VTWIN

Software Manual

405.1200.038.2 - Rel.: 2.20 of 26/03/2007 - Only for VTWIN Ver. 3.XX or above

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 VT 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 -> VTWIN menus" and then jump directly to "Chapter 7 -> Using VTWIN".

Contents

Foreword

The programming manual is the same for all models of Video Terminal and for the different types of accessories, since there are no differences in the method of programming.

Video Terminals can be divided into two groups: those with a keyboard and those without, but equipped with a touch screen.

Most of the examples used in this manual refer to a Touch Screen product. With this type of Video Terminal there is no such thing as a page sequence (you will find an explanation of what these are further on); on e other hand, if you use a Video Terminal with a keyboard there will be page sequences.

The options given in the masks differ according to the type of Video Terminal, thus the same mask in one case contains the option Start up page, while in the other case the option is called Start Sequence.

The information contained in this manual refers to top of the range terminals. To check which functions a given VT has, consult the relevant Hardware Manual.

The manual

The programming manual is the tool that allows the user to create his or her own application packages for the VIDEO TERMINALS (VT) and for the accessories that can be connected, ADAPTERS (ADT), by using the VTWIN 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 or touch button.
🕮	Identifies an option within a window.

🗁	Identifies a folder.
⚠	Draws attention to essential points**.**

Introduction

What is VTWIN?

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

VTWIN will only work in a **Windows 95/98/Me/2000/XP** or **Windows NT** environment.

Requirements for displaying the Help on Line

VTWIN contains an exhaustive Help on Line (HoL)**, available in the future**, 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.)

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

What is a project?

The definition of the term project changes according to whether the product being used is a VT or ADT directly connected to a personal computer.

In the context of a VT, a PROJECT may be defined as a set of screens (defined later as PAGES) having the same dimensions as the display of the VT 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.

In the context of an ADT, a PROJECT may be defined as a set of variables arranged in groups (referred to later as *Groups of Variables*), which must be used by the applications created by the user to handle the variables (read/write) using a Personal Computer (see “Chapter 18 -> ActiveX“).

Type of project

VTWIN allows the user to generate two types of project: one called “Single VT” and one called “ESA-NET network”. The difference between the two is simply that the former allows you to create projects for VTs and ADTs that are not connected as an ESA-NET network, while the latter allows you to create projects to be used by VTs and ADTs connected in a network. By connecting a number of VTs and ADTs as an ESA-NET network the information held in one of the devices connected directly to one of the participants can be shared by the remaining VTs and ADTs in the network (see Hardware Manual). One network project can be the source of one or more single projects (the maximum number being the number of terminals making up the network project) and, vice versa, several single projects can make up a single network project.

The projects generated with the 2.xx versions of VTWIN and opened using version 3.xx are automatically duplicated in an equivalent format (.VTS).

Files generated by a VT project

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

Table 0.1: Significance of the files (Part 1 of 2).

Extension	Location	Significance
.VTS	Project directory	Single project file -- all the files needed for the project are obtained from this file. **Loss of this file will cause the project to be lost.**
.VTN	Project directory	Network project file -- all the files needed for the project are obtained from this file. **Loss of this file will cause the project to be lost.**
.OBJ	Project PrjEditor	Temporary file obtained by compiling the .vts or .vtn file containing the text part of the project. This file is generated each time the project is compiled; it is present only for the time needed to transfer the .bin file and is then incorporated in the project file.
.OBG	Project PrjEditor	Temporary file obtained by compiling the .vts or .vtn file containing the graphic part of the project. This file is generated each time the project is compiled; it is present only for the time needed to transfer the .bin file and is then incorporated in the project file.
.BIN	Main directory of VTWIN	File transferred to the VT. It is obtained following a command to transfer the project from the PC to the VT. It groups together the information contained in the .obj and .obg files. The file remains in existence only for the time needed for the transfer and is then automatically deleted.
.PRJ	Main directory of VTWIN	Temporary project file. A numerical file may be found with this extension if VTWIN has been closed incorrectly. **The file can be removed once VTWIN has been closed.**
.BMP	Main directory of VTWIN	Temporary project file. A numerical file may be found with this extension if VTWIN has been closed incorrectly. T**he file can be removed once VTWIN has been closed. (Before removing check that the file has not been created on purpose by the user.)**

Table 0.1: Significance of the files (Part 2 of 2).

Extension	Location	Significance
.LDB	Project PrjEditor	Temporary file containing information on managing the database. A numerical file may be found with this extension if VTWIN has been closed incorrectly. **The file can be removed once VTWIN has been closed.**
.RCP	Dnloader directory	Recipe file obtained by using the VT Backup function. This file can be exclusively used to transfer it into another VT by using the Restore function.

The removal of the .VTS or .VTN file will cause an irrevocable loss of the project.

Files generated by an ADT project

Table 0.2 lists the extensions of the files generated by the ADT project.

Table 0.2: Significance of the files.

Extension	Location	Significance
.VTS	Project directory	Single project file -- all the files needed for the project are obtained from this file. **Loss of this file will cause the project to be lost.**
.VTN	Project directory	Network project file -- all the files needed for the project are obtained from this file. **Loss of this file will cause the project to be lost.**
.OBJ	PC_cmp directory	File to be transferred to the ADT. File obtained by compiling the .vts file or the .vtn file containing the text section of the project. This file is generated each time the project is compiled.
.FW	PC_cmp directory	File to be transferred to the ADT. File obtained by compiling the .vts file or the .vtn file containing the ADT operrative system, and the the drive associating ADT and device.
.CFG	PC_cmp directory	File obtained by compiling the .vts or .vtn file containing the information need for ADT to communicate with the PC. This file is generated each time the project is compiled.

The removal of the .VTS or .VTN file will cause an irrevocable loss of the project.

Chapter 1 Installing and updating VTWIN

This chapter is made up of a total of 2 pages.

HW requirements: minimum specifications

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

• Processor:	PENTIUM 166Mhz
• Operative system:	Windows 95/98/Me/2000/XP/NT 4.00 SP3 or later
• RAM memory:	32 Mbytes

HW requirements: ideal specifications

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

• Processor:	PENTIUM 200Mhz or later
• Operative system:	Windows 95/98/Me/2000/XP/NT 4.00 SP3 or later
• RAM memory:	16 Mbytes

Installation procedure

Insert the Cd-Rom in its drive; if on the PC you activate the function “Auto insert notification”; the presentation of VTWIN is displayed automatically; alternatively click on ***Start > Run…***

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

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

Follow the instructions on screen.

Updating procedure

Once VTWIN has been successfully installed, the automatic update procedure can start. Generally this procedure is applied in order to resolve any problems of functioning and/or where new devices are being implemented. The procedure can be followed immediately after installation, if the CD-Rom supplied already contains the update files (Service Pack), otherwise the updating can be done later.

To carry out the update all the applications must be closed.

To update, simply put the CD-Rom containing the Service Pack into the appropriate drive and:

- identify the folder containing the up-date and double-click on the file Setup.exe; the correct path on Cd-Rom is:

  \Service Pack\Vtwin 4.xx\Spxxx-xx\Disk1\
  (where x stands for the version of VTWIN and the version of the Service Pack)

  Follow the instructions on screen.

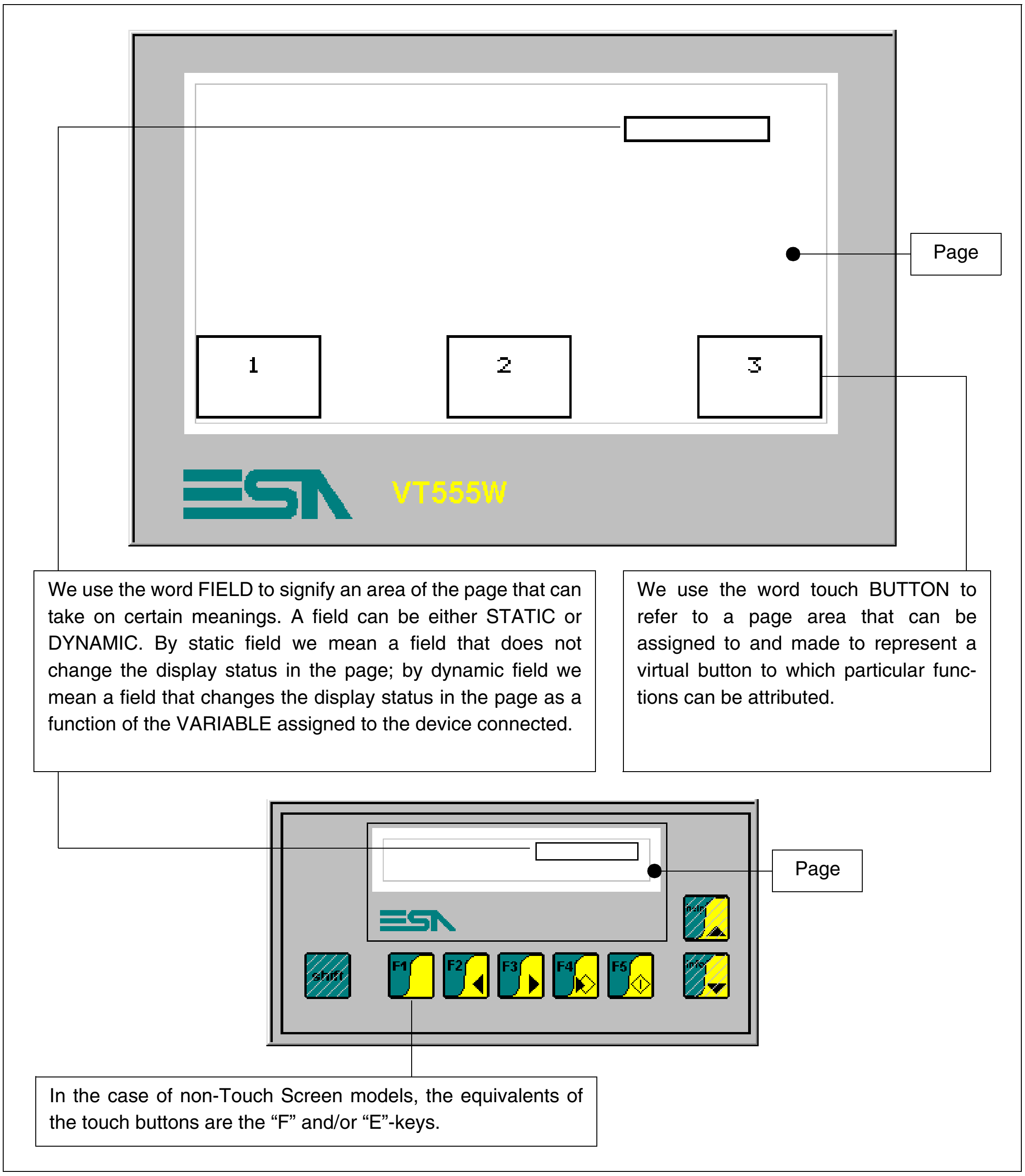
  The service can be on any magnetic support with appropriate capacity. In any event, the correct path must be established.

Chapter 2 Page, field, variable and areas: relationships

This chapter consists of a total of 4 pages.

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. In the case of Touch Screen models the term TOUCH BUTTON is also used. The example we give two casually chosen products to simplify the explanation of these concepts.



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.

We use the word touch BUTTON to refer to a page area that can be assigned to and made to represent a virtual button to which particular functions can be attributed.

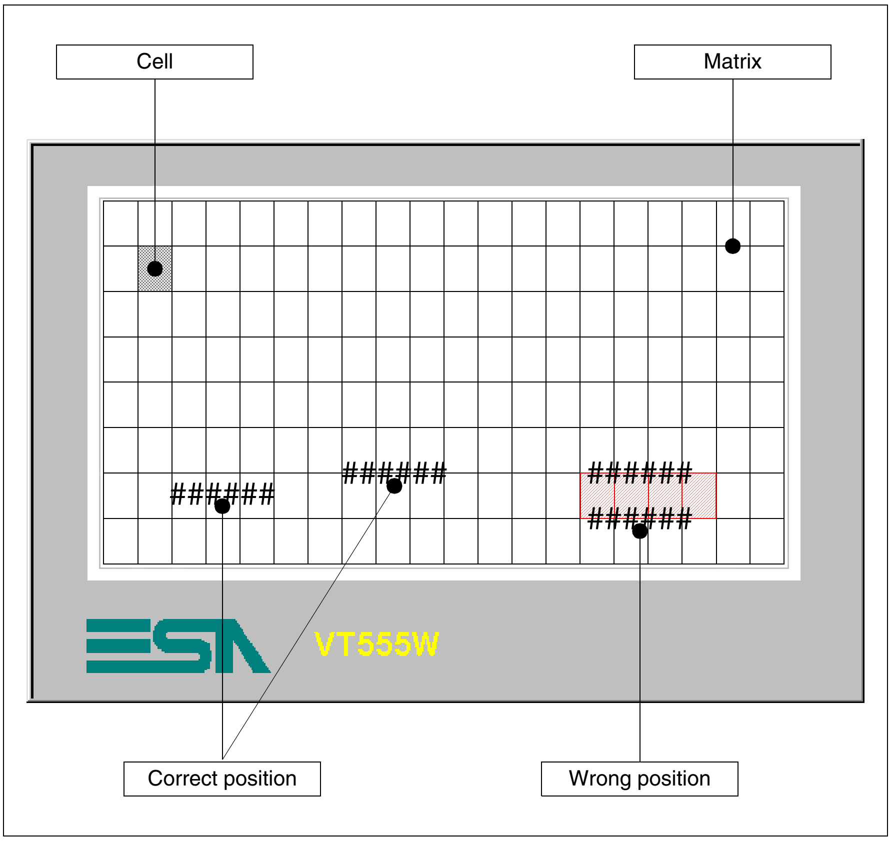
In the case of non-Touch Screen models, the equivalents of the touch buttons are the “F” and/or “E”-keys.

Positioning a field within the page

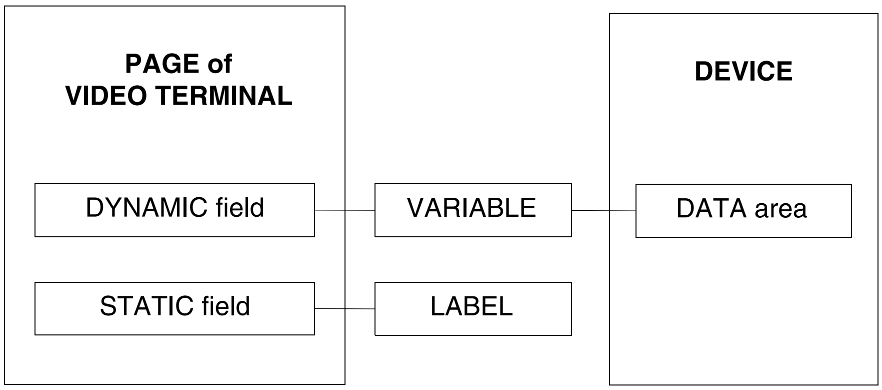
There is no particular problem with the positioning of a field within the page of a terminal when using a keyboard, whether in the case of a “Modify Field Enabled” or a read-only field. None of the above applies to models with a touch screen.

In the case of these latter models, the positioning of fields like “Modify Field Enabled” is of key importance, if we are to avoid errors being generated at the compilation stage.

Instead of a keyboard, touch screen models use a touch-sensitive screen, divided into rows and columns that between them constitute a *Matrix.* To be correctly placed with “Modify Field Enabled” there should be no overlapping of fields in the same touch-sensitive area.

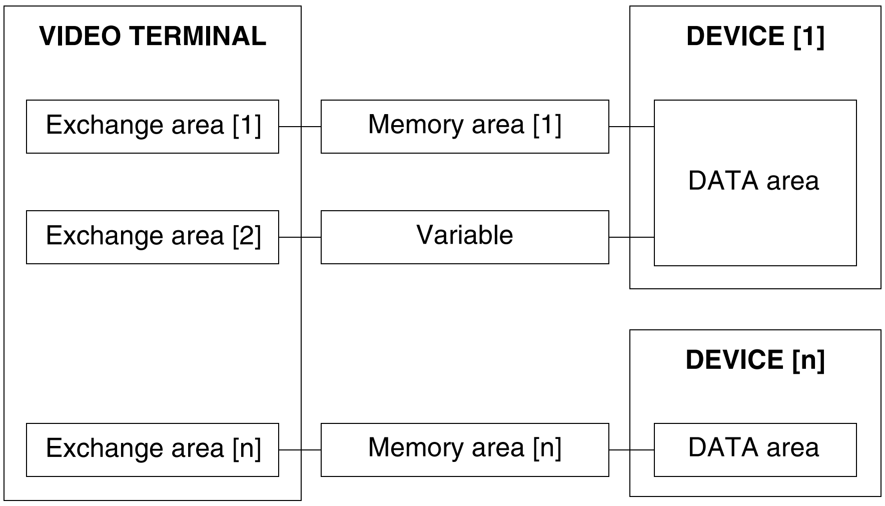


Relationship between fields and variables



The VARIABLE enables the user to assign a data in the device connected to a field. By device we mean any apparatus connected to the terminal or the terminal itself when the internal Registers are used.

Relationship between exchange area and memory area



Using THE MEMORY AREA the data in the connected device can be assigned to the EXCHANGE AREA and/or Variable of the VT.

Chapter 3 VTWIN menus

This chapter consists of a total of 2 pages.

Types of menu

Before setting out the menus with their respective meanings, we need to establish that VTWIN consists of two parts: the first (Configurator) allows the operator to configure a single panel or the ESA-NET network, the other part (Editor) allows you to create the project real and proper. For this reason the menus are called for simplicity Configurator menu and Editor menu.

Chapter 4 The functions in detail

This chapter consists of a total of 118 pages.

This chapter consists of a total of 118 pages.

Page

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

Pages can be of two types: Text and Graphic (in the case of VTs that provide this facility). The first type of page, as implied by the name, only allows the use of alphanumeric characters and symbols; with the other type of page, however, images and drawings can also be used.

The maximum number of pages that can be created depends on the type of VT being used (See relevant Hardware manual).

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.
- Background color
  Used to select the background color of the page.
- Page help
  Supplementary information of help to the user and visible on the VT.
- Edit mode
  Allows the user to access automatic setting of the field following the one currently being set (applicable only to VTs with keyboard).

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

- Multilanguage label
- Numeric field
- ASCII field
- Dynamic text field
- Bar field
- Symbolic field
- Mobile symbolic field
- Hour/Date field
- Trend
- Indicator
- Sliding selector
- Sliding potentiometer
- Selector knob
- Potentiometer knob
- Touch button
- Touch area
- Line
- Rectangle
- Ellipse
- Arc
- Bitmap image

⚠ **These elements have been listed in the same order as they appear in the VTWIN 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.

