to insert the information messages. (See “Chapter 4 -> Information Messages“).


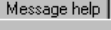
Select the  desired.

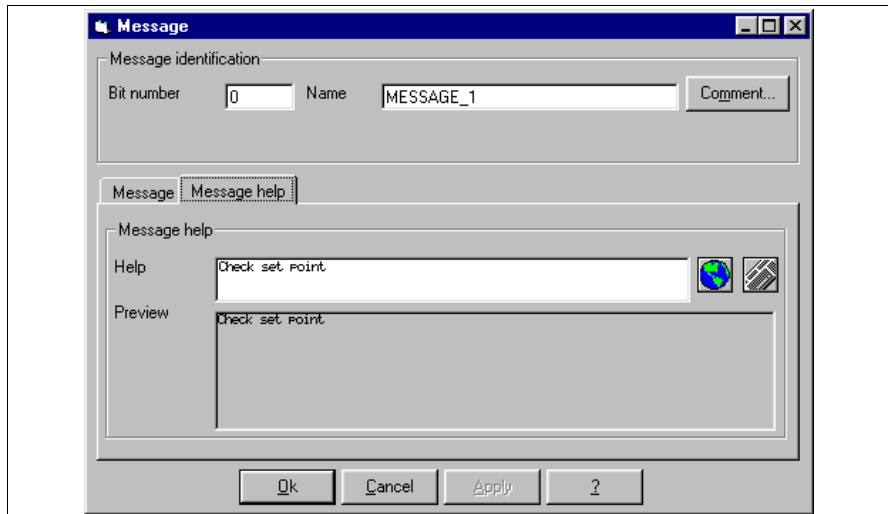
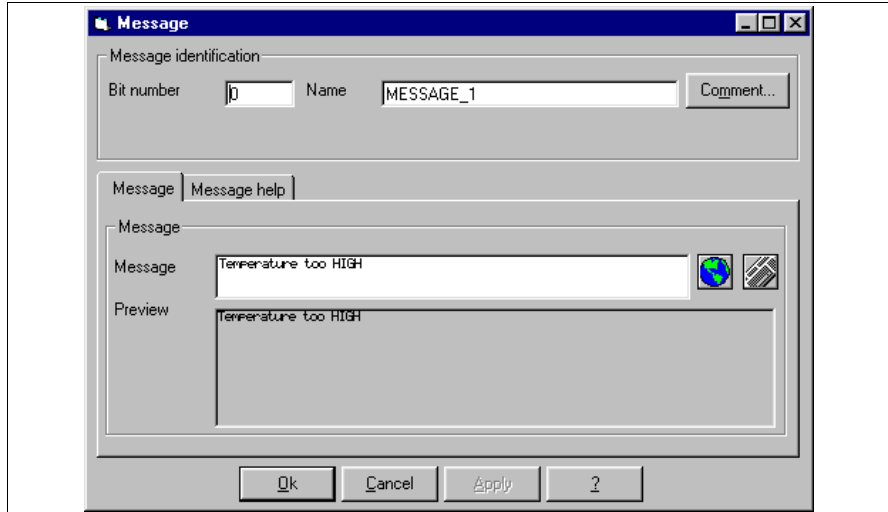
Click on  Add.




*Insert the bit number to which the message must be assigned; assign the name to the message and edit the text.*

*The comment is not assigned, because further information is not needed to explain the function of the message*

*Then browse the  by clicking on  Message help.*



*Insert the help message.*

Click on  to insert the translation. Insert the text and click on OK to accept; you will return to the previous mask, click again on OK.

Using the same procedure insert all the messages (See “Table A.4.; Appendix A -> Information messages”).

Now the project is complete and ready to be compiled and transferred. (See “Chapter 8 -> Compiling and transferring a project”).

## Compiling a project

Compilation is a method of creating a file automatically in a format the TC panel can recognize.

During the compilation there is a control phase that makes it possible to detect any errors introduced while the project was being built. If an error is detected during compilation, it is highlighted by the text in the compilation window being colored red and at the same time the errors being displayed.

To start compiling click on **Tools > Project compilation**.

Compilation can be configured as follows.

Stop at first error:

The compilation can be stopped at the first error encountered in the project.

No stop:

Even if an error is encountered, compilation will not stop but proceed possibly finding other errors.

Stop after N. errors:

The user can decide the number of errors to detect before stopping the compilation.