The field *Multilanguage label* cannot be displayed with a transparent background.

A *Multilanguage label* field can be displayed in Reverse (the background and foreground colors are inverted).

The field *Multilanguage label* can be displayed in Blinking mode (For Blinking see "Chapter 7 -> Edit > Colors").

The minimum resolution for positioning the label depends on the type of VT being used. That is:

• Graphic terminals	->	1 pixel
• Text terminals	->	1 character

The field *Multilanguage Label* can have assigned to it any of the fonts in that language (see "Chapter 6 -> 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

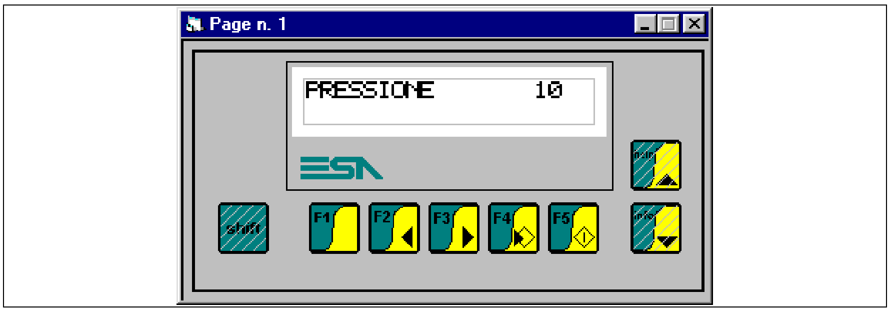
All textual information contained in a project has, for each language configured (see "Chapter 6 -> Project language") a string of characters that defines how such information should be represented.
From now on all textual information of this type is called a Multilanguage text, while the string of characters 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).

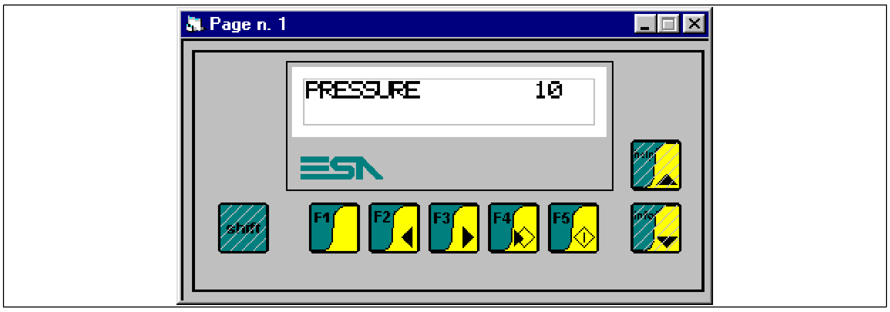
Example.

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.

Numerical Field can be displayed with a transparent background.

A *Numerical field* can be displayed in Reverse (the background and foreground colors are inverted).

A *Numerical Field* can be displayed in Blinking mode.

A *Numerical Field has a Threshold* parameter, making it possible to change the background and/or foreground colors using variables in the connected device.

A *Numerical Field* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after confirming the setting.

The minimum resolution for positioning the field depends on the type of VT being used. That is:

• Graphic terminals	->	1 pixel
• Text terminals	->	1 character

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:

🗀 General options.

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 Page 4-52 -> “Variables“).

Variable (✤):

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

Leading zeros:

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

Example.

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

☑ Leading zeros	Display format:	000100
☐ 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, hexadecimal, floating point or fixed point.

Example.

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

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

The floating point format, as the name implies, makes it possible to display values without predetermining the position of the decimal point, this adapting itself to the value to be displayed.

The floating point format is nothing other than the representation of the result of a calculation executed within the VT using a variable (whole number) of the connected device and the linear scaling of the VT (see Page 4-58 -> “Linear scaling:“).

This format is only valid in Read mode, in Write a whole value is discarded in any case.

The floating point format is very useful when the VT has connected to it a device which does not only use whole numbers for data exchange, but also numbers in a table.

We advise limiting the use of this function to applications like that illustrated in the following example.

Example.

Let us suppose we re connecting a device dedicated to controlling the frequency of a motor, and that the display of this device shows the value of the frequency using a non-whole number (0.125Hz), but in reality the frequency value is a whole number (1Hz). In other words, the device carries out a multiplication between the real value and 0.125.

To achieve the same kind of data display on the VT terminal, just set the linear correction as follows:

Minimum on terminal (on display) = 0.125
Maximum on terminal (on display) = 1250
Minimum in device = 1
Minimum in device = 1000

The possibility of setting minimum and maximum values with a decimal point depends, as far a the linear scale is concerned, on the type of VT being used (See Hardware Manual).

The fixed decimal point format, as the term implies, allows values to be represented with the position of the decimal point already predetermined. This function is applicable exclusively to Floating Point data (see Page 4-54).

By defining the number of digits after the decimal point you can round the displayed number up/down.

Example.

Supposing you have a Floating Point data item that contains a certain number and that you want to limit the number of digits after the decimal point to two. The value displayed will be the following:

Real data = 1.1199999999
Displayed data = 1.12

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 VT: 30
Value transferred to device: 300

Digits after the decimal point:

The number of digits to be displayed after the decimal point can be defined. This field appears only if you use the fixed decimal point numerical format.

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.

Separation characters are not admissible in the numerical format *Floating Point* and *Fixed Point*.

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.

🗀 Mode.

Field Index:

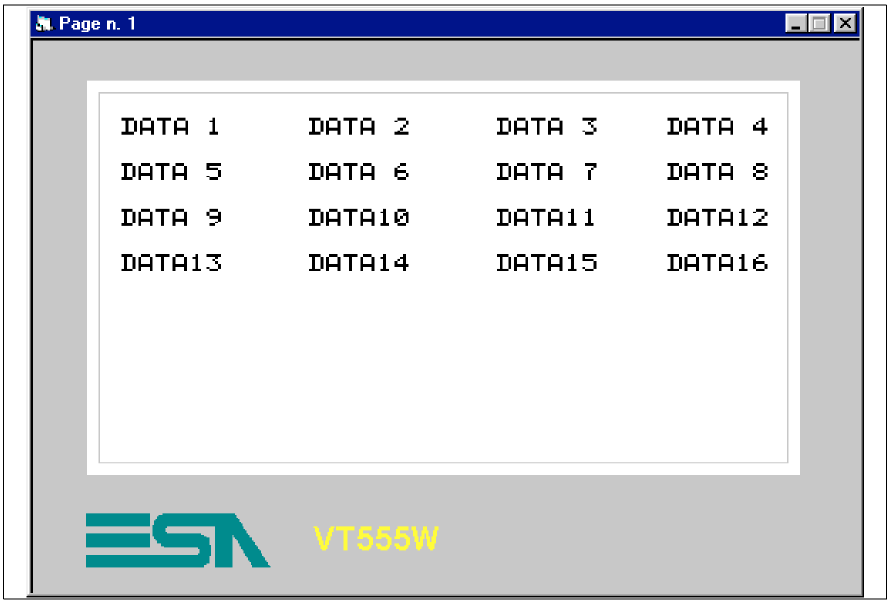
For the sequence followed by the cursor positioning itself on the settable data. The positioning follows an ascending order, that is, from the lowest index to the highest. The key to the order is Index - Row - Column.

Example.

We enter 16 read/write data items from DATA 1 to DATA16, and assign indices as follows:

Data	Index
1	0
2	1
3	0
4	2
5	0
6	2
7	2
8	2

Data	Index
9	3
10	4
11	5
12	5
13	6
14	7
15	4
16	1



DATA 1	DATA 2	DATA 3	DATA 4
DATA 5	DATA 6	DATA 7	DATA 8
DATA 9	DATA10	DATA11	DATA12
DATA13	DATA14	DATA15	DATA16

Arrange data as in figure.

The cursor positions will be in the following order:

Data 1-3-5	(Index 0)	Data 2-16	(Index 1)
Data 4-6-7-8	(Index 2)	Data 9	(Index 3)
Data 10-15	(Index 4)	Data 11-12	(Index 5)
Data 13	(Index 6)	Data 14	(Index 7)

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 VT 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 4-3 -> “Refresh time“) 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 VT
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 VT
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 VT.

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 VT. 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 Page 4-62 -> “Exchange areas“).

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.

Password:

This makes it possible to assign a level of protection to the field, linked to the introduction of a security code. (See “Chapter 6 -> Password“)

🗀 Threshold.

Type:

This allows you to assign the type of threshold required: none, single, double or bit - single threshold.

Example.

Suppose we wish to display the temperature value of a blast furnace in a steelworks such that the color of the temperature displayed changes as its value changes, specifically: black when the temperature is normal (0-2000°C), yellow when the temperature is too high (2001-2500°C) and red when the temperature level is critical (> 2501 °C); the background should be white in all cases. To achieve this the type to be chosen is “double threshold”, which makes it possible to set the variable for controlling the color.

Source:

See Numerical Field Page 4-6.

Variabile (✤):

See Numerical Field Page 4-6.

Threshold:

The value contained in this parameter takes on a different meaning according to the type of threshold that has been selected.

Single and double threshold:

Using this, a threshold value can be assigned beyond which the foreground and/or background color will change. The thresholds become effective when the value of the variable is greater than the value set.

Example.

In the context of the previous example (Page 4-13 -> “Type:“) the values to be inserted are: for threshold #1 - 2000; for threshold #2 - 2500.

Bit-structured single threshold:

Using this, you can assign the number of the bit to be checked to change the foreground and/or background color.

Example.

If this parameter is set at 3, it means that when bit 3 of the threshold variable has a logical status of 1 the color will change.

Foreground:

This allows you to assign the threshold value beyond which the foreground and/or background colors should change. The thresholds intervene when the value of the variable is greater than the value set.

Example.

Basing ourselves on the preceding example (Page 4-13 -> “Type:“), the values to be set are: threshold #1: 2000, threshold #2: 2500.

Background:

This allows you to assign the background colors for the numerical data to coincide with the change in the value of the threshold variable.

Example.

Basing ourselves on the same example (Page 4-13 -> “Type:“) the color to be set to obtain the desired effect is white for the first, second and third areas.

Blinking:

There are three types of blinking: foreground only, permitting the display/hiding the foreground object; background only, permitting the display/hiding of the background object; and foreground plus background, permitting an invertion of colors between the two planes.

Preview:

See Numerical Field Page 4-15.

🗁 Automatic Operation.

Enabled:

Used to handle the function.

Event:

Used to select which event is to activate the function.

Automatic operation:

Used to define which mathematical operation to perform once the defined triggering event occurs.

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* cannot be displayed with a transparent background).

An *ASCII Field* can be displayed in Reverse (with background and foreground colors inverted).

An *ASCII Field* can be displayed in Blinking mode.

A *ASCII Field* has a *Threshold* parameter, making it possible to change the background and/or foreground colors using variables in the connected device.

An *ASCII Field* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after confirming the setting.

The minimum resolution for positioning the label depends on the type of VT being used. That is:

• Graphic terminals	->	1 pixel
• Text terminals	->	1 character

The *ASCII Field* can have assigned to it any of the fonts in that language (see “Chapter 6 -> 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.

🗁 General options.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

See Numerical Field Page 4-6.

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 4-9.

🗁 Mode

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗀 Threshold.

Type:

See Numerical Field Page 4-13.

Source:

See Numerical Field Page 4-13.

Variable (✤):

See Numerical Field Page 4-13.

Threshold:

See Numerical Field Page 4-13.

Foreground:

See Numerical Field Page 4-14.

Background:

See Numerical Field Page 4-14.

Blinking:

See Numerical Field Page 4-14.

Preview:

See Numerical Field Page 4-15.

🗀 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

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 4-92 -> “Text Lists“).

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* cannot be displayed with a transparent background.

A *Dynamic Text Field* can be displayed in Reverse (with background and foreground colors inverted).

A *Dynamic Text Field* can be displayed in Blinking mode.

A *Dynamic Text Field* has a *Threshold* parameter, making it possible to change the background and/or foreground colors using variables in the connected device.

A *Dynamic Text Field* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after the settings have been made.

The minimum resolution for positioning the field depends on the type of VT being used. That is:

• Graphic terminals	->	1 pixel
• Text terminals	->	1 character

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

Text 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.

🗀 General options.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable:

See Numerical Field Page 4-6.

Text lists (✤):

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

🗁 Type.

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.

🗁 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗁 Threshold.

Type:

See Numerical Field Page 4-13.

Source:

See Numerical Field Page 4-13.

Variable (✤):

See Numerical Field Page 4-13.

Threshold:

See Numerical Field Page 4-13.

Foreground:

See Numerical Field Page 4-14.

Background:

See Numerical Field Page 4-14.

Blinking:

See Numerical Field Page 4-14.

Preview:

See Numerical Field Page 4-15.

🗁 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Bar field

A *Bar Field* is defined as one permitting the representation of a variable in a bar graph format.

Bar Fields are dynamic fields relating to a numerical variable.

Bar Fields can be represented in a bottom -> top, left -> right, top -> bottom or right -> left direction.

A *Bar Field* cannot be displayed with a transparent background.

A *Bar Field* can be displayed in Reverse (with background and foreground colors inverted).

A *Bar Field* can be displayed in Blinking mode.

A *Bar Field* has a *Threshold* parameter, making it possible to change the background and/or foreground colors using variables in the connected device.

A *Bar Field can* activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after the settings have been made.

The minimum resolution for positioning the field is 1 pixel.

A *Bar Field* has 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.

🗁 General options.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

This is the variable of the device connected to which the field relates; minimum and maximum limiting values must be assigned to this variable.

Screen:

Use this to choose how the bar is to be displayed.
(Left -> Right, Right -> Left, Top -> Bottom, Bottom -> Top)

Field Index:

See Numerical Field Page 4-9.

🗁 Mode.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗁 Threshold.

Type:

See Numerical Field Page 4-13.

Source:

See Numerical Field Page 4-13.

Variable (✤):

See Numerical Field Page 4-13.

Threshold:

See Numerical Field Page 4-13.

Foreground:

See Numerical Field Page 4-14.

Background:

See Numerical Field Page 4-14.

Blinking:

See Numerical Field Page 4-14.

Preview:

See Numerical Field Page 4-15.

🗁 Display.

Direction:

With this you can choose the display mode of the bar.
(Left -> Right, Right -> Left, Top -> Bottom, Bottom -> Top)

Scale type:

With this you can choose which kind of scale to display in relation to the bar.

Number of values:

With this you can choose the number of values to be displayed in the scale.

Number of notches:

With this you can choose the number of notches to be displayed in the scale between one value and another.

Background:

With this you can assign the color in which to display the background of the bar field.

Range of movement:

With this you can assign the color in which to display the area used for the range of movement of the bar.

Bar:

With this you can assign the color in which to display the bar.

Values:

With this you can assign the color in which to display the values of the scale.

Type:

With this you can assign the color the scale should take on according to the value assigned to the interval (see Numerical Field Page 4-13).

Preview:

See Numerical Field Page 4-15.

🗁 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Symbolic field

A *Symbolic field* is that field which allows the user to represent binary data as an image (dynamic bitmap). Everything we have said about *Dynamic Text fields* applies to this type of field, apart from the fact that if the value of the variable assigned to the list of images (see Page 4-93 -> "Lists of images") does not identify a valid image, the display will show the word **'ERROR'**.

Even if the examples given are still conceptually valid, we should now refer to images and no longer to texts.

🗁 General options.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable:

See Numerical Field Page 4-6.

Lists of images (✤):

Allows you to select which list of images to assign to the variable.

🗁 Type.

Type:

Allows you to select the how the dynamic image is to be displayed.

First bit:

See Dynamic text field Page 4-21.

Last bit:

See Dynamic text field Page 4-21.

🗁 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗁 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Mobile symbolic field

A *Mobile Symbolic Field* is a field that enables you to represent binary data as images (dynamic bitmaps); in addition, using a variable in the connected device, it allows the image to be moved to an preselected by the user.

With a *Mobile Symbolic Field* there can be no transparency, so the area to which the image is moved cannot contain background images.

All explanations regarding *Dynamic Text Fields* applies equally to *Mobile Symbolic Fields,* except that if the value of the variable assigned to the list of images (see Page 4-93 -> “Lists of images“) does not identify a valid image, the screen displays the message **‘ERROR’**.

Although the examples given are still conceptually valid, it is necessary to refer to images here and not texts.

🗁 General options.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable:

See Numerical Field Page 4-6.

🗁 Type.

Image list (✤):

You can select which list of images to assign to the variable.

Type:

You can choose the type of dynamic image.

First bit:

See Dynamic text field Page 4-21.

Last bit:

See Dynamic text field Page 4-21.

🗁 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗁 Movement.

Movement directions:

This parameter determines the direction of the movement of the dynamic symbol: horizontally and vertically, only horizontally or only vertically.

Source (Horizontal variable):

This parameter allows you to set the origin -- either device or data memory -- of the variable for moving the image horizontally. (see Page 4-52 -> “Variables“).

Variable (Horizontal variable) (✤):

This is the variable referred to by the field.

Source (Vertical variable):

This parameter allows you to establish the origin -- either device or data memory -- of the variable for moving the image vertically. (see Page 4-52 -> “Variables“).

Variable (Vertical variable) (✤):

This is the variable referred to by the field.

🗀 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Hour/Date field

The *Hour/Date field* is used to display the time and the date.

The *Hour/Date field* is a non-editable field relating to the clock and calendar inside the VT.

An *Hour/Date Field* cannot be displayed with a transparent background.

An *Hour/Date Field* can be displayed in Reverse (with background and foreground colors inverted).

The *Hour/Date Field* can be displayed in Blinking mode.

The minimum resolution for positioning the field depends on the type of VT being used. That is:

• Graphic terminals	->	1 pixel
• Text terminals	->	1 character

The *Hour/Date field* can be displayed in the form of Time short, Time long, Date and Day of the week.

Time short:

The time is shown in the form ##:##. This type of display exists in two different formats.

24 hour format HH:MM

HH	->	00 … 23	Hour
MM	->	00 … 59	Minutes

12 hour format HH:MMx

HH	->	01 … 12	Hour
MM	->	00 … 59	Minutes
x	->	a ; p	

Where [a] is the abbreviation of the word antemeridian which means before midday (00:00 to 11:59), while [p] is the abbreviation of the word post-meridian which means after midday (12:00 to 23:59).

Time long:

Conceptually this is just like Time short but allowing in addition seconds to be displayed (HH:MM:SS or HH:MM:SSx).

Date:

Is represented as ##/##/####. This type of display exists in two different formats.

Format DD/MM/YYYY

DD	->	01… 31	Day
MM	->	01…12	Month
YYYY	->	n…2096	Year

Format MM/DD/YYYY

The same as the previous format but with the inversion of the position of DD and MM.

Day of the week:

The day of the week is displayed. This type of display exists in two different formats.

Format 1:

Shows the days of the week in numerical format from 0 to 6 (0=Sunday … 6=Saturday).

Monday Format:

Shows the days of the week taking textual information from a list of assigned dynamic texts. The list starts from Sunday and ends with Saturday. If the list is not compiled in the way described the display of the days will be wrong.

Trend

A *Trend* is that field that allows a variable to be represented in graphic format, showing the pattern of the value of a variable over time. The individual graph line related to a variable's value is called the Channel. A *Trend* may contain more than one channel (see Hardware Manual).

A *Trend Field* is a dynamic field that relates to a numerical variable.

A *Trend Field* can be represented in linear mode [∿], one pixel point [▪], an X point [⨯], a +point [✚], X and + points [✱] and an O point [☐].

A *Trend Field* cannot be displayed with a transparent background).

A *Trend Field* can be displayed in Reverse (with background and foreground colors inverted).

The minimum resolution for positioning the field is 1 pixel.

The *Trend Field* has various parameters that must 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 4-6.

Comment:

See Numerical Field Page 4-6.

Channels:

It is possible to specify the number of channels to display in a trend.

(Regarding the maximum number of channels see Hardware Manual). The channel called Reference channel determines the number of readings of all the channels.

Trend buffer (✣):

This allows one of the trend buffers present in the project to be assigned.

Number of samples:

Determines the number of values to be shown at the same time in the trend.

Marker type:

This determines the way the trend is displayed: linear mode [⁄⁄], one pixel point [·], an X point [✕], a +point [✚], X and + points [✱] and an O point [□].

Color (Marker type):

Determines the color for representing the channel.

Upper limit:

Allows the operator to fix the upper limit of the channel, beyond which the display is truncated (see following example).

Color (Upper limit):

Defines the color for that part of the channel that exceeds the upper limit (see following example).

Lower limit:

Allows the operator to fix the lower limit of the channel, below which nothing is displayed (see following example).

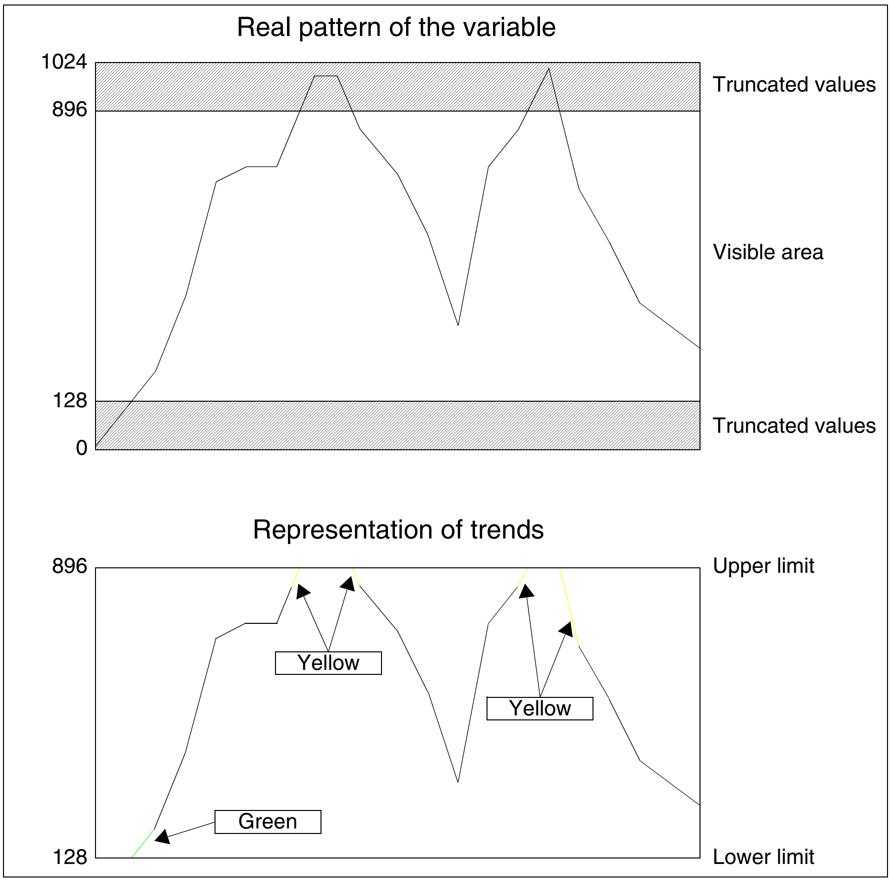
Color (Lower limit):

Defines the color for that part of the channel that exceeds the lower limit (see following example).

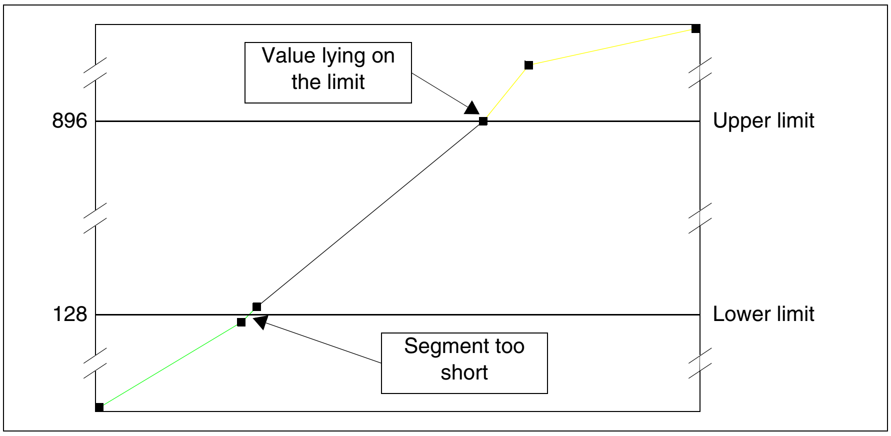
Example:

Suppose we have a single-channel trend that displays the trend buffer of a variable having a value between 0 and 1024. The channel is represented using a black line, below the lower limit of the trend (fixed at a value of

128) it is green, while above the upper limit (fixed at a value of 896) it is yellow. (In the example the number of samples is omitted because it is irrelevant to the understanding of limits).



The color change at the limits will not be visible if the variation in value between one sample and another is not such as to generate a sufficiently long segment or, alternatively, if one of the two values that might generate the segment lies on the limit.



Touch Button

A *Touch Button* is the name given to that field that permits the user to create on the screen a predefined rectangular form in which an identifying label or image can be introduced; this field can also have functions (Page 4-82 -> "Functions assignable to F and/or E keys and to touch buttons:") or direct commands (Page 4-89 -> "Direct Commands") assigned to it.

For minimum resolution see Page 4-78 -> "Touch button".

Touch Buttons have various parameters assigned to them which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Project touch buttons (✤):

A button to be inserted in the page can be chosen from those in the list of touch buttons.

Touch area

By *Touch area* we mean that invisible and transparent rectangular area to which you can assign functions (Page 4-82 -> "Functions assignable to F and/or E keys and to touch buttons:") or direct commands (Page 4-89 -> "Direct Commands").

The minimum dimension of a *Touch area* is one touch screen cell (See Hardware Manual - Technical Characteristics of Touch screen).

A *Touch area* can have a password level assigned to it.

A *Touch area* is valid only for the page in which it is defined.

A *Touch area* can have superimposed on it solely read-only fields and objects.

A *Touch area* has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

Name:

The Name defines the area. It is best to assign it in such a way as to make it easy to recognize and make its significance clear to the programmer.

Comment:

A comment can be assigned, which, if possible, should be the full explanation of the function and meaning of the area, but can also be an alphanumeric character sequence.

Function:

The area can have assigned to it one of the functions listed (Page 4-82 -> “Functions assignable to F and/or E keys and to touch buttons:“) or direct commands (Page 4-89 -> “Direct Commands“).

Function specification (✤):

Specifies the mode of operation of the selected function.

Example.

If a macro function has been chosen, the paramter allows the user to define which macro to activate.

Object:

Further sub-list within the function specified.

Password:

See Numeric Field Page 4-13.

Line

A *Line* is an unbroken succession of pixels (points) that give rise to a linear form, so a line cannot be curved.

A *Line* is a static field and is not assigned to any variable in the device.

Lines can be in any direction, horizontal, vertical or oblique; the minimum resolution needed for positioning and drawing them is 1 pixel.

A *Line* can be displayed in Blinking mode.

A *Line* cannot exceed the area of display.

Rectangle

A *Rectangle* is an unbroken succession of pixels that gives rise to a quadrilateral figure.

A *Rectangle* is a static field and is not assigned to any variable in the device.

The minimum resolution needed for positioning and drawing a *Rectangle* is 1 pixel.

The dimensions of the *Rectangle* cannot exceed the area of display.

A *Rectangle* can be displayed in Blinking mode.

A *Rectangle* can have assigned to it an infill attribute.

Ellipse

An *Ellipse* is an unbroken succession of pixels that gives rise to a circular form.

An *Ellipse* is a static field and is not assigned to any variable in the device.

The minimum resolution needed for positioning and drawing an *Ellipse* is 1 pixel.

The dimensions of the *Ellipse* cannot exceed the area of display.

An *Ellipse* can be displayed in Blinking mode.

An *Ellipse* can have assigned to it an infill attribute.

Arc

An *Arc* is an unbroken succession of pixels that gives rise to a curve.

An *Arc* is a static field and is not assigned to any variable in the device.

An *Arc* can be in any direction, horizontal, vertical or oblique; the minimum resolution needed for positioning and drawing it is 1 pixel.

An *Arc* can be displayed in Blinking mode.

An *Arc* cannot exceed the area of display.

Bitmap Image

A *Bitmap Image* is a field relating to a graphic image.

A *Bitmap Image* is a static field and is not assigned to any variable in the device.

A *Bitmap Image* has assigned to it various parameters that must be compiled. These parameters are listed below.

Project image:

Used to insert images contained in the list of images (Page 4-93 -> “Lists of images“).

Indicator

An *Indicator* is that field that makes it possible to display the value of a variable in a graphic format, having the representational form of an analog indicator.

An *Indicator* is a dynamic field relating to a numeric variable.

An *Indicator* cannot be displayed in a transparent background.

The minimum resolution for positioning the field is pixel.

An *Indicator* has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

🗀 General option.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

This is the variable in the device connected to which the field relates; this variable must have assigned to it minimum and maximum limiting values.

🗀 Display.

Direction:

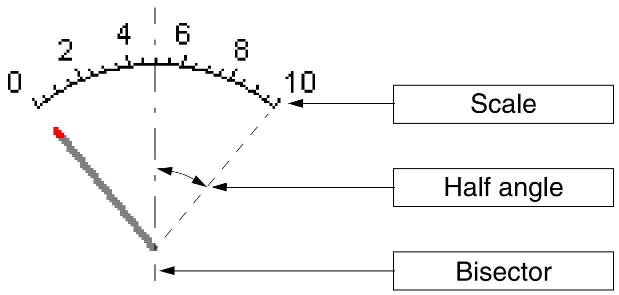
With this you can choose the mode of display of the indicator (Top, Bottom, Right, Left).

Scale type:

With this you can choose which type of scale to display in relation to the indicator.

Half angle:

With this you can choose angle of the aperture (in degrees) of the scale using its bisector as a reference. The maximum aperture of the half angle is 170°.



Number of values:

With this you can choose the number of values to be displayed in the scale.

Number of notches:

With this you can choose the number of notches to be displayed in the scale between one value and another.

Background:

Allows the color in which to display the background of the indicator to be assigned.

Needle:

Allows the color in which to display the needle of the indicator to be assigned..

Notch:

Allows the color in which to display the point of the indicator nearest the scale to be assigned.

Values:

Allows the color in which to display the values in the scale to be assigned.

Type:

Allows the color that the scale should assume according to the value to be assigned to the interval (see Numeric Field Page 4-13).

Preview:

See Numerical Field Page 4-15.

Sliding potentiometer

A *Sliding potentiometer* is a field that makes it possible to display the value of a variable in graphic format.

A *Sliding potentiometer* is a dynamic field that relates to a numeric variable.

A *Sliding potentiometer* cannot be displayed on a transparent background.

A *Sliding Potentiometer* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after the settings have been made.

The minimum resolution for positioning the field is pixel.

A *Sliding potentiometer* has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

🗁 General option.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

See Indicator Page 4-39.

🗀 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗀 Display.

Direction:

See Indicator Page 4-39.

Scale type:

See Indicator Page 4-40.

Number of values:

See Indicator Page 4-40.

Number of notches:

See Indicator Page 4-40.

Background:

See Indicator Page 4-40.

Cursor:

See Indicator Page 4-40.

Notch:

See Indicator Page 4-40.

Values:

See Indicator Page 4-41.

Type:

See Indicator Page 4-41.

Preview:

See Indicator Page 4-41.

Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Sliding selector

A *Sliding selector* is a field allowing a series of values contained in a variable to be displayed in graphic format.

A *Sliding selector* is a dynamic field relating to a numeric variable.

The graphic display of the cursor position occurs via the interpretation of the value contained in a variable and related to the position of the cursor itself.

A *Sliding selector* can have from 2 to 16 positions.

If the value of the variable assigned to the sliding selector exceeds the maximum value, the screen will show a series of characters [!].

A *Sliding selector* cannot be displayed on a transparent background.

A *Sliding selector* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after the settings have been made.

The minimum resolution for positioning the field is pixel.

A *Sliding selector* has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

🗁 General option.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

See Indicator Page 4-39.

🗁 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗁 Positions.

Number of positions:

Used to define the number of positions for the selector.

Positions values:

Used to show the correspondence between the cursor position and the value of the variable.

Selected values:

Used to enter a value to assign to the position of the cursor.

🗁 Display.

Direction:

See Indicator Page 4-39.

Scale type:

See Indicator Page 4-40.

Background:

See Indicator Page 4-40.

Cursor:

See Indicator Page 4-40.

Notch:

See Indicator Page 4-40.

Values:

See Indicator Page 4-41.

Preview:

See Indicator Page 4-41.

🗀 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Potentiometer knob

A *Potentiometer knob* is a field that allows the value of a variable to be displayed in graphic format.

A *Potentiometer knob* is a dynamic field relating to a numeric variable.

A *Potentiometer knob* cannot be displayed on a transparent background.

A *Potentiometer knob* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after the settings have been made.

The minimum resolution for positioning the field is pixel.

A *Potentiometer knob* has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

🗀 General option.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

See Indicator Page 4-39.

🗀 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗀 Display.

Direction:

See Indicator Page 4-39.

Scale type:

See Indicator Page 4-40.

Half angle:

See Indicator Page 4-40.

Number of values:

See Indicator Page 4-40.

Number of notches:

See Indicator Page 4-40.

Background:

See Indicator Page 4-40.

Cursor:

See Indicator Page 4-40.

Notch:

See Indicator Page 4-40.

Values:

See Indicator Page 4-41.

Type:

See Indicator Page 4-41.

Preview:

See Indicator Page 4-41.

🗀 Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Selector knob

A *Selector knob* is a field that allows a series of values contained in a variable to be displayed in graphic format.

A *Selector knob* is a dynamic field relating to a numeric variable.

The graphic display of the cursor position occurs via an interpretation of the value contained in a variable related to the position of the cursor itself.

A *Selector knob* can have from 2 to 16 positions.

If the value of the variable assigned to the selector knob exceeds the maximum value, the screen shows a series of characters [!].

A *Selector knob* cannot be displayed on a transparent background.

A *Selector knob* can activate an Automatic Operation (see Page 4-113) if the Enter key is pressed after confirming the setting.

The minimum resolution for positioning the field is pixel.

A *Selector knob* has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

🗀 General option.

Name:

See Numerical Field Page 4-6.

Comment:

See Numerical Field Page 4-6.

Source:

See Numerical Field Page 4-6.

Variable (✤):

See Indicator Page 4-6.

🗀 Mode.

Field Index:

See Numerical Field Page 4-9.

Continuous read:

See Numerical Field Page 4-11.

One-shot read:

See Numerical Field Page 4-11.

Modify field enabled:

See Numerical Field Page 4-12.

Bit-wise protection:

See Numerical Field Page 4-12.

Bit number:

See Numerical Field Page 4-12.

Password:

See Numerical Field Page 4-13.

🗁 Positions.

Number of positions:

Vedi Sliding selector Page 4-45.

Positions values:

Vedi Sliding selector Page 4-45.

Selected values:

Vedi Sliding selector Page 4-45.

🗁 Display.

Direction:

See Indicator Page 4-39.

Scale type:

See Indicator Page 4-40.

Half angle:

See Indicator Page 4-40.

Background:

See Indicator Page 4-40.

Cursor:

See Indicator Page 4-40.

Notch:

See Indicator Page 4-40.

Values:

See Indicator Page 4-41.

Type:

See Indicator Page 4-41.

Preview:

See Indicator Page 4-41.

Automatic Operation.

Enabled:

See Numerical Field Page 4-15.

Event:

See Numerical Field Page 4-15.

Automatic operation:

See Numerical Field Page 4-15.

Variables

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

The Variable can be configured directly in VTWIN or using one of more files external to VTWIN.

The files that are supported are:
- Instruction list (AWL)
- Symbol list (ASC)
- Comma separated values (CSV)
- Text (TXT)

Variables can only be configured with AWL and ASC files in the case of certain devices (see “Appendix B“**),**

AWL and ASC file generation depend exclusively on the program of the device used (see appropriate manuals supplied by maker).

For the use of CSV and TXT files see “Chapter 6 -> Export to file“ **and/or** “Chapter 6 -> Import from file“**.**

By “device” we mean any apparatus connected to the terminal or the terminal itself when the inyernal Registers are used.

Internal Registers are memory areas inside the terminal that are available to the user.

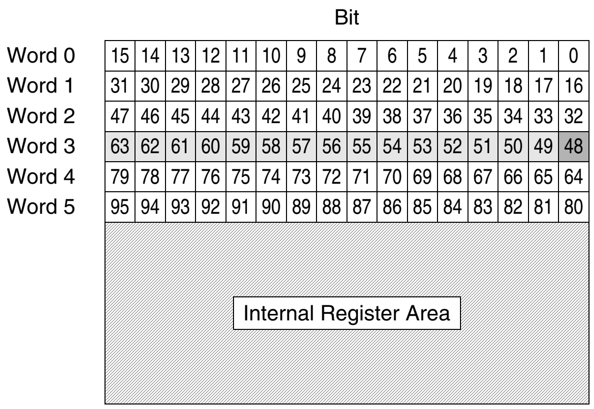
The internal Registers area is NOT retentive, so if the terminal is

switched off the data will be lost even if there is a battery.