Display warnings:

The user can decide whether to display warnings too during compilation. The warnings are not considered errors, so compilation proceeds, but they advise the operator that a part of the project has omitted and/or not completely compiled.

Output:

Shows how the compilation is proceeding. The information displayed can be saved in a file by pressing the  Save output.

**Transferring the project**

Once the project has been correctly compiled it must be transferred to the operator terminal. To do this click on **Tools > Project transfer**. If the project has not been compiled yet, the system automatically proposes the compilation window with the above listed parameters; otherwise the transfer window used for selecting the parameters for communication between the PC and the TC comes up.

To prepare the TC for transferring the project see the relevant Hardware Manual.

The parameters for transferring the project must be compiled; these are listed below.

Communication port:


With this the communication port used by the PC can be chosen.

Baud rate:

With this the speed of data transfer can be selected.

Fw update:

With this you specify whether during the transfer the TC Firmware is also to be transferred.

It is used to force the loading of the firmware. If TCWIN recognizes that the firmware contained in the panel is an old version, that firmware is automatically reloaded. Normally this  is not activated because, for one thing, the transfer times become much longer. It can be activated in the event that there be doubts as to whether the TC is functioning correctly.

---

---

## Chapter 9      Creating and printing documentation

### **Importance of documentation**

The creation of documentation is an important phase in the development of a project.

It is possible at any time to consult, re-elaborate or simply re-check what has been created. At the termination of a project this assumes an even greater importance where problems are detected after a period of time; not least, in the event of data being lost, it is possible to get back to what was originally elaborated.

The type of documentation created is settable by the user and makes it possible to prepare print patterns that can be used as the needs of the moment dictate.

### **Print the project**

To print documentation the following steps must be followed. Click on *Tools > Print...*

The print program is activated with the following parameters to compile and/or select.

Setting:

Name of printer:

The name of the printer used is shown.

Printer port:

Shows the port to which the printer is connected.

Pattern names:

Allows you to select which print pattern to use.

Pattern descriptions:

Shows what the pattern displayed and/or chosen permits you to print.

Preview:

Shows what the documentation will look like when printed.

---

**Check:**

This enables you to browse the print preview.

**Overview:**

This enables you to select the Layout (1 or 2 pages) for the preview display.

**Edit:**

Activates the editing of a print pattern.

**Name:**

Name of the pattern being edited; the name too can be edited.

**Optional sections:**

This allows you to edit the type of information that you wish to print. You can also edit the order in which this information is to be printed. (The print order is the order in which the information is inserted in the list).

**Available sections:**

This is a list of the information that can be printed.

**Sections selected:**

This is the list of the information chosen to be printed.

**Global settings:**

These permit the selection of print settings.

**Include cover:**

With this you can determine whether or not to have a cover. If you choose to do so, you can choose from the list (assuming at least one cover has been created).

**Index:**

Determines whether or not to have an index.

---

---

Project information:

Determines whether or not to have the project information.

Comments in all sections:

Determines if there should be comments in all the sections.

Page settings:

Allows the user to determine the Layout of the page. (To be valid for all the pages).

Margins:

Allows the user to define the margins of the page.

Header:

Makes it possible to write a header line and choose whether to print it or not.

Footer:

Makes it possible to write a footer line and choose whether to print it or not.

Page numbers:

Determines where to put the page number.

---



---


## Chapter 10      Creating a back-up of the project

### Importance of a Back-up

This operation that only takes a few seconds protects the user from any accidental losses of data.

It is a good habit to create every so often for reasons of safety a back-up copy of what you create and/or edit.

It is important to save the .MDB file; all the files necessary for the project itself can be obtained from this file.

 **If you lose the source project (.MDB), the information contained in it will be lost definitively; a possible recover from the TC panel or the possession of the compiled files alone permit the project to be transferred to another terminal comparable to the one for which the project had been created, but it will not be possible to edit the project in any way.**

### How to create a Back-up

To create a copy of the project, click on *File > Save as...* (see “Chapter 5 -> Save as...”). Use a support medium other than the hard disk and if possible put it in a safe place.



## Chapter 11 Defining the fonts

TCWIN contains a program allowing the character fonts to be edited and/or created.

In the case of text panels 7 characters (from 1 to 7) can be redefined. This is because the display used contains a predefined non-modifiable set of characters. Different fonts can be created, but for every font created there are always 7 characters (from 1 to 7) that can be modified.