The size of the area depends on which terminal is being used (see Hardware Manual) and is the same whether divided into Registers (Word, Dword, String and Floating-Point) or Bits.

Example.

Suppose we define two variables, the first uses internal Register IR3, while the second uses Bit 48 of the internal register area. As can be seen from the table below, Bit 48 corresponds to the first bit of Register IR3 (Word 3).



Word 0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Word 2	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Word 3	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
Word 4	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64
Word 5	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80

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 alphanumeric 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.

VTWIN 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

Floating Point Variables:

This type of variable, as its name implies, can be represented with the decimal point in a non-pre-fixed position, the position varying according to the value of the data held by the device. The format accepted depends on what has been defined (4 bytes). The display resolution depends on the resolution of the calculation by the device.

The display format is decimal (123) not scientific (1.23E2).

Example.

Let us suppose that we want to insert a variable with a 6 character display format: ######

The values are displayed as follows:

Value	Displayed form
23567	23567
2356.7	2356.7
235.67	235.67
2.3567	2.3567

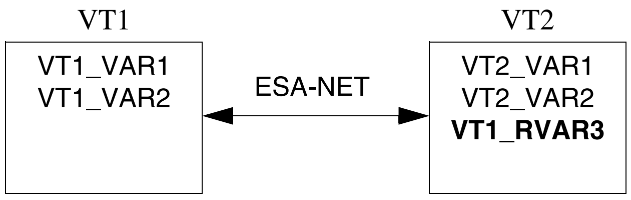
The *Variable* of a terminal can also be shared with other terminals; this means that the information contained in a device directly connected to a VT can be displayed and/or edited by other VTs too, provided they are connected in an ESA-NET network (see Hardware Manual).

The *Variable* shared in this way is called Public Variable (see “Chapter 6 -> Public data“). The maximum number of public variables depends on the terminal (see Hardware Manual).

A *Variable* is referred to as a Remote Variable if declared on one terminal but belonging to another. All this is possible as long as you are in the context of an ESA-NET network project (see Hardware Manual).

A remote variable does not exist physically until it is declared on the other terminal as well.

The remote variable is nothing other than a Public Variable used by other terminals before being declared as such (see “Chapter 6 -> Public data“).



The variable 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.

🗁 General options.

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 alphanumeric character sequence.

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

Source:

This determines to which device to assign the variable.

Importing variables:

This calls up the mask for inserting individual variables into the project, importing them by selecting them from one or more files external to VTWIN. These files must be generated using the programme for managing the project for the device connected to the VT and must be acquired by VTWIN (see "Chapter 5 -> Project for Single VT:" and "Chapter 6 -> Import from file").

⚠ **This function becomes active only if a device is selected from the source list that supports the function (see "**Appendix B"**).**

Data Area:

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

Type:

This selection determines the display mode of the data area: Bit, Byte, Word, Dword, String, Floating point. 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

⚠ **VTWIN 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

🗀 Limits and linear scaling.

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. The value can be assigned using a constant or a numerical variable.

Example:

Minimum = 0
Maximum =100

or

Minimum = VAR001	(containing the value 0)
Maximum = VAR002	(containing the value 100)

⚠ **The variables used for the limits must be numerical and of the same type (e.g. Floating point or whole numbers).**

⚠ **Constant and variable limits cannot be used at the same time.**

⚠ **The variables used for the limits (only in this context) take on the same read mode as the limited variable (one-shot or continuous read).**

⚠ **The variables whose limits are based on other vari-**

ables can only be used with numerical fields.

Linear scaling:

Using this parameter a display value can be attributed that is different from the value actually contained in the device. The value can be assigned using a constant or a numerical variable (see also Page 4-57 -> “Input limits:“).

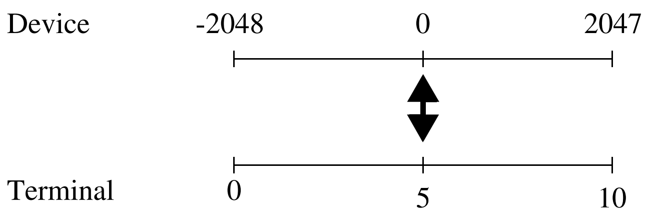
Example.

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.

Variable groups

A *Group of Variables* may be defined as a set of variables that can be read/written individually or as a block, either synchronously or asynchronously (for more details see HoL).

A *Group of Variables* may be also composed of public variables.

The *Group of Variables* of an adapter cannot be shared with other adapters or terminals. This means that the information contained in the group can neither be displayed nor modified even by other ADTs or VTs connected in an ESA-NET network (see Hardware Manual).

The *Group of Variables* has assigned to it various parameters that have to be compiled; some of them are obligatory (✤), others depend on what the user needs to represent. The parameters in question are listed below.

Name (✤):

> Name with which the group is defined. It is advisable to assign a name that makes it easy for the programmer to recognize and understand the significance of the group.

Elements of the group:

> You can specify which of the declared variables must be included in the group.

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 ▢, by using a command from the device connected or, alternatively, as a start-up sequence when the VT 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 interval 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 other 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.

Page Sequences 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.

Switch-on led:

The switching on (or not) of the green led of any of the F □□ and/or E □□.

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.

Memory areas

Memory area is the name given to an uninterrupted zone of memory, defined within the connected device.

Memory areas are necessary if you wish to use the *Data exchange area*, for example, the message area, alarm area, etc.

The abovementioned does not apply to value-structured Alarms and Information Messages: these use the variables directly.

The *Memory areas* can be declared as *Public* and/or *Remote Areas* (See Page 4-52 -> “Variables“ and “Chapter 6 -> Update Public data“).

The *Memory area* 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.

Name:

Name that defines the area. 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 area and its meaning, but it can also be an alphanumeric character sequence.

Source:

You can specify which device to assign the area to.

Data area:

The user can specify which data area of the device is to be defined. The list of data areas depends on the type of device chosen.

Length (✤):

Specifies how many elements should make up the memory area.

⚠ **The length always refers 16bit registers.**
Address (✤):

This field is used to define 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

Exchange areas

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

The *Exchange areas* are structures containing information; which type of information depends on the model of VT being used and 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 latter being in turn sub-divided into:

- External leds command area (Fixed light)
- External leds command area (Blinking light)
- Internal red leds command area (Fixed light)
- Internal red leds command area (Blinking light)
- Internal green leds command area (Fixed light)
- Internal green leds command area (Blinking light)
- Function command area

It is easier to manage the leds using the leds command areas.

The write area updates the device connected only when there is a change of status of area in the VT; this area is called the *Status Area* and is divided into:

- Terminal status area (exchanged every 3-5 seconds independently of the variation in its status)
- Internal ▢▢ status area
- External ▢▢ status area
- Internal ▢▢ status area (Real Time)
- External ▢▢ status area (Real Time)
- Internal led status area
- External led status area
- Recipe status area
- Printer command status area
- Command response status area
- Trend status area

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

Message area:

This area is used by the VT 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* and *Alarm messages* (See Page 4-73 -> "Information

Messages“ and Page 4-75 -> “Alarms“); the length in words of the areas depends on the VT being used.

Command area:

This area is used by the device connected to make the VT carry out certain functions and/or commands.

External leds command area (fixed light):

This area consists of 2 words in binary code that define which led is to be commanded (0 = off, 1 = on).

WORD NUMBER	NAME OF WORD
0	EXTERNAL LED COMMAND
1	EXTERNAL LED COMMAND

External leds command area (blinking light):

This area consists of 2 words in binary code that define which led is to be commanded (0 = off, 1 = on).

WORD NUMBER	NAME OF WORD
0	EXTERNAL LED COMMAND
1	EXTERNAL LED COMMAND

Internal red leds command area (fixed light):

This area consists of 2 words in binary code that define which led is to be commanded (0 = off, 1 = on).

WORD NUMBER	NAME OF WORD
0	INTERNAL RED LED COMMAND
1	INTERNAL RED LED COMMAND

Internal red leds command area (blinking light):

This area consists of 2 words in binary code that define which led is to be commanded (0 = off, 1 = on).

WORD NUMBER	NAME OF WORD
0	INTERNAL RED LED COMMAND
1	INTERNAL RED LED COMMAND

Internal green leds command area (fixed light):

This area consists of 2 words in binary code that define which led is to be commanded (0 = off, 1 = on).

WORD NUMBER	NAME OF WORD
0	INTERNAL GREEN LED COMMAND
1	INTERNAL GREEN LED COMMAND

Internal green leds command area (blinking light):

This area consists of 2 words in binary code that define which led is to be commanded (0 = off, 1 = on).

WORD NUMBER	NAME OF WORD
0	INTERNAL GREEN LED COMMAND
1	INTERNAL GREEN LED COMMAND

Function command area:

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

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

To make the VT 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 VT 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 VT 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 VT 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 VT and the device.

An example.

You want to set the date in the VT as 27 September 1997. 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. 17.

COMMAND “17”: SETDATE

The command SETDATE has 2 parameters and produces an updating of the VT’s calendar 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	SETDATE	
1	GG	MMM
2	AA	AA
3		

Where:
GG = day (in BCD)
MMM = month (in BCD)
AAAA = year (in BCD, 4 digits)

First of all, the necessary parameters are set:

WORD NUMBER	NAME OF WORD	
0	0	
1	27	09
2	19	97
3	0	

After entering the parameters we write the command code:

WORD NUMBER	NAME OF WORD	
0	17	
1	27	09
2	19	97
3	0	

The VT 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.

WORD NUMBER	
17	
27	09
19	97
0	

-> Execution ->

NAME OF WORD	
0	
27	09
19	97
0	

Status area:

This area is used by the VT to inform the device of any change that has occurred in the operational status of the VT or in response to a request command coming from the device connected. The VT 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 VT; word 1 contains the sequence number that appears on the display if the active context is that of Project pages; 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 VT in the event that Project pages is not active.

WORD NUMBER	NAME OF WORD
0	STATUS WORD
1	SEQUENCE IDENTIFIER
2	PAGE IDENTIFICATION
3	FIELD IDENTIFIER

Word 1 is not used in Touch screen models, and word 3 acquires the meaning of word 2 above. There is no field identifier with Touch screen models.

Example.

As in the previous case, the user wants to set the VT date at 27 September 1997. First of all, the user must determine which command to use: this command is 17.

See Page 4-66 -> “COMMAND “17”: SETDATE“

Let us suppose that the device writes the wrong command code (for example 170) in the command word.

WORD NUMBER	NAME OF WORD	
0	170	
1	27	09
2	19	97
3	0	

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

COMMAND AREA

WORD NUMBER	
170	
27	09
19	97
0	

-> Execution ->

STATUS AREA

WORD NAME
0000000000010000
N.U.
0
0000000000000001

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 internal LEDs:

This area is composed of 4 words. These words are in binary code and define the status of the leds. The area is exchanged with the device the moment there is any change in the information contained in it.

WORD NUMBER	NAME OF WORD
0	GREEN LED STATUS WORD
1	GREEN LED STATUS WORD
2	RED LED STATUS WORD
3	RED LED STATUS WORD

Status area for external LEDs:

This area is composed of 2 words. These words are in binary code and define the status of the LEDs. The area is exchanged with the device the moment there is any change in the information contained in it.

WORD NUMBER	NAME OF WORD
0	STATUS WORD
1	STATUS WORD

Status area for recipes:

This area is composed of 1 fixed word. This word is in binary code and defines the status of the recipe transmission carried out by active synchronized transfer. (See “Chapter 6 -> Synchronized recipe transmission:“).

WORD NUMBER	NAME OF WORD
0	RECIPES STATUS WORD

Status area for internal keys:

This area is composed of 6 words. These words are in binary code and define the status of the □□ pressed. The entire area is exchanged with the device when a □ is pressed.

WORD NUMBER	NAME OF WORD
0	OPERATIVE KEYS STATUS
1	OPERATIVE KEYS STATUS
2	OPERATIVE KEYS STATUS
3	OPERATIVE KEYS STATUS
4	FUNCTION KEYS STATUS
5	FUNCTION KEYS STATUS

If more than two keys are pressed simultaneously, the VT no longer perceives the change of state and the area is thus not updated.

The operative keys status area is updated with the bit corresponding to the last key pressed, while the functional keys status area is updated with both bits.

Status area for external keys:

This area is composed of 2 words. These words are in binary code and define the status of the □□ pressed. The entire area is exchanged with the device when a □ is pressed.

WORD NUMBER	NAME OF WORD
0	OPERATIVE KEYS STATUS
1	OPERATIVE KEYS STATUS

If more than two keys are pressed simultaneously, the VT no longer perceives the change of state and the area is thus not updated.

The status area for external keys is updated with both bits.

Status area for internal keys (Real Time):

This area is composed of 6 words. These words are in binary code and define the status of the ▢▢ pressed. The area is exchanged with the device when a ▢ is pressed or released.

WORD NUMBER	NAME OF WORD
0	OPERATIVE KEYS STATUS
1	OPERATIVE KEYS STATUS
2	OPERATIVE KEYS STATUS
3	OPERATIVE KEYS STATUS
4	FUNCTIONAL KEYS STATUS
5	FUNCTIONAL KEYS STATUS

If more than two keys are pressed simultaneously, the VT no longer perceives the change of state and the area is thus not updated.

The operative keys status area is updated with the bit corresponding to the last key pressed, while the functional keys status area is updated with both bits.

Status area for external keys (Real Time):

This area is composed of 2 words. These words are in binary code and define the status of the ▢▢ pressed. The area is exchanged with the device when a ▢ is pressed or released.

WORD NUMBER	NAME OF WORD
0	EXTERNAL KEYS STATUS
1	EXTERNAL KEYS STATUS

If more than two keys are pressed simultaneously, the VT no longer perceives the change of state and the area is thus

not updated. The above is not relevant if the driver with I/O extension is used (for example, VT160W I/O); in this case all the □□ can be pressed simultaneously.

The status area for external keys is updated with both bits, or, in the case of a driver with an I/O extension with all the bits of the keys pressed.

Printer status area:

This area is composed of 2 words. These words are in binary code and define the status of the printer connected. The area is exchanged with the device when there is at least one variation in it. The area is sent to the device before confirming the active print command.

WORD NUMBER	NAME OF WORD
0	ASP STATUS
1	LPT STATUS

Trend status area:

This area is composed of 1 word. These words are in binary code and define the status of the trend. The area is exchanged with the device when there is at least one variation in it.

WORD NUMBER	NAME OF WORD
0	TREND STATUS

Command response area:

This area is composed of 4 fixed words. (numbered from 0 to 3). Word 0 defines the command that the VT must execute, word 1 to 3 are identified as parameter words.

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

Example:

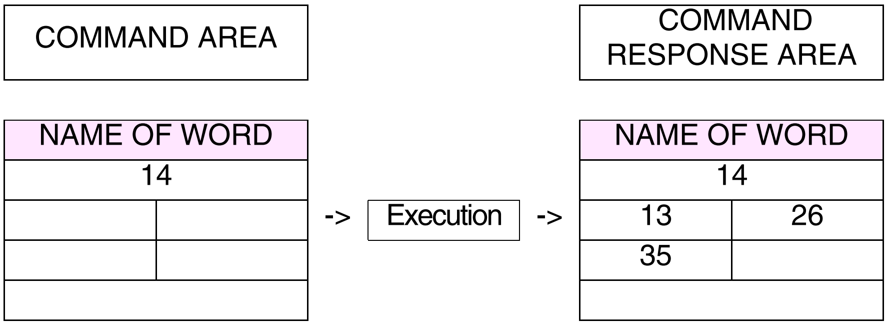
If you want to ask the VT the time, the appropriate command to use is no. 14.

(To understand how to give a command to the VT see example Page

4-66 -> “COMMAND “17”: SETDATE“).

The VT clock indicates 13:26:35.

Once the command has been executed the VT answers to the question



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:

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.

Refresh time:

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

Source:

The user can define which device to assign the data exchange area to.

Memory area:

One of the available memory areas can be assigned.

Information Messages

Information Messages are texts displayed when the device registers an event and communicates it to the VT using the message input area (See Page 4-63 -> “Message area:“). The VT prepares a display context appropriate for messages.

An *Information Message* can contain a Message Field (See Page 4-74 -> “Information Message field:“): this permits the numerical display of the magnitude that has triggered the event of the message. Depending on the model being used, the date and time of the arrival of the message can be given too.

Information Messages can be displayed using any of the project fonts.

When required, the messages can be displayed in rotation automatically (See “Chapter 6 -> Automatic scrolling:“), 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.

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 alphanumeric character sequence.

Exchange area (✤):

> Used to assign the data exchange area. (See Page 4-62 -> “Exchange areas“).

Bit number (✤):

> Indicates the bit to which the information message should be related. (The message is activated when the bit specified is put at status 1). (See Page 4-62 -> “Exchange areas“).

Value (✤):

> Indicates the value to which the info-message should be related. (The message is activated when the assigned value is inserted in the variable).

This function applies exclusively to value-structured Information Messages.

🗁 Message.

Message (✤):

The message to be displayed is edited.

Dimension:

This enables you to establish the size of the character used to display the message.

Preview:

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

🗁 Information message field.

Information Message field:

With this you can assign the numerical variable to be displayed, which contains the value of the measurement that has activated the message.

Example.

Safety pressure exceeded.
4000

🗁 Help message.

Help Message:

Use this to edit the text of the help page.

Font:

Used to choose the font assigned to the language in which the text is displayed.

Preview:

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

🗁 Alarm help button.

Go to page number:

Makes it possible to call up a page by pushing a button.

Button label:

Used to label the ▢ used to call up a page assigned to it.

Preview:

Shows what the button will look like on the display of the VT.

🗁 Print options.

Print Message:

The Information Message is printed as it arrives.

Alarms

Alarms are texts displayed when the device registers an event and communicates it to the VT using the alarm input area (See Page 4-63 -> “Message area:“). The VT prepares a display context appropriate for alarms.

Alarms function in accordance with ISA-1A norms that require the operator to acknowledge the alarm before deactivating it.

An *Alarm* can contain an Alarm Field (See Page 4-77 -> “Alarm field:“): this permits the numerical display of the value that has triggered the event of the message.

When displaying alarms, the VT, besides displaying the descriptive message, gives the date and time for the received event, the departed event and the acknowledged event. Display modes vary with the model of terminal used. If the size of the display permits, all this information is shown in the same page, otherwise more than one page is used.

The *Alarms* have a history file containing the events that have occurred and giving, when set, the date and time of reception [>], acknowledgment [#] and departure [<]. The character in square brackets indicates how these are represented on screen.

When the history is full two procedures are possible:

- FIFO (First In - First Out)
- Ignore exceeding alarms

FIFO (First In - First Out):

> Using this method, the history, when full, automatically eliminates the alarms one at a time as and when new alarms arrive, starting with the first saved.

Ignore exceeding alarms:

> With this procedure, once the history is full, any new alarms are not saved; the buffer can be emptied either by using of an internal command or by using the data exchange (See Page 4-82 -> “Functions assignable to F and/or E keys and to touch buttons:“ e Page 4-62 -> “Exchange areas“).

The events stored in the history buffer can be displayed in chronological order from the most recent to the earliest or viceversa. (See “Chapter 6 -> Ordering alarm buffer starting from most recent alarm:“). The number of alarms depends on the type of VT. (See Hardware Manual).
If the dimensions of the display allow, the date and time are given in the same pagine as the alarm, otherwise on further pages.

An *Alarm* can be displayed in any project font. When required the alarms can be displayed automatically in rotation (see “Chapter 6 -> Automatic scrolling:“), otherwise they are scrolled using the appropriate □□. (see relevant Hardware Manual); the alarms are displayed in chronological order.

Any alarm can have a related Help Page.

As with *Alarms* the related Help Pages can be displayed in any project font. A Help page can recall a project page when a □ is pressed bearing a label defined by the user (Touch Screen models only).

The *Alarms* 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 alarm. 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 alarm and its meaning, but it can also be an alphanumeric character sequence.

Exchange area (✤):

Used to assign the data exchange area. (See Page 4-62 -> “Exchange areas“).

Bit number (✤):

Indicates the bit to which the alarm must relate. (The alarm is activated when the bit specified is put at 1). (See Page 4-62 -> “Exchange areas“).

Value (✤):

Indicates the value to which the info-message should be related. (The message is activated when the assigned value is inserted in the variable).

⚠ **This function applies exclusively to value-structured Information Messages.**

🗁 Alarm.

Alarm (✤):

The alarm to be displayed is edited.

Dimension:

Used to establish the size of the character with which to display the alarm.

Preview:

Shows what the alarm will look like on the display of the VT.

🗁 Alarm field.

Alarm field:

With this you can assign the numerical variable to be displayed, which contains the value of the measurement that has activated the message.

Example.

Safety pressure exceeded.
4000

🗁 Alarm help.

Alarm Help:

Used to edit the Help Page text.

Font:

Use this for choosing the font with which to display the text.

Preview:

Shows what the Help Page will look like on the display of the VT.

🗁 Alarm help button.

Go to page number:

Allows a page to be called up by pressing a button.

Button label:

Allows you label the ▢ used to call up the page assigned to it.

Preview:

Shows what the button will look like on the display of the VT.

🗁 Print options.

Print alarm:

The alarm is printed as it arrives.

Insert in history buffer:

Used to insert the alarm in the alarm history buffer.

Touch button

A *Touch Button* is the name given to that field that permits the user to create on the screen a predefined rectangular form in which an identifying label or image can be introduced; this field can also have functions and/or commands assigned to it.

The minimum dimension of a *Touch button* è 1 pixel.

The minimum space between two *Touch buttons* is a key-width.

A key-width is assumed to be the minimum distance in pixels between one

Touch Button and another.This dimension depends on the type of VT and is a characteristic of the touch screen being used. (See Hardware Manual - Touch screen technical characteristics).

A *Touch Button* has a *Threshold* parameter, making it possible to change the background and/or foreground colors using variables in the connected device, in addition to making it possible to hide and/or disable the button.

The minimum dimension required for inserting a text in a *Touch button* with a surround is 2x2 key-sizes.

A password level can be assigned to a *Touch Button*.

Touch Buttons 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.

🗁 General options.

Name:

Name defining the button. 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 message and its meaning, but it can also be an alphanumeric character sequence.

Password:

See Numerical Field “Chapter 6 -> Password“.

Preview:

Used to see what the ▢ will look like.

▣ Border:

Used to specify whether the border of the ▢ is to be displayed.

Width:

The width of the ▢ is specified.

Height:

The height of the ▢ is specified.

🗀 Function.

Function:

One of the functions listed below can be assigned to the touch button. (See Page 4-82 -> “Functions assignable to F and/or E keys and to touch buttons:“)

Definition of function (✤):

Specifies the operational mode of the function selected.

Example. If the function “macro” is chosen, the parameter allows you to establish which macro to activate.

Object:

Further sub-list in the specified function.

🗀 Button label.

Button label (✤):

Used to insert a multilanguage label for identifying the button. The label can be justified within the button by using the keys for this.

⚠ **Justification of the text is possible only by using Windows based fonts (see** “Chapter 6 -> Project language“**).**

🗀 Background image.

Background image:

Used to insert a background image for identifying the button. The label can be justified within the button by using the keys for this.

⚠ **Justification of the image is possible only by using Windows based fonts (see** “Chapter 6 -> Project language“**).**

Fit button:

Used to automatically adapt the dimensions of the image to those of the button. The proportions of the image can be altered if required.

🗁 Threshold variable.

Source:

See Numerical Field Page 4-13.

Variable (✤):

See Numerical Field Page 4-13.

🗁 Threshold.

Type:

See Numerical Field Page 4-13.

Threshold:

See Numerical Field Page 4-13.

Foreground:

See Numerical Field Page 4-14.

Background:

See Numerical Field Page 4-14.

Blinking:

See Numerical Field Page 4-14.

Image:

Used to assign the images for displaying the background of the button to change in line with the value of the threshold variable.

Hide:

Used to make the button invisible and inactive.

Disable:

Used to make the button inactive but still visible.

Preview:

See Numerical Field Page 4-15.

For non-touch screen models F□□ and certain E□□. Functions can be assigned to the F and/or E□□ and as in the case of touch buttons functions can be assigned. F and E□□ are configurable using ***Page > Definition of internal keys*** and ***Page > Definition of external keys*** (see “Chapter 6 -> Internal keys definition“), or else by clicking directly on the □ of the page of VTWIN displayed in the foreground.

Functions assignable to F and/or E keys and to touch buttons:

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.

Go to page:

Used to call up a defined page from amongst those contained in the project.

Macro:

Used to assign a macro to a specified button. (See Page 4-98).

Internal Command: PASSWORD - Modify password

Allows all passwords to be edited by displaying a series of dedicated page.g. The change is permitted at a given level and those below.
(See “Chapter 6 -> Password“).

Example.

Given 10 password levels from 0 (supervisor) to 9, if in response to the login request the password corresponding to level 4 is introduced, the way is open to changing the passwords from 4 to 9, the other levels are concealed.

Internal Command: PASSWORD - Login password

Used to enter the mode for setting the password relating to the entire project. If the password inserted is contained in the list of VT passwords, then access is given to data protected by passwords of a lower level; alternatively, if the password is wrong a message is displayed and you remain on the same level as before. (See “Chapter 6 -> Password“).

Example.

Given that there are 10 levels of password from 0 (supervisor) to 9 and assuming a page containing 10 items of data each protected by a password such that Data item 1 -> Level 0 and Data item 10 -> Level 9, if the password corresponding to level 4 is inserted at the time of the login request, you will have the possibility of editing data items 5 to 10, while if you try to edit a higher level a data item with a higher level of password a new password is requested.

Internal Command: PASSWORD - Logout password

Used to restore the original level of the password, annulling the function of login password. If the logout page has been defined, after the user’s confirmation response to the system’s message, the VT passes to the defined page. (See “Chapter 6 -> Password“).

Internal Command: PIPELINE - Run pipeline

Enables you to have the specified pipeline executed. (See Page 4-100 -> “Pipelines“).

Internal Command: PRINT - Hardcopy

Used to print out what appears on the display.

Internal Command: PRINT - Print alarm history

Used to print history buffer of alarms.

Internal Command: PRINT - Printer form feed

Causes a sheet to be expelled from the printer whether it is fully printed, partly printed or completely blank.

Internal Command: PRINT - Put total sheet number to 0

Used to reset all the counters of printed pages.

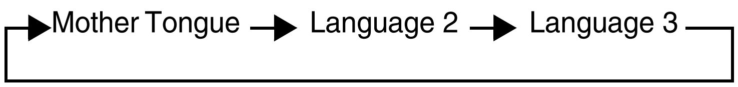
Internal Command: PROJECT - Change language

Use this to change the current project language, substituting one of those declared. The new language remains active even after a new start-up until the language is changed again.

⚠ **The language is not retentive with VT50-VT60.**

Internal Command: PROJECT - Commute language

Used to commute the current project language with one of the declared languages sequentially.



The new language remains active even after a new start-up until the language is changed again.

⚠ **The language is not retentive with VT50-VT60.**

Internal Command: PROJECT - Clear trend buffer

Used to reset to zero the values acquired in whatever mode. (See Page 4-109 -> “Trend buffer“).

Internal Command: PROJECT - Display page directory

This displays a system page with all the pages of the project listed. From this page can be selected the page to be displayed.

Internal Command: PROJECT - Display project information

This displays the project information page.

Internal Command: PROJECT - Display sequence 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: PROJECT - Next Page

Used to call up the first page found with a higher number than the one displayed. (It does not need to be consecutive).

Internal Command: PROJECT - Page Help

Used to call up Page Help.

Internal Command: PROJECT - Previous Page

Used to call up the first page found with a lower number than the one displayed. (It does not need to be consecutive).

Internal Command: PROJECT - Quit project

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

Internal Command: PROJECT - Read trend sample

Used to acquire a reading of the channel(s) of all the trends defined with the mode Single Sample on Command. (See Page 4-33 -> “Trend“).

Internal Command: PROJECT - Read trend block

Used to acquire an entire block of readings saved in the device connected, being readings of all the trends defined with mode Block of Readings on Command. (See Page 4-33 -> “Trend“).

Internal command: PROJECT - Save the alarm history and trend buffer onto a flash

This allows you to save the alarm history into the non-volatile memory of the terminal in the case of those VTs that have no buffered batteries. (See Hardware Manual).

Internal Command: PROJECT - Service Page

Used to call up the status page of the drive; from this point on it is possible, by pressing the allotted ▢, to enter programming mode; from this page you can accede to the contrast control, and the regulation can be effected by pressing the same option then using the arrow-up and arrow-down buttons. When this has been done, all the settings can be saved using the appropriate button. If, on the other hand, the date and time are pressed, you enter date and time setting mode by means of the appropriate ⧠⧠.

Internal Command: PROJECT - Show alarms history

Allows the alarm buffer to be shown on the display.

Internal Command: PROJECT - Start trend

Used to start the reading of the channel(s) of all the trends defined with the mode Single automatic sample. The VT setting is active reading. (See Page 4-33 -> “Trend“).

Internal Command: PROJECT - Stop trend

Used to stop the reading of the channel(s) of all the trends defined with the mode Single automatic sample. The VT setting is active reading. (See Page 4-33 -> “Trend“).

Internal Command: RECIPE - Delete recipe

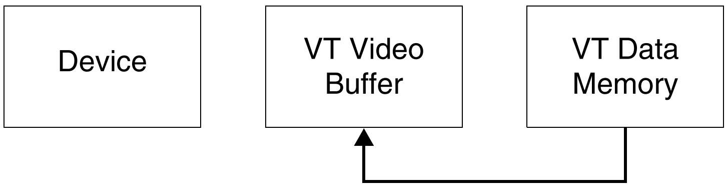
Enables you to delete the recipe specified directly from the VT memory. This is a non-confirm function. If the recipe to be deleted does not exist, a message appears to inform you. (See “Chapter 6 -> Data memory structure“).

Internal Command: RECIPE - Display recipe directory

This displays a system page with all the pages in the memory listed. From this page can be selected the recipe to be displayed or transferred to the device.

Internal Command: RECIPE - Load recipe from memory

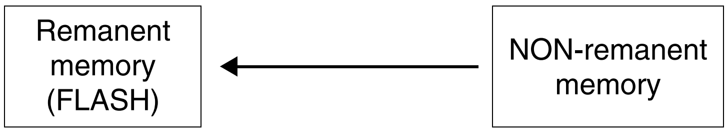
Enables you to load a recipe data from the memory of the VT and display it. (See “Chapter 6 -> Data memory structure“).



Internal command: RECIPE - Copy recipe into remanent memory

Command valid for VT575W - VT585WB - VT595W, allowing all the recipes contained in the volatile memory to be copied into the remanent memory (Flash). It might take some seconds (up to

15 seconds) to copy. (See “Chapter 6 -> Data memory structure“).

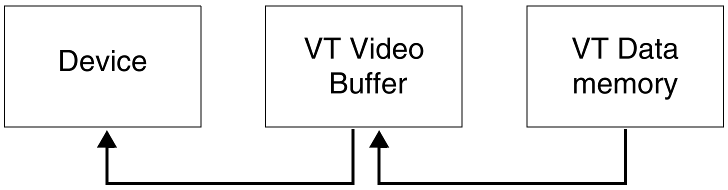


If you choose not to tick the box with the command "Automatically copy recipes into remanent memory” in the "Data memory structure" mask (see “Chapter 6 -> Automatically copy recipes into remanent memory:“) we recommend using this command in comnination with:

• Cancel recipe
• Save recipe received from device into data memory
• Save recipe into data memory

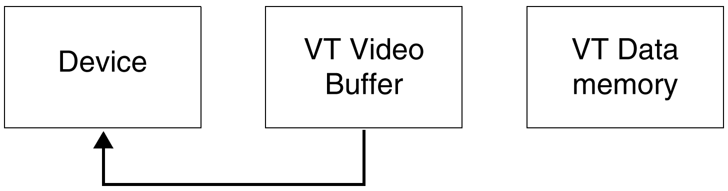
Internal Command: RECIPE - Send recipe to device

Used to transfer a recipe directly from the VT memory to the device connected. In the event that the recipe to be transferred is not in the archive, a message is displayed. (See “Chapter 6 -> Data memory structure“).



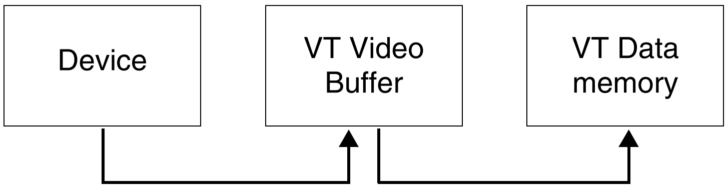
Internal Command: RECIPE - Send recipe from video buffer to device

Used to transfer the recipe being displayed on the VT to the device connected. (See “Chapter 6 -> Data memory structure“).



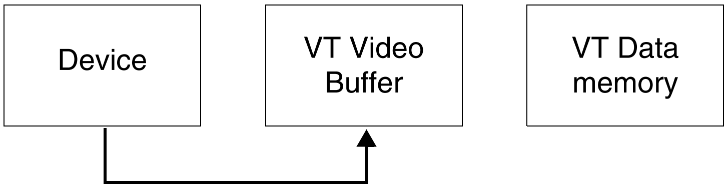
Internal Command: RECIPE - Save recipe from device in data memory

Use this to transfer a recipe directly from the device connected to the VT memory. If a recipe you try to save already exists the VT displays a message of confirmation for the overwrite. (See also Page 4-86 -> “Internal command: RECIPE - Copy recipe into remanent memory“ and “Chapter 6 -> Data memory structure“).



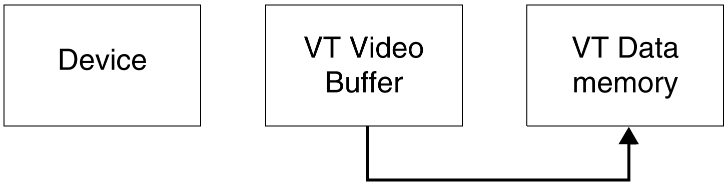
Internal Command: RECIPE - Save recipe from device in the video buffer

Used to transfer a recipe directly from the device connected onto the VT display. (See “Chapter 6 -> Data memory structure“).



Internal Command: RECIPE - Save recipe in data memory

Allows a recipe on the display to be saved in the VT memory. If a recipe you try to save already exists the VT displays a message of confirmation for the overwrite. (See also Page 4-86 -> “Internal command: RECIPE - Copy recipe into remanent memory“ and “Chapter 6 -> Data memory structure“).



Set bit status permanently:

Puts a bit permanently at 1.

Resets bit permanently:

Puts a bit permanently at 0.

Sets the bit in real time:

Puts the bit at 1 as long as the ▢ is depressed.

Resets the bit in real time:

Puts the bit at 0 as long as the ▢ is depressed.

Inverts the value of the bit:

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

Value-structured direct command:

Changes the value of a given variable.

Report:

Permits the related report to be printed.

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 or operating mathematical calculation. Value-structured direct commands affect the entire value of the assigned numerical variable. The commands that are admissible are:

ADD Command

When the ▢ is pressed, the value specified is added to that of the variable assigned and the result is written to the device connected.

Example.

Value of the variable 120, value specified 45. After pressing the ▢, the value 165 is transferred to the device.

SUBTRACT Command

When the ▢ is pressed, the value specified is subtracted from that of the variable assigned and the result is written to the device connected.

Example.

Value of the variable 120, value specified 45. After pressing the ▢, the value 75 is transferred to the device.

AND Command

When the ▢ is pressed an AND logical operation is performed between the decimal value specified and that of the variable assigned and the result is written to the device connected.

Example.

Value of the variable 120, converted into binary format 1111000; value specified 45, converted into binary format 101101. After pressing the ▢ the value 101000 is transferred to the device, converted into decimal 40.

OR Command

When the ▢ is pressed an OR logical operation is performed between the decimal value specified and that of the variable assigned and the result is written to the device connected.

Example.

Value of the variable 120, converted into binary format 1111000; value specified 45, converted into binary format 101101. After pressing the ▢ the value 1111101 is transferred to the device, converted into decimal 125.

XOR Command

When the ▢ is pressed an XOR logical operation is performed between the decimal value specified and that of the variable assigned and the result is written to the device connected.

Example.

Value of the variable 120, converted into binary format 1111000; value specified 45, converted into binary format 101101. After pressing the ▢ the value 1010101 is transferred to the device, converted into decimal 85.

SET Command

When the ▢ is pressed the variable is overwritten with a pre-fixed value.

Example.

Value of variable 120, value specified 45. After pressing the ▢ 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 alphanumeric character sequence.

Source:

Used to specify which device to assign direct commands to.

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.

Operation:

Defines the value-structured command desired.

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 4-18 -> “Dynamic Text Field“)

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 alphanumeric character sequence.

Values:

Used to assign the value to the variable for displaying the text associated in list, in the event that this type of value-structured dynamic text is used.

Texts:

Shows the texts contained in the list.

List of texts selected (✤):

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

Lists of images

The *Lists of images* function is used in the project, like that for texts, to make an image correspond to the value of a numerical variable.

The image lists serve to construct the *Dynamic Bitmaps*.

An image list must contain at least two images.