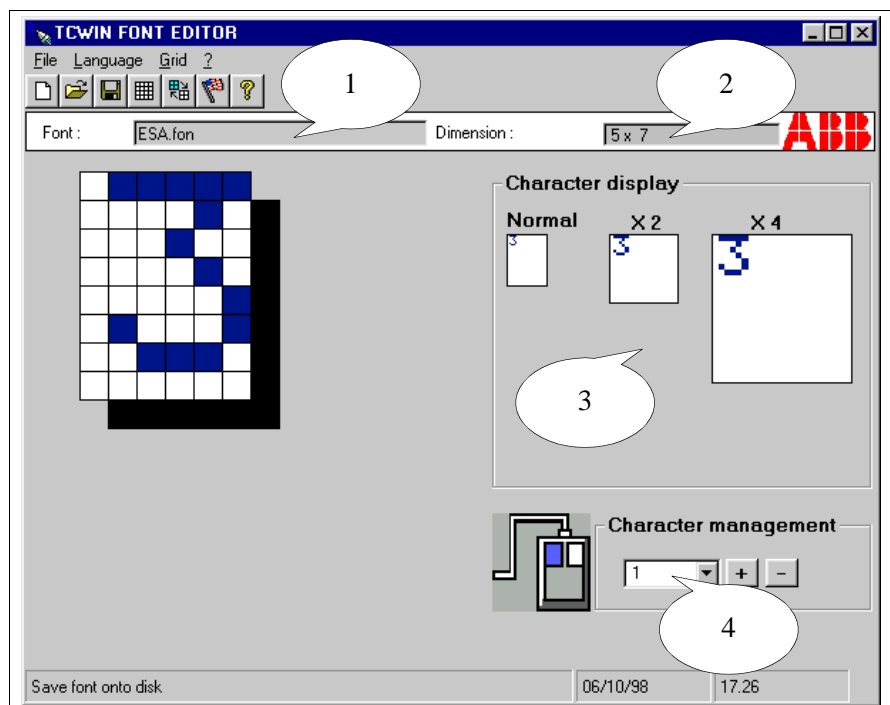
To call up the program click on **Tools > Font definition**. The main window becomes activated.

1) Displays the fonts currently operational.

2) Displays the dimensions of the font in operation.








3) Shows how the character is displayed in various enlargements.

4) Shows the currently active number of the character in the table. The image of the mouse with the left button is blue indicates that the character is editable.



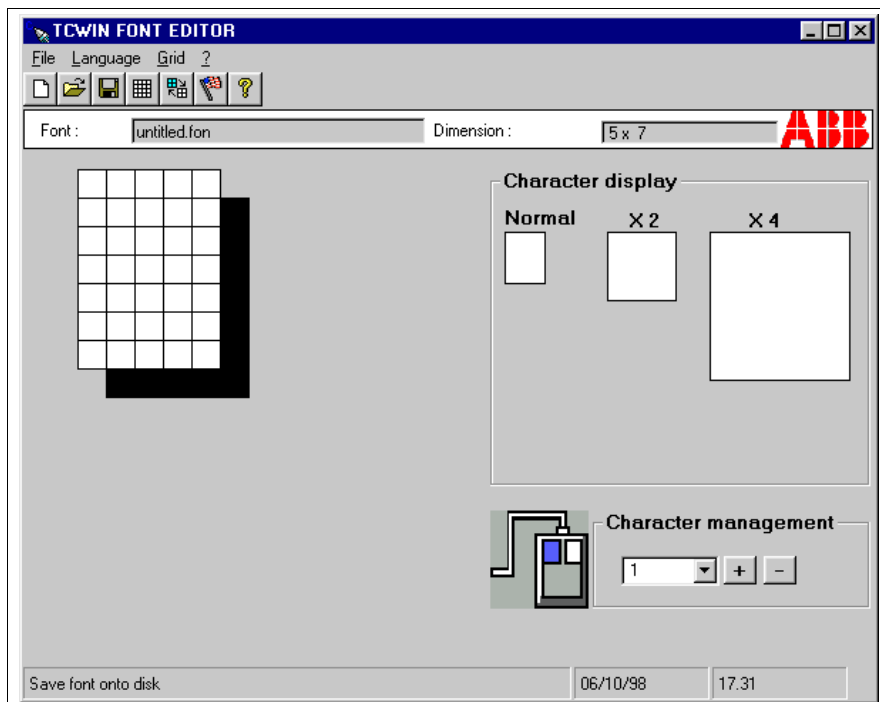
**Meaning of the icons used in the menus** The table shows all the icons of the menus together with their meanings.