Lists of Images 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 image 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 image list and its meaning, but it can also be an alphanumeric character sequence.

Valori:

Usedtoassignthevaluetothevariablefordisplayingtheimageassociated in list, in the event that this type of value-structured image list is used.

Images (✤):

Shows the images contained in the list.

image selected:

Used to add images to the list.

Preview:

Displays the image to be added.

Images

Before proceeding with the explanation, it is necessary to point out that what is presented in this paragraph depends on the type of terminal being used and on its hardware revisions. (see Table 4.3 on page 98)

Project Images are graphic images used in the project to draw backgrounds or to be associated to buttons, lists of images, etc.

Project Images can be created with a graphic editing program (E.g. Paint) and may be monochromatic, scale of grey (16 greys), 16 colors or 256 colors.

Project Images with a number of colors in excess of those supported by the terminal in use are automatically resampled to reduce and adapt the colors (“dithering”).

Using “dithering” excellent results can be obtained if the number of colors of the image is reduced from 16.8 million (24bits) or 65,536 (16bits) to 256 colors. A further reduction in the number of colors may alter the image substantially.

If the terminal used does not support 256 colors, it may prove necessary to generate the images that have the correct number of colors, in order to avoid their degenerating in the importation phase.

The Palette of the monochromatic images must be made up as follows:

Bit at 1 -> White
Bit at 0 -> Black

If the settings of the gray tones of the gray scale images are not as follows, theese images will not be displayed properly on the VT.

If the image displayed on the VT has grays that are slightly different from those displayed on VTWIN, you are advised to use the contrast control on the terminal. (See Hardware Manual).

Table 4.1: RGB setting for matching gray tones (Mix 0 - 255) (Part 1 of 2)

Color in VTWIN		Red (R)	Green (G)	Blue (B)
BLACK	1	0	0	0
WHITE	2	255	255	255

Table 4.1: RGB setting for matching gray tones (Mix 0 - 255) (Part 2 of 2)

Color in VTWIN		Red (R)	Green (G)	Blue (B)
GRAY	3	17	17	17
	4	34	34	34
	5	51	51	51
	6	68	68	68
	7	85	85	85
	8	102	102	102
	9	119	119	119
	10	136	136	136
	11	153	153	153
	12	170	170	170
	13	187	187	187
	14	204	204	204
	15	221	221	221
	16	238	238	238

Color images must be set using DOS 16 color Palette if the program used permits; otherwise personalize the colors as follows to have the correct display on the VT.

If the image displayed on the VT has slightly different colors from those displayed on VTWIN, we advise adjusting the contrast of the terminal. (See Hardware Manual).

Table 4.2: RGB setting for color matching (Mix 0 - 255)

Color in VTWIN		Red (R)	Green (G)	Blue (B)
BLACK	1	0	0	0
WHITE	2	255	255	255
DARK RED	3	128	0	0
DARK GREEN	4	0	128	0
DARK YELLOW	5	128	128	0
DARK BLUE	6	0	0	128
DARK VIOLET	7	128	0	128
DARK CYAN	8	0	128	128
DARK GRAY	9	128	128	128
LIGHT GRAY	10	192	192	192
LIGHT RED	11	255	0	0
LIGHT GREEN	12	0	255	0
LIGHT YELLOW	13	255	255	0
LIGHT BLUE	14	0	0	255
LIGHT VIOLET	15	255	0	255
LIGHT CYAN	16	0	255	255

The *Project Images* have a series of functions dedicated to image handling before the definitive importation into VTWIN. These functions are:

Compression in the terminal:

Used to transfer images to the terminal in compressed graphic format. The compression reduces a given image's occupation of the terminal's graphic memory by up to 55%.

Compression entails an increase in the refresh time of the image in the terminal of up to 3 times that necessary for a non-compressed image. Roughly speaking, between 0.5 seconds and 1.5 seconds are needed to refresh an image of 640x480 pixels.

Re-load:

Updates the representation of the original image by reloading it from the source disk.

Cut-out tool:

Allows you to select a part of the image, automatically eliminating the remaining parts.

Re-size:

Allows you to modify the dimensions of the image before it is inserted in VTWIN. This function has a number of subfunctions:
Adapt image to screen:

Allows you automatically to adapt the dimensions of an image to those of the display.

Maintain proportions:

Allows you to maintain the proportions of the image during re-sizing.

Type of resizing:

Allows you to establish which type of algorithm to use for re-sizing the image.

Dimensions:

Allows you to set the new dimensions of the image.

Zoom:

Allows you to enlarge of reduce the image being displayed.

Filters:

Allows you to apply certain filters to improve the display of the image to be imported.

Project Images can have their dimensions changed even after being imported into VTWIN, by selecting the image and pulling its corners (image stretching).

⚠ **Stretching entails an increase in the refresh time of the image in the terminal of up to twice that necessary to refresh a similar re-sizing in the VTWIN. Roughly speaking, between 0.5 seconds and 1.0 second are needed to refresh an image of 640x480 pixels.**

The minimum resolution for positioning *Project Images* is 1 pixel.

Project Images 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 image. 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 image and its meaning, but it can also be an alphanumeric character sequence.

File Name (✤):

Indicates the name of the source image prior to its being imported into VTWIN.

Directories:

Indicates the directory on disk where the source image is located.

List Files of Type:

Indicates the format possible for the source image.

Drive:

Indicates the physical unit where the source image is located.

Preview:

Displays the image that is to be inserted.

The table below lists the different graphic possibilities of VTWIN associating them with the terminals and their various revision.

Table 4.3: Graphic functions correlated with hardware revisions of the VT .

TERMINAL		FUNCTIONS					
Model	**Revision**	**16 Colours**	**256 Colours**	**Stretching image**	**Compres-sion image**	**Images format (see note)**	**Dithering**
VT300W	1	--	--	--	--	●	●
VT310W	1	--	--	--	--	●	●
VT320W	1	●	--	--	--	●	●
VT330W	1	●	--	--	--	●	●
	2	●	--	--	--	●	●
	3	●	--	--	--	●	●
	≥4	--	●	●	●	●	●
VT155W	1	--	--	--	--	●	●
VT185W	1	●	--	●	●	●	●
VT505H	1	--	--	--	--	●	●
VT505W	1	--	--	--	--	●	●
VT515W	1	--	--	--	--	●	●
VT525H	1	●	--	--	--	●	●
VT525W	1	●	--	--	--	●	●
VT555W	1	--	--	--	--	●	●
VT560W	1	--	--	--	--	●	●
VT565W	1	●	--	--	--	●	●
	2	●	--	--	--	●	●
	3	●	--	--	--	●	●
	≥4	--	●	●	●	●	●
VT575W	1	--	●	●	●	●	●
VT580W	1	--	●	●	●	●	●
VT585W	1	●	--	--	--	●	●
	2	●	--	--	--	●	●
	3	●	--	--	--	●	●
	≥4	--	●	●	●	●	●
VT585WB	1	--	●	●	●	●	●
VT595W	1	--	●	●	●	●	●
Note: JPG, CMP, ICO, CUR, PCX, PCD, FPX, DIC, TIF, WMF, EMF, XPM, IFF, PMB, JBG, PSD, PNG, TGA, EPS, RAS, WPG, PCT.							

-- : not available

Images in TIFF format must not be compressed.

Macro

This makes it possible to group together a number of internal functions and/or Direct commands to be carried out in sequence.

Macros are subject to the following limitations.

- The following functions and Direct commands cannot be inserted:
  Direct command “Set bit as real time”
  Direct command “Reset bit as real time”
  Function “Page list”
- The function “Exit from project” can be inserted only as a last-placed function.
- I comandi relativi alla gestione “Ricette” si possono inserire solo come ultime funzioni.

Macros 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 macro. 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 macro and its meaning, but it can also be an alphanumeric character sequence.

Function List (✤):

Indicates the list of functions covered by the macro.

Function:

Used to choose the functions to include in the function list.
(See Page 4-82 -> “Functions assignable to F and/or E keys and to touch buttons:“).

Function Specification:

Specifies the parameter of a generic function.

Example.

If a macro function has been selected, this makes it possible to establish which macro to activate.

Object:

Further sub-list within the function specified.

Pipelines

The *Pipeline* function allows an exchange of information between the different devices via the VT.

The *Pipeline* function can be used with the devices connected to any of the serial ports available on the VT.

When the VT is switched on, the *Pipelines* are always active, each with its own mode of operation.

Pipelines see to the conversion of the format between source variables and destination variables using the following criteria:

- If the source variable has limits and/or mathematical correction assigned to it, these are ignored.
- If the destination variable has limits assigned, no value can be written other than those. In particular if the value to be written is lower than the lowest limit the lowest limit is written; if the value to be written is above the upper limit the upper limit is written. In all other cases the value is written without changes.
- If the destination variable has a mathematical correction assigned, this is applied before the value is written. Before writing, also check that the value is admissible within the limits (if any). If not, proceed as indicated in the previous paragraph.
- If the source variable is in word form and the destination in byte form, the value written will only be valid if the value contained in the source variable is such as to be completely containable within the byte; the sign is also counted.
- If the dimension of the destination variable is greater than that of the source variable, any value read will be correctly sent to the destination device.
- If the source variable has a sign (For example, -52) and the destination variable has no sign a 0 will be written. If, on the other hand, the source value is a positive number, the source value will be written with no variation.
- If the dimension of the source variable is greater than that of the destination variable, only that partof the value that can be contained within the destination variable is written. There follow some examples of source variables in word format, destination variables in bytes:

Source variable (Word)		Destination variable (Byte)	
Decimal	Hexadecimal	Decimal	Hexadecimal
128	80	128	80
1024	400	0	0
1026	402	2	2

- If the source variable is in floating point format and the destination variable is numerical only the whole numbers will be written, bearing in mind the preceeding point.

- If the source variable is numerical and the destination variable is in floating point format, the entire number will be written.
- If the source variable is in ASCII and the destination variable is either numerical or in floating point format, the numerical equivalent of the ASCII value is written to the numerical variable. If the ASCII data is in alphabetical form, the value written to the numerical/floating point variable has no sense.
- If the source variable is numerical and the destination variable is in ASCII, a string with the numerical value read iswritten. If the length of the ASCII variable is such as not completely to contain the number, the string will contain characters that have no sense.

Pipelines can be defined with the following read/write modes:

- Read and write by polling
- Read by polling and write with each change
- Read and write by command

Read and write by polling:

The VT reads the source variable at constant, preset time intervals (from 100ms to 10h), the write of the destination variable occurs immediately after each read.

Read by polling and write with each change:

The VT reads the source variable at constant, preset time intervals (from 100ms to 10h), while the writing of the destination variable only occurs if the value read from the source variable is different from the preceding one. If the value has changed the writing will take place immediately after the read.

The values read from the source variable are saved in the VT in a single buffer (see Hardware Manual) appropriate for all pipelines.

Every pipeline occupies at least 2 bytes; in the case of pipelines with string variables count 1 byte for each character with rounding up to a full word (5 charactere occupy 3 words).

Read and write by command:

The VT reads a source variable and writes the associated destination variable after each internal command or by means of data exchange. (See Page 4-62 -> “Exchange areas“).

The *Pipeline* function 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.

🗁 General options.

Number:

Indicates the identifying number of the pipeline.

Name:

Name defining the pipeline. 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 the explanation in full of the function and the meaning, but may also be an alphanumeric sequence of characters.

Source (source variable):

You can specify from which device to read the variable.

Variable (source variable) (✤):

You can specify to which variable to assign the pipeline.

Source (destination variable):

You can specify to which device to write the variable.

Variable (destination variable) (✤):

You can specify to which variable to assign the pipeline.

🗁 Mode.

Read/write:

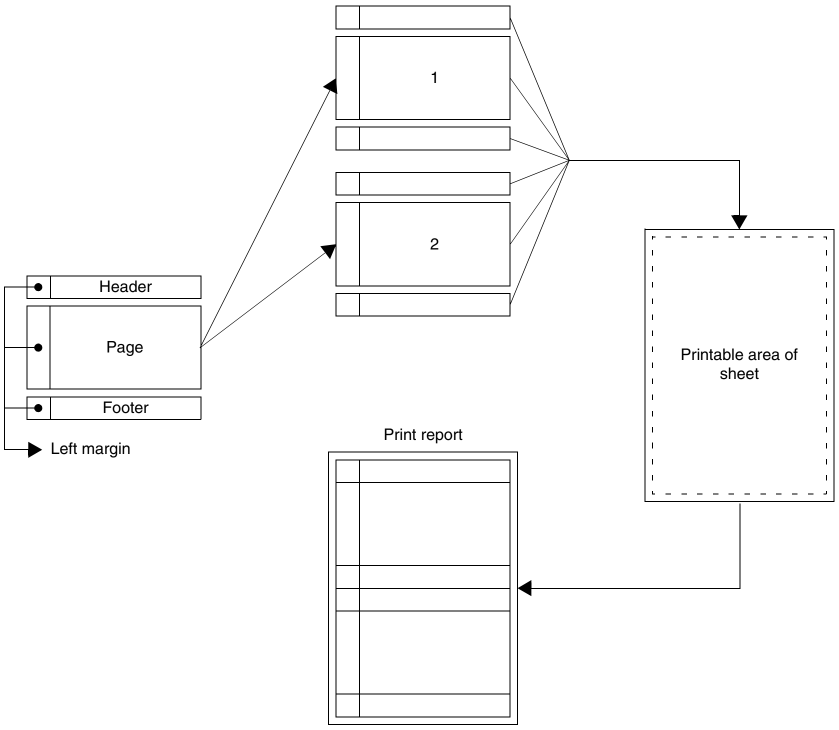
You can specify which mode to use.

Polling time:

Allows the time interval between one read and the next to be specified.

Print page

Before defining what a print page is, we have to make clear in advance that when we speak of a page we do not mean the physical sheet issuing from the printer, but an ensemble of lines that comprise “logical” page. Page counters relate to this type of page and not to physical sheets. For example, let us suppose we have a printer which uses sheets with a printable area of 80 rows; if in the definition of the page we declare that the page is composed of 40 rows, the sheet will take up 2 pages and the counter will register 2 even if physically it was all printed on one sheet.
The diagram below illustrates the relationship between a “logical” page and a physical sheet or print report.



A *Print Page* is a collection of fields that determine the aspect of the document as defined by the user and that can be sent to a printer directly connected to the VT.

Pages **cannot** contain graphical elements like images and drawings.

Only text printing is possible and standard ESA fonts must be used; customized fonts can cause faulty representation in the printing.

Pages have various Attributes, namely:

- Number
  A progressive number identifying a page in the list.

- Name
  A name to define the page so as to make its function easy to recognize.
- Uses predefined parameters
  Used to determine whether the print parameters need to be specified in the page or whether the general parameters of the project should be accepted.
- Rows per Page
  Used to determine the number of printable rows per page if the pre-defined parameters are not used.
- Columns per Page
  Used to determine the number of printable columns per page if the predefined parameters are not used.

Once the print page has been created, any change in this parameter means the user must modify all the multilanguage labels with a number of columns in excess of the number now set.

- Left Margin
  Used to determine the width in numbers of characters of the margin per page if the predefined parameters are not used.

The left margin is added to the number of columns per page, thus: page width = number of columns + left margin.

All the elements that can be included in a text page can be used in a print page. (See Page 4-3 -> “Page“)

Each print page can accept the insertion of a maximun of 64 variables in the case of VT170W while with other terminals the maximum is 128.

The dynamic fields inserted in a page are automatically limited to read-only fields, so it is not possible to vary data in this type of page.

Elements inserted in a print page assume new attributes called *Print Attributes* which are:

- Next page
  Makes it possible to go to the next page and then print the field with the attribute active.
- Bold
  Permits the field to be printed in bold.
- Underlined
  Permits the field to be printed in underline.
- Barred
  Permits the field to be printed in barred script.
- Italics
  Permits the field to be printed in italics.

- Code 1
  Configurable by the user.
- Code 2
  Configurable by the user.
- Code 3
  Configurable by the user.
- Code 4
  Configurable by the user.
- Code 5
  Configurable by the user.
- Code 6
  Configurable by the user.

These attributes are activated by selecting the object and clicking on ***Edit > Print Attributes***

Header and footer

The *Header and Footer* are a collection of fields determining the aspect of the top and bottom of the page as defined by the user.

Headers and footer are assigned to a print page.

The header is printed once only at the top of the page.

The footer is printed once only at the bottom of the page.

Headers and footers **cannot** contain graphical elements like images and drawings.

When the header and footer are declared for one printer they cannot be assigned to another printer: if necessary they can be duplicated. (If assigned to the printer connected to the ASP port they cannot also be assigned to the printer connected to the LPT port.)

Header and footer have various Attributes, namely:

- Name
  Name to define the header and footer so as to make the recognition of their function easier.
- Use predefined parameters
  Makes it possible to choose whether the print parameters must be specified in the header and footer or whether to accept the general parameters of the project.
- Printer
  Use this to select the printer from among those defined in the project, to which to assign the header and footer.

- Rows per Page
  Used to determine the number of printable rows per page if the predefined parameters are not used.
- Columns per Page
  Used to determine the number of printable columns per page if the predefined parameters are not used.
- Left Margin
  Used to determine the number of characters making up the width of the left margin per page if the predefined parameters are not used.

The following objects can be put into headers and footers:

- Alarms history sheet number
- Alarms sheet number
- General sheet number
- Hardcopy sheet number
- Report sheet number
- Date/Time field
- Multilanguage label

The above-listed objects, applied to the header and footer assume the same attributes as those applied to the print page. (See Page 4-3 -> "Page")

These objects are inserted using the Numerical Field and then selected from the list of the source variable.

Page number of alarm history file:

Is the number of pages printed in a historic alarm buffer; it is reset to zero with each new printing.

Page number of alarms:

Is the number of pages of alarms printed; it is reset to zero with each new printing.

General page number:

Is the total number of pages printed; it is reset to zero each time the VT is switched on or by using the appropriate function or when the device sends the relevant command using the command area.

Hardcopy page number:

Is the number of hardcopy pages.

Report page number:

> Is the number of pages contained in a report; it is reset to zero with each new printing.

Date/Time field

> See Page 4-31 -> “Hour/Date field“

Multilanguage Label

> See Page 4-4 -> “Multilanguage label“

Report

Report is a function that allows the creation of a “relationship” between the single print pages to the point of forming a single document.

This function allows one of the headers and footers defined in the project to be assigned to the pages.

There is only one header and footer for any given *Report.*

A project may consist of more than one *Report.*

The same page can be part of more than one report.

Reports can be sent to only one printer (ASP or LPT).

Reports can be classified into Reports with pages in Start/Stop Sequence or in Random Sequence.