Table 11.1: List of icons used in TCWIN Font Editor, menu attribution and meaning.

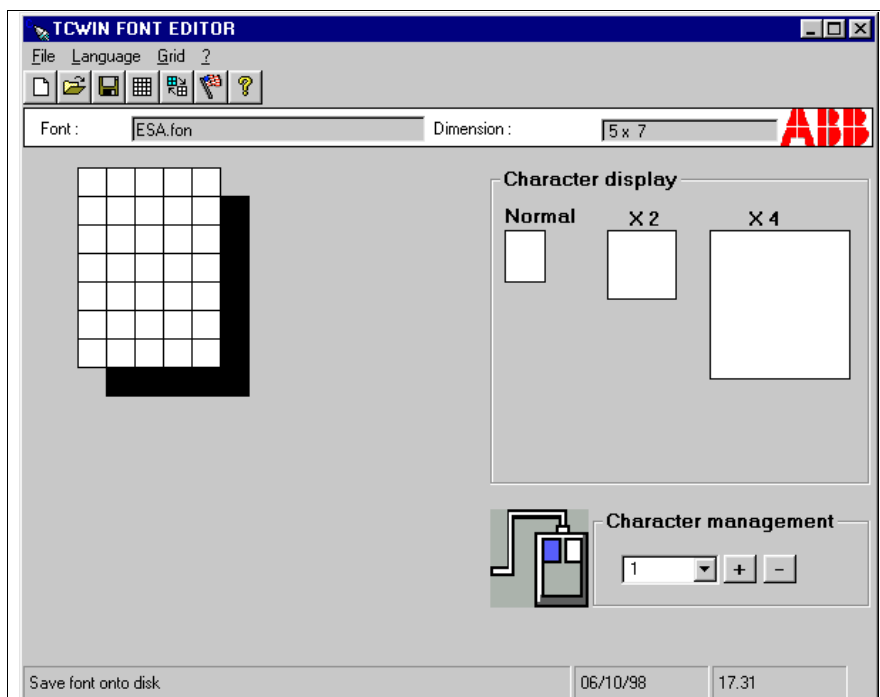
Tool Bar	Pulldown menu	Action
	<i>File &gt; New</i>	Creates a new font.
	<i>File &gt; Open font</i>	Opens an already existing font.
	<i>File &gt; Save font</i>	Save a font on disk.
	<i>Grid &gt; Clear</i>	Deletes the content of the grid.
	<i>Grid &gt; Invert</i>	Inverts the content of the grid. (White becomes black and vice versa).
	<i>Language</i>	Allows the language of the program to be selected.
	?	Calls up the Font Help.

## Personalizing a Font

Below is an example of personalizing a project font by generating a new one, the font CUST5X7.fon.



Click on *File > Open font*



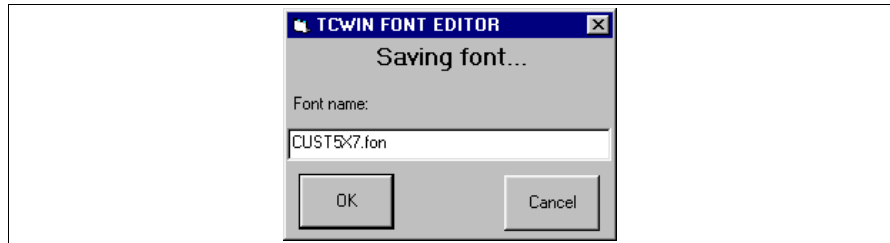
Select the font as illustrated.

Click on OK.