Start/Stop Sequences

> For this type of sequence the first and last pages must be indicated. The number of the Start Page must be lower than the number of the Stop Page. It is not necessary for all the pages in the interval between Start and Stop to be present, but at least one must otherwise this type of sequence would not make sense.

Random Sequences:

> In this type of sequence, the pages can be put in any order. At least one page must be present for this type of sequence to make sense.

Reports 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.

Number:

Indicates the identifying number of the report.

Name:

Name defining the report. 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 report and its meaning, but it can also be an alphanumeric character sequence.

Start/Stop Sequence:

Allows this kind of sequence to be selected.

Random Sequence:

Allows this type of sequence to be selected.

Start Page (✤):

Active only if a start/stop sequence has been chosen: allows the first page in the sequence to be specified.

Stop Page (✤):

Active only if a start/stop sequence has been chosen: allows the last page in the sequence to be specified.

Page selected:

Active only if a random sequence has been chosen: allows the selection of pages in the sequence to be specified.

Use predefined settings:

Used to choose whether the print parameters must be specified in the report or whether the general parameters of the project are accepted.

Send Form-feed at the end of the page/footer:

When this box is selected the page can change even if it is not full.

Header:

With this one of the headers present in the project can be assigned to the report.

Footer:

With this one of the footers present in the project can be assigned to the report.

A print report can contain only one header and only one footer, valid for all pages.

Trend buffer

With the *Trend buffer* you can assign to the channel of a trend the variable to be kept under control and represented.

The *Trend buffer* variable can be read in three modes:

- Single sampling on polling
- Single sampling on command
- Block of sampling on command

Single sampling on polling:

The VT reads the variable (acquires a sample) at regular predetermined intervals (from 500ms to 24h).

Single sampling on command:

The VT acquires the sample after each internal command or by way of the data exchange. (See Page 4-62 -> “Exchange areas“).

Block of sampling on command:

The VT reads a block of readings acquired by and saved in the connected device. It is the job of the device to acquire the readings of the variable. A sample - irrespective of the format of the memory area: Bit, Byte, Word - occupies a minimum of 4Bytes (or 8Bytes if the memory area of the device is in DoubleWord format. This function is recommended when the variation in the value of the variable happens very quickly (>1sec.).

The values acquired are saved in the *Trend buffer*, a buffer inside the VT. The length of the buffer may vary, being definable according to the number of samples to be saved; each sample occupies 4Bytes, and the total quantity of memory available is related to the type of VT being used (see Hardware Manual). The greater the number of samples, the more detailed the graphic representation of the channel.

When the *Trend buffer* is full two procedures are possible:

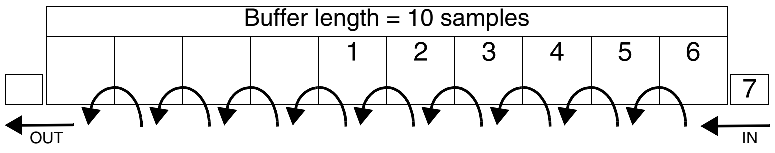
- FIFO (First In - First Out)
- Ignore exceeding samples

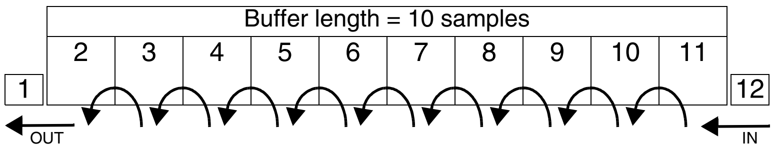
FIFO (First In - First Out):

Using this method, the buffer, when full, automatically eliminates the samples one at a time as and when new samples arrive, starting with the first saved.

Example.

Suppose there are 10 samples. The diagram below shows how the individual samples behave within the buffer.





Ignore exceeding samples:

With this procedure, once the buffer is full, any new samples are not saved; the buffer can be emptied either by using of an internal command or by using the data exchange (See Page 4-82 -> “Functions assignable to F and/or E keys and to touch buttons:“ e Page 4-62 -> “Exchange areas“)

The *Trend buffer* function 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.

🗁 General options.

Number:

This indicates the identifying number of the trend buffer

Name:

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

Comment:

A comment can be assigned that should, if possible, be a full explanation of the function and meaning, but could also be an alphanumeric character sequence.

Source:

It is possible to specify the device to which to assign the trend buffer.

variable(✤):

This parameter has two different meanings, depending on the sampling mode used. Using Automatic Single Sampling and Single Sampling on Command, it defines which of the device's variables must be checked, while using Block of Readings on Command, it defines the starting variable of the block within the device in which the samples have been saved.

Example.

Suppose the variable VAR1 is assigned to register R100, the block in which the sample is saved will start from register R100.

🗁 Mode.

Number of samples:

This parameter has two different meanings, depending on the sampling mode used. Using Automatic Single Sampling and Single Sampling on Command, it defines the number of samples (and thus the area of memory occupied) to be saved in the VT at the same time, while using Block of Readings on Command, it defines the length of the block within the device in which the samples have been saved.

Example.

Suppose we wish to define a number of samples equal to 60; using a device with 32 bit registers where each sample will occupy 8Bytes (1 complete register), 480Bytes (8x60) will be needed, that is, 60 registers. If the block starts from R100, it will run from R100 to R160.

Sampling mode:

The sampling mode can be specified (See Page 4-109 -> “Trend buffer“).

Polling rate:

This function allows the user to define the interval between one reading and the next.

Queue type:

The user can define how samples in excess of the length of the buffer should be treated. (See Page 4-110 -> “FIFO (First In - First Out)“).

Equations The function *Equations* is used to solve simple mathematical expressions.

The function *Equations* supports add [+], subtract [-], multiply [*] and divide [/] operations and can perform calculations using two levels of brackets.

The function *Equations* can use up to four variables for a single expression (1 for the resultat, 3 as operands) and 99 constants.

Example of an equation.

[VAR_1] = [VAR_2]*28+(([VAR_3]*2)+([VAR_4]*2))

The function *Equations* esegue calcoli con variabili a Bit, Byte, Word, Dword e Floating-point.

The function *Equations* performs calculations when called on by an automatic Operation (see Page 4-113).

The number of *Equations* that can be used depends of the terminal being used (see Hardware Manual).

The *Equations* have associates to them various parameters that must be compiled; some are obligatory (✤), others depend on the representational requirements of the user. The parameters are as follows:

Name:

The name serves to define the equation. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned that should, if possible, be a full explanation of the function and meaning, but could also be an alphanumeric character sequence.

Equation (✤):

Used to insert the mathematical operators and the operands of the mathematical expression by typing directly in the window or using the the relevant buttons.

Automatic operations

The function Automatic *Operations* makes it possible to have the terminal perform a certain Action when an Event occurs.

The possible actions are:

- Perform an Equation (see Page 4-112 -> “Equations“).
- Perform a Function (see Page 4-82 -> “Functions assignable to F and/or E keys and to touch buttons:“).

A triggering Event is defined as the occurance of a condition that is being monitored.

Events that trigger actions are:

- Expiry of an internal timer
- Value acquired by a variable
- Pressing the ▢ Enter after having set a variable

Expiry of an internal timer:

Allows an automatic operation to be performed when an assigned internal timer expires (see “Chapter 6 -> Timer“)

Example.

Suppose we define an internal timer with a value of 1 secondo, this means that when this time expires the action associated with the automatic operation is performed.

Value acquired by a variable:

Allows an automatic operation to be performed when the variable being monitored assumes a given value or enters a given band of values. The variable is checked at pre-established time intervals (100ms to 25sec).

Example.

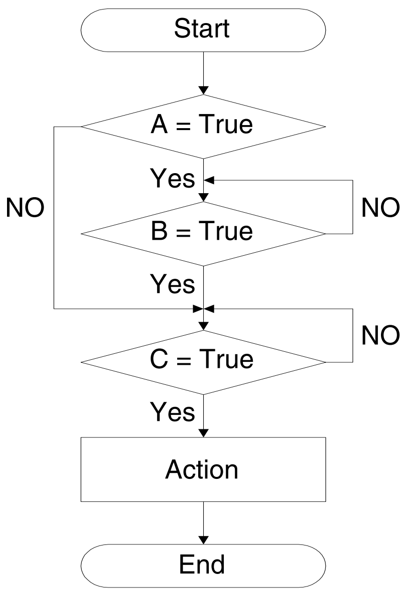
Suppose we check a variable every 100ms and have defined 1024 as the value to check. This means that when the allo scadere del tempo di controllo la variabile viene controllata, e quando raggiunge il valore di 1024, verrà eseguita l'action associata all'Automatic Operation.

Pressing the □ Enter after having set a variable:

Allows an automatic operation to be performed when, after having created a field on the terminal, you press □ Enter.

The function *Automatic operations*, if necessary, can have a further condition which, on the basis of the value of a variable, enables or disables the checking of an event. This mechanism is called Enabling Condition.

Flow chart of an Automatic Operation



A =Use enabling condition
B =Enabling Condition
C =Event

When condition A is satisfied, the system checks for condition B, otherwise condition B is ignored and the device passes directly to checking for condition C. If condition B is being checked then it is necessary to wait until the condition is satisfied. Once condition B is satisfied the device passes to checking condition C. If condition C is also satisfied then the action can take place.

The number of *Automatic Operations* that can be used depends on the terminal being used (see Hardware Manual).

The Automatic Operations linked to the same Event are performed in the order of their introduction.

This means that if they are not introduced correctly, there is the risk that the action linked to an event is late.

Example.

Event:	Timer_1

Automatic Operation_01:	Equation_01
Automatic Operation_02:	Equation_02
Automatic Operation_03:	Equation_03

Equation_01:	[VAR_1] = [VAR_4]+[VAR_5]
Equation_02:	[VAR_3] = [VAR_1]+**[VAR_2]**
Equation_03:	**[VAR_2]** = [VAR_6]+[VAR_7]

Note that Equation_02 produces the correct result [VAR_3] with a delayed event, given that variable [VAR_2] does not yet contain the updated values. Indeed, this is recalculated in the next passage, Equation_3.

To eliminate this inconvenience, just call up the Equations within the Automatic Operations in the following order:

Automatic Operation_01:	Equation_01
Automatic Operation_02:	**Equation_03**
Automatic Operation_03:	Equation_02

The display order of the Automatic Operations may not be the same as the order of introduction; display is in alphabetical order.

Order of Introduction	Display order
Operation_001	Operation_001
Operation_003	**Operation_002**
Operation_002	Operation_003

The *Automatic Operations* have associated with them various parameters that must be compiled; some are obligatory (✤), others depend on the representational need of the user. The parameters are those listed below:

Name:

The name serves to define the Automatic Operation. It is wise to assign a name that helps the programmer recognize it and identify its contents.

Comment:

A comment can be assigned that should, if possible, be a full explanation of the function and meaning of the Automatic Operation, but it could also be an alphanumeric character sequence.

🗁 Action.

Object:

Used to determine whether the action to be performed by the terminal is an Equation or a Function.

Equation Name (✤):

Active only if the object is an Equation: used to select which Equation to run.

Function (✤):

Active only if the object is a Function: you can assign one of the functions listed (Page 4-82 -> "Functions assignable to F and/or E keys and to touch buttons:") or direct commands (Page 4-89 -> "Direct Commands").

Function definition (✤):

Specifies the operational mode of the selected function.

Example.

If the function Macro is chosen it allows the user to establish which macro to activate.

Value:

Further sub-list if the specified function.

🗁 Event.

Type:

You can determine what type of event is to be monitored to have the assigned action performed.

Timer name (✤):

Active only if the type Timer is selected: used to select which timer is to be monitored for the action to be performed.

Source:

Active only if the type Variable is selected: used to determine the origin of the variable: whether it is in the device, is one of the data memory variables or is of another type. (Vedi Page 4-52 -> “Variables“).

Variable (✤):

Active only if the type Variable is selected: this is the variable, to which the event refers.

Polling interval:

Active only if the type Variable is selected: used to define the interval between one read and the next (from 100ms to 25sec).

Condition:

Active only if the type Variable is selected: used to define the checking criterion of the condition, whether “Equal to” or “In between”.

Value:

Active only if the type Variable is selected: used to define the value to assign to compare the condition.

🗁 Enabling condition.

Use enabling condition:

Used to activate this function.

Source:

See Event Page 4-116.

Variable (✤):

See Event Page 4-117.

Condition:

See Event Page 4-117.

Value:

See Event Page 4-116.

Chapter 5 Configurator menu in detail

This chapter consists of a total of 14 pages.

The menu options listed below do not function if the project under consideration is opened (using the project editor). Save and close.

File

New…

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

1) Enables you to create a Single VT project.

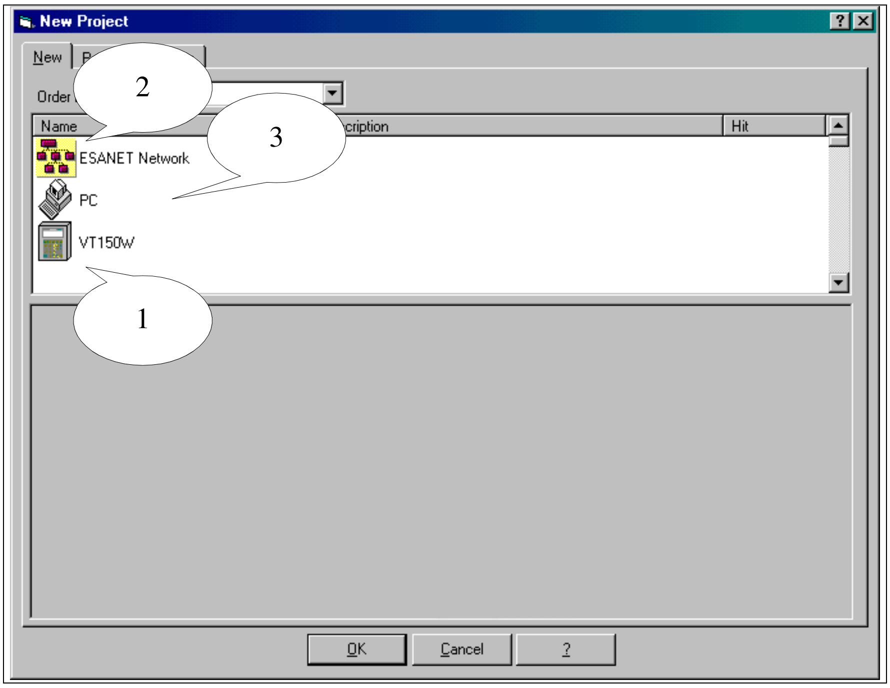
2) Enables you to create an Adapter project.

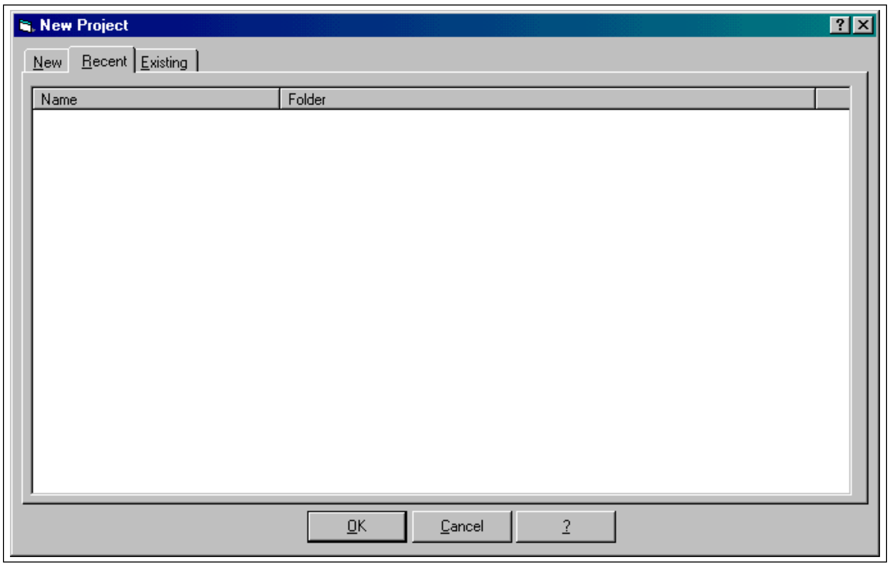
3) Enables you to create a project in the ESA-NET network.

Select one icon or the other.

Click on ☐ Ok.

If you wish to open a recent project, select the 🕮 Recent; the following mask will be displayed.





The list of available projects is displayed.

Select the one required.

Click on ☐ Ok.

1) Allows you to insert the project to be opened

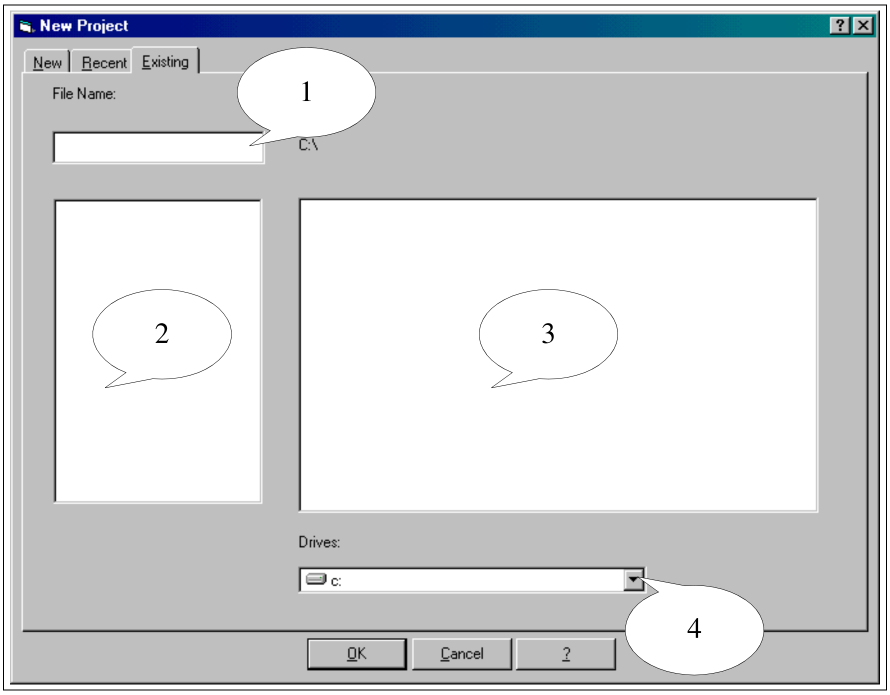
2) Allows you to display the files that are available.

3) Allows you to display the folders that are available.

4) Allows you to choose the disk drives

To open the project select the files required.

Click on ▢ Ok.



Project for Single VT:

After selecting the icon Single VT in the above image and confirming the choice with OK, recall the project configurator mask set out below.