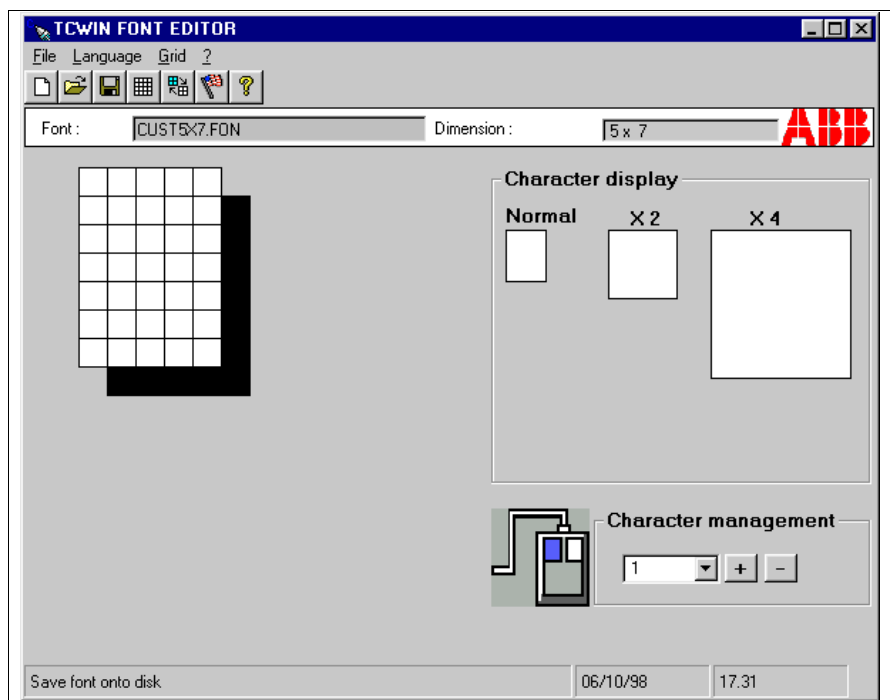
Click on *File > Save font*

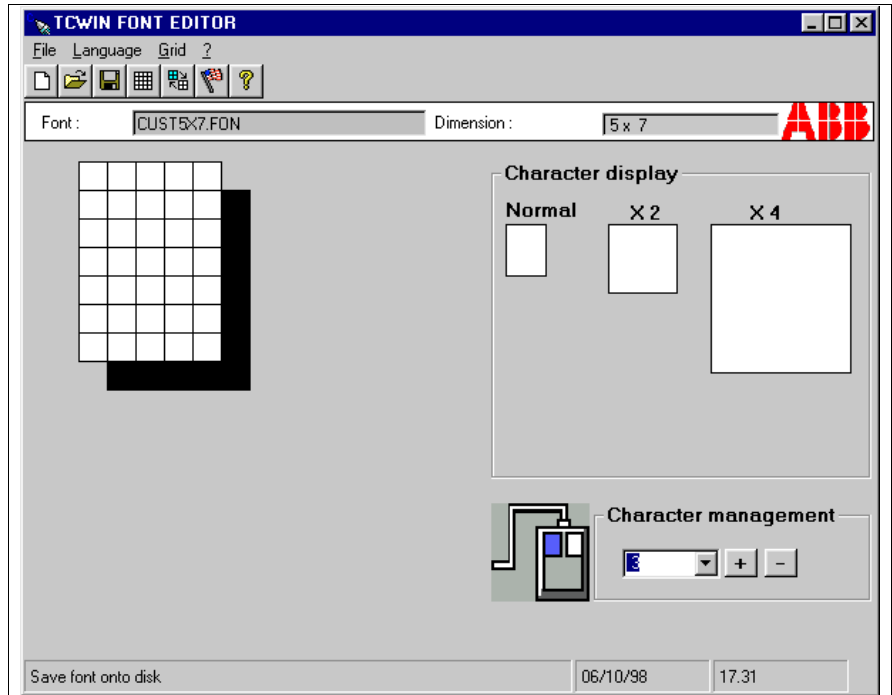
Assign a name as  
illustrated.

Click on OK.



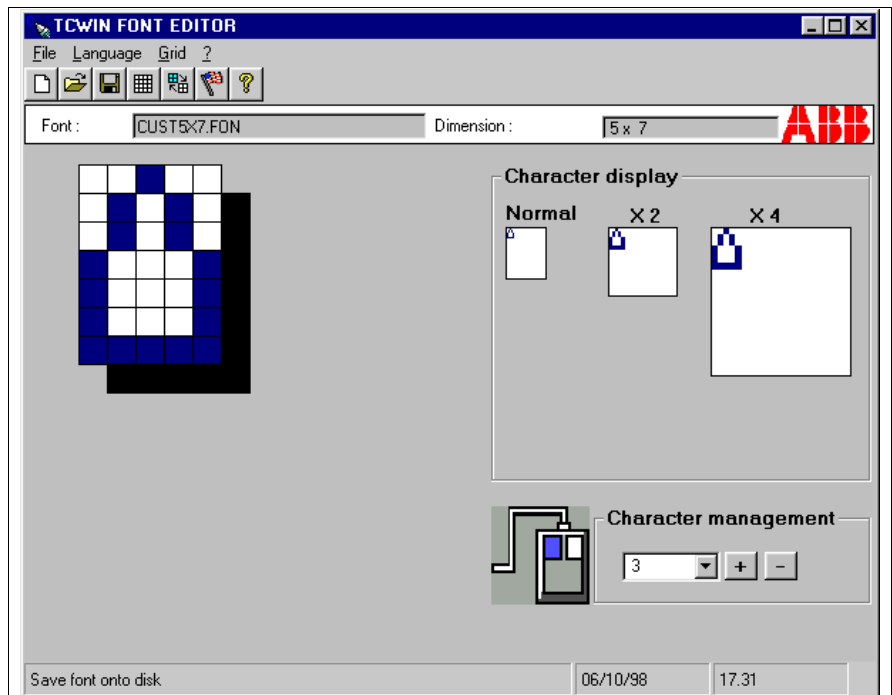
This operation saves the initial font which, in the event of some error, is not ruined. After this operation the new font is automatically loaded.





Set "Character management" arbitrarily on 3.

Select a pixel at a time and click till the character desired is obtained.



Once the character has been completed, click on *File > Save font*.



# Appendix A

The tables below show the contents of the DEMO project.

Table A.1: Variables

Name	Comment	Memory MW	Flag	Sign	BCD	Setting		TC scale		Device scale	
						Min.	Max.	Min.	Max.	Min.	Max.
ALARM TEMPERATURE	--	106,0	--	no	no	0	100	--	--	--	--
DIRECT COMMAND	--	110,0	--	no	no	--	--	--	--	--	--
dT	--	104,0	--	no	no	0	10	--	--	--	--
SET POINT	--	102,0	--	no	no	0	400	--	--	--	--
TEMPERATURE	--	100,0	--	no	no	--	--	0	50	0	1024

Table A.2: List of pages

Page	Name	Coment	Refresh	Help	Help Text
1	Startup page	First page with main menu	500mS	no	--
2	Setting parameters_1	--	500mS	no	--
3	Setting parameters_2	--	500mS	no	--
4	Ope_1	--	500mS	no	--
5	Ope_2	--	500mS	no	--
6	Command	--	500mS	no	--

Table A.3: Screen sequence

Sequences	Name	Comment	Start/Stop	Random	Start Page	Stop Page
1	MAIN	--	yes	no	1	1
2	SETTING	--	yes	no	2	3
3	OPE	--	yes	no	4	5
4	COMMAND	--	yes	no	6	6

Table A.4: Information messages

Name	Message	Bit N.	Help	Help Text
MESSAGE_1	Temperature too HIGH	0	yes	CHECK SET POINT
MESSAGE_2	Motor failure Cooling	1	no	--
MEGGASE_3	Motor failure Cooling	2	no	--

Table A.5: Direct commands

Name	Variable	Type
AUTOMATC	DIRECT COMMAND	Bit
OFF/ON	DIRECT COMMAND	Bit

Table A.6: Translations (Part 1 of 2)

English	Italian
Alarm T:	All. dT :
BOTTLING PLANT	IMBOTTIGLIAMENTO
Check set point	Controlla Set-P
Check set point	Controllare set point
COMMAND	COMANDI
F1:AUTO	F1:AUTO
F1:MAIN MENU	F1:MENU PRINCIPALE
F1:MAIN MENU	F1:MENU PRINCIPALE
F1:MAIN MENU	F1:PRINCIPALE
F1:SETTING	F1:SETTING
F2:COMMAND	F2:COMANDI
F2:OFF/ON	F2:OFF/ON
F2:OPE	F2:OPE
MAIN	PRINCIPALE
Motor failure Cooling	Termica Raffreddamento
Motor failure Heating	Termica Riscaldamento
OPE	OPE
Set-P. :	Set-P. :

Table A.6: Translations (Part 2 of 2)

English	Italian
SETTING	SETTAGGI
T:	dT:
TEMPERATURE CONTROL	CONDIZIONAMENTO
Temperature too HIGH	Temperatura elevata
Temperature °C :	Temperatura °C :