⚠ **The parametrization masks common also to the ESA-NET network project will only be shown in the phase of the opening of the Single VT project.**

⚠ **To know which printers canbe used directly connected to the VT see** “Chapter 16 -> Printers directly connectable to the VT“**.**

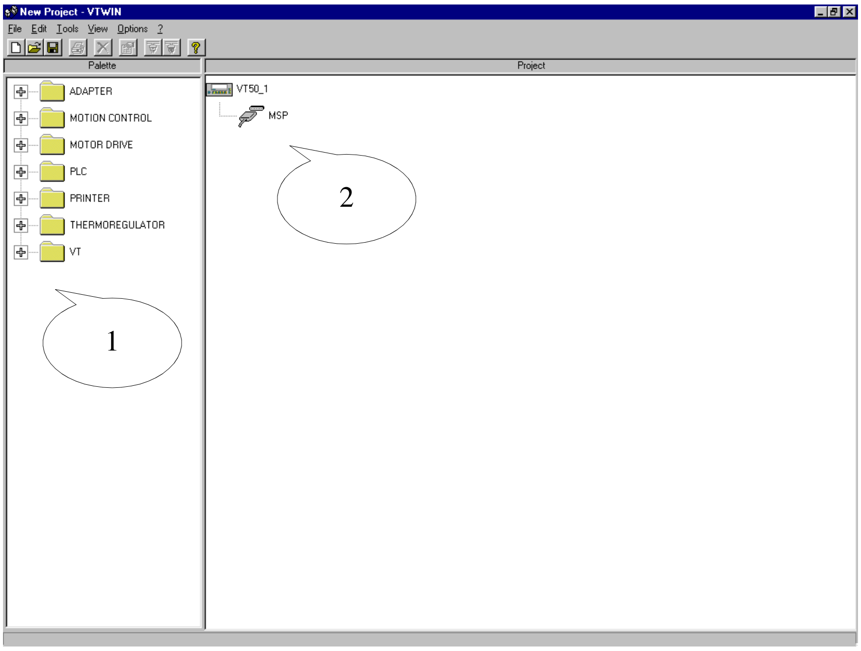
1) All the devices that can be included in a project are listed

2) The devices included in a project are listed.

There are two ways of changing the type of terminal.

a) Select the icon on the Project side, then click on Tools>Convert, and choose the VT you want in the list provided.
Confirm with Ok.

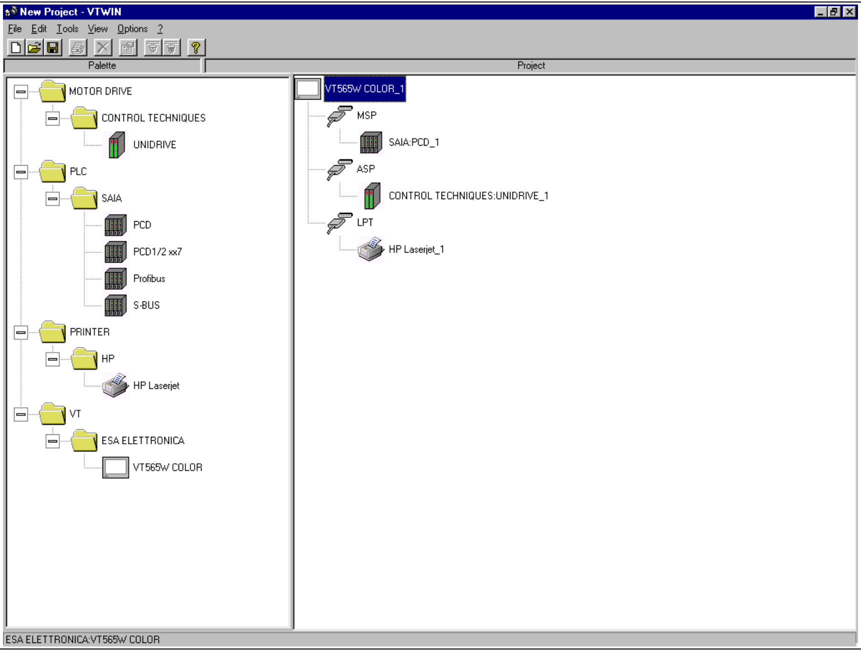
b) Select the icon on the Project side, click on ☒, then select a new VT on the Device side and drag it to the Project side.



One by one select all the devices to be connected to the VT.

Drag them from the Device side to the port where they are to be connected (eg. MSP, ASP,LPT) on the Project side.

To display and/or edit the parameters of the devices and/or communication ports of a project, select the icon required and then click on Edit>Property; the masks listed below appear.



⚠ **Should the device not hook up, it means that it is not support by that port.**

You are advised not to change the device once the project has been started. When one device is substituted by another the result could be the whole or partial loss of the variables (see "Appendix C")

1) Name assigned to the device connected is displayed.

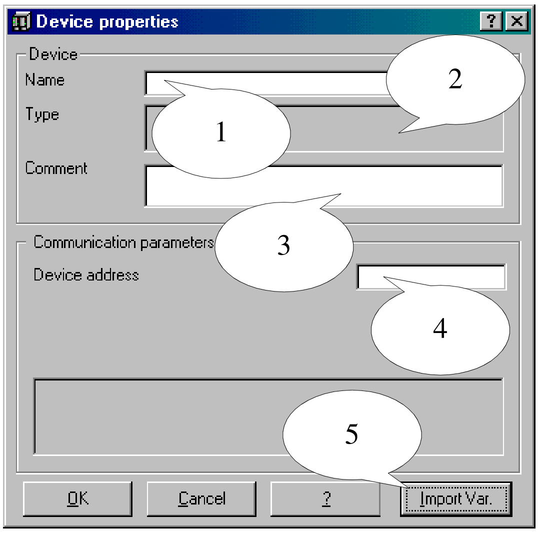
2) The type of device connected is displayed.

3) Comment assigned.

4) The parameters of the device are displayed.

5) Allows you to select the file(s) to import, that contain the variables.

Click on ▢ Import Var., to display the following mask..



Point 5 appears only if the device hooked up supports the Import Variables function.

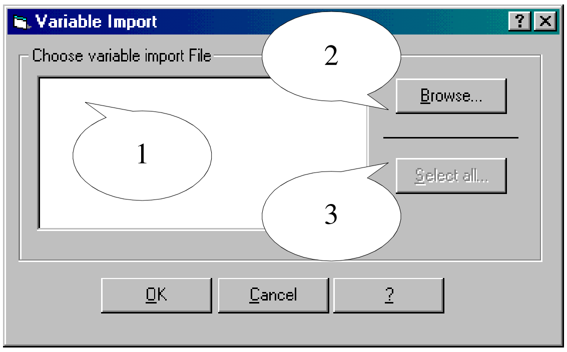
The Import Variables function does not have to be used, but it simplifies the procedure for inserting variables in a project (see "Chapter 6 -> Import from file"**).**

1) Shows which file(s) to import.

2) Allows you to select the folder holding the file(s).

3) Allows you to select all the files simultaneously.

Select the files to import. Click on ▢ OK to confirm and return to previous mask.



The "Device parameters" may vary in line with the device connected.

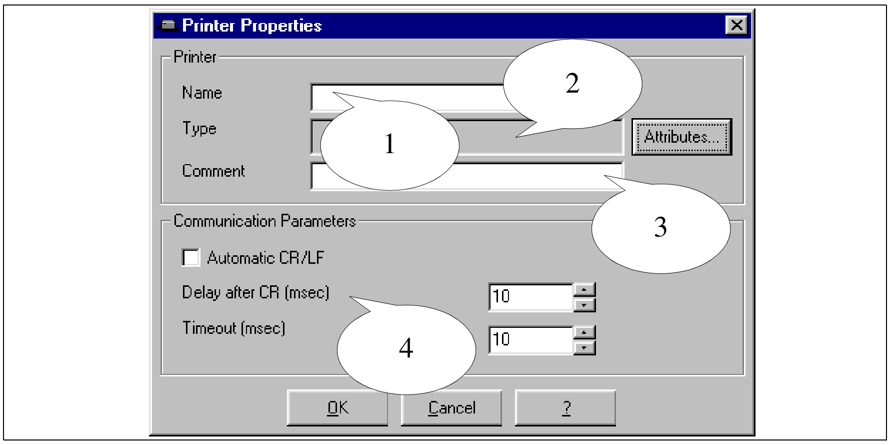
1) The name assigned to the printer connected appears.

2) The type of printer connected appears.

3) The comment is assigned.

4) Print parameters are displayed.

Click on ☐ Attributes; the following mask appears.



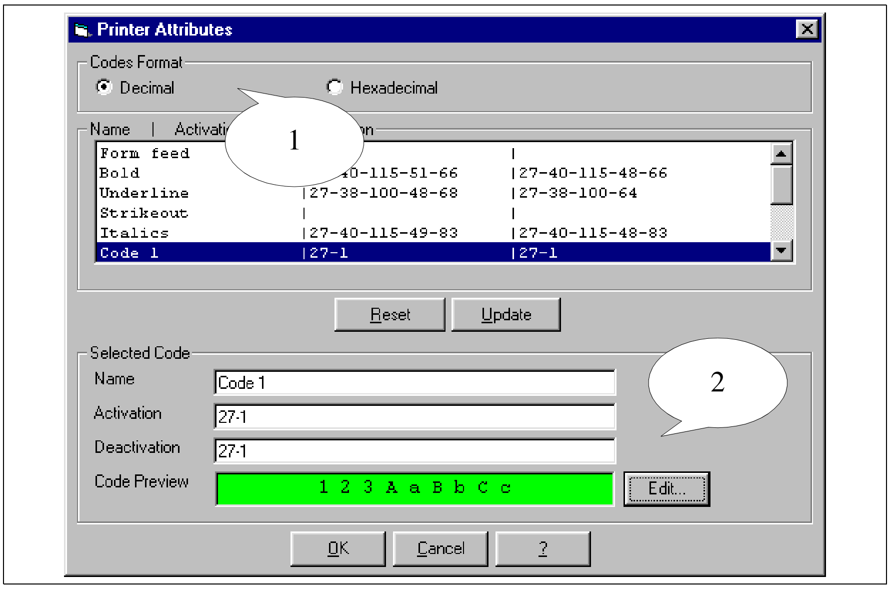
Printer
Name
Type
Comment

Communication Parameters
☐ Automatic CR/LF
Delay after CR (msec) 10
Timeout (msec) 10

1) Selection of printer code formats.

3) For customizing print attributes Code 1 to Code 6.

Select a 🕮 and click on ☐ Edit.



Codes Format
[✓ Decimal] Hexadecimal

```
Name  |  Activati
Form feed          |                    |
Bold               |     0-115-51-66    |27-40-115-48-66
Underline          |27-38-100-48-68     |27-38-100-64
Strikeout          |                    |
Italics            |27-40-115-49-83     |27-40-115-48-83
Code 1             |27-1                |27-1
```

Selected Code
Name: Code 1
Activation: 27-1
Deactivation: 27-1
Code Preview: 1 2 3 A a B b C c

(2) Name:

Name of the code to be sent to the printer; if possible use a name that identifies its function.

(2) Activation:

Command for activating the attribute.

(2) Deactivation:

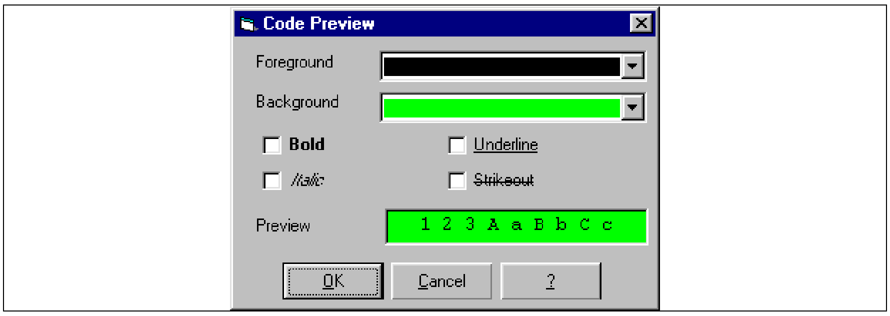
Command for deactivating the attribute.

(2) Preview:

This allows you to personalize the form of the preview of the action commanded. This becomes useful when the personalized code has to be identified within a given context.

Define the preview.

Click on Ok.

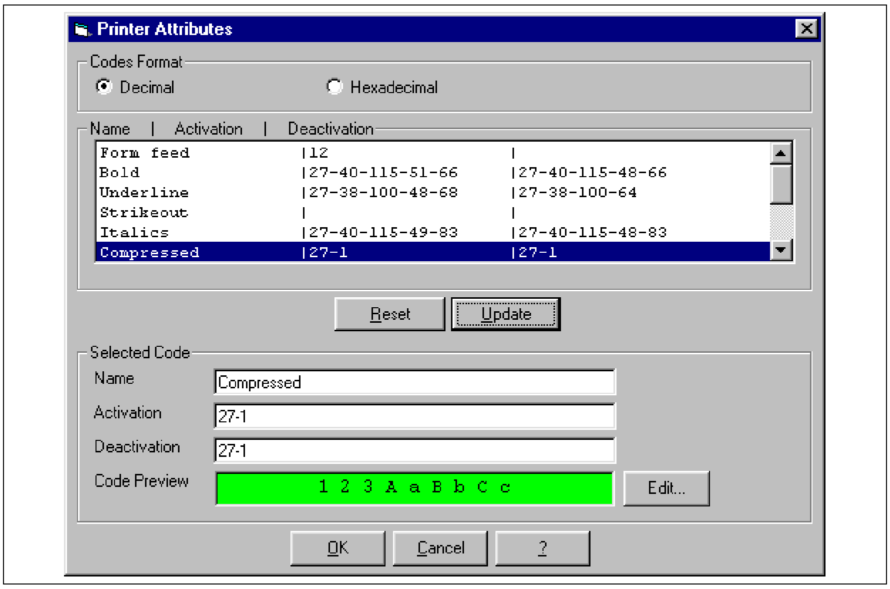


Code Preview

Foreground	⬛
Background	🟩
☐ Bold	☐ Underline
☐ Italic	☐ Strikeout
Preview	1 2 3 A a B b C c

Click on Update to confirm changes.

Click on Ok.



Printer Attributes

Codes Format: [✓ Decimal] Hexadecimal

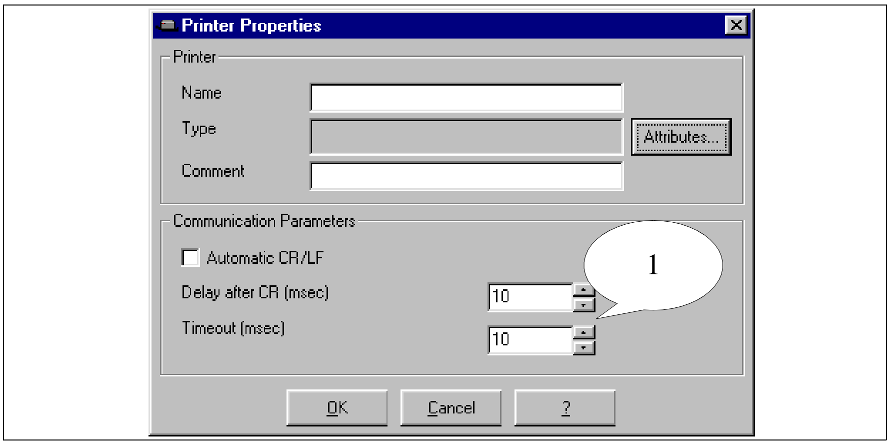
```
Name  |  Activation  |  Deactivation
Form feed          |12                |
Bold               |27-40-115-51-66   |27-40-115-48-66
Underline          |27-38-100-48-68   |27-38-100-64
Strikeout          |                  |
Italics            |27-40-115-49-83   |27-40-115-48-83
Compressed         |27-1              |27-1
```

Selected Code

Name	Compressed
Activation	27-1
Deactivation	27-1
Code Preview	1 2 3 A a B b C c

1) Activation of control parameters for the printer.

Click on Ok.



Printer Properties

Printer

Name	
Type	
Comment	

Communication Parameters

☐ Automatic CR/LF	
Delay after CR (msec)	10
Timeout (msec)	10

(1) Automatic CR/LF:

If the printer is equipped with this function, it can handle the command “new line” after a given number of columns, otherwise the VT assumes this role.

(1) Delay after CR (msec):

Time to wait until the carriage re-assumes a rest position before the VT is used to send a command or character.

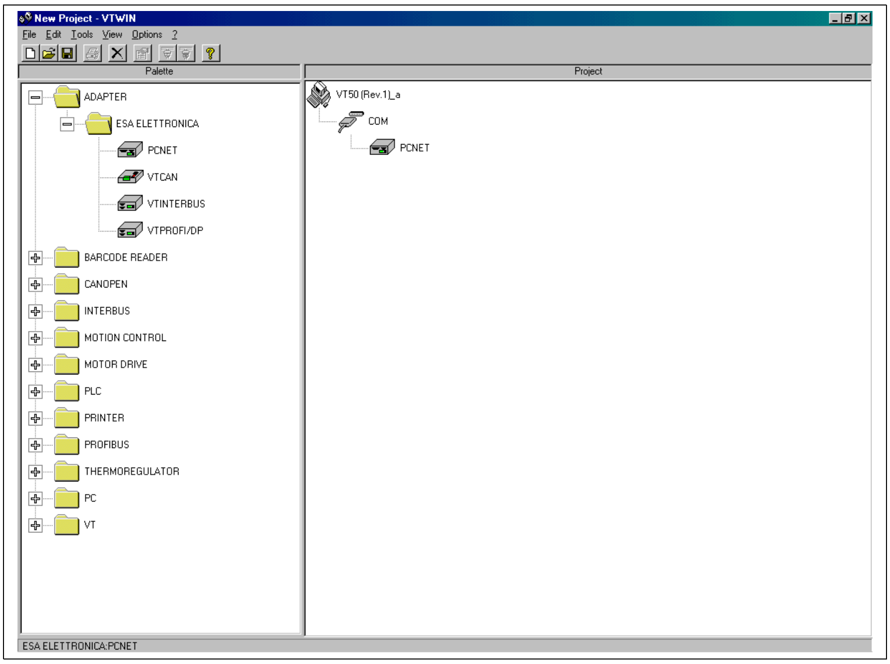
(1) Timeout (msec):

Delay that elapses between one sent given by and the next.

Adapter project:

The procedure for creating this is the same as for “Single VT project”; the only difference is in the type of device to be used: not a VT but a PC plus ADT.

As already outlined, access the following mask, convert from VT to PC and connect an adapter or select the Adapter icon and confirm the choice with OK (see Page 5-3).



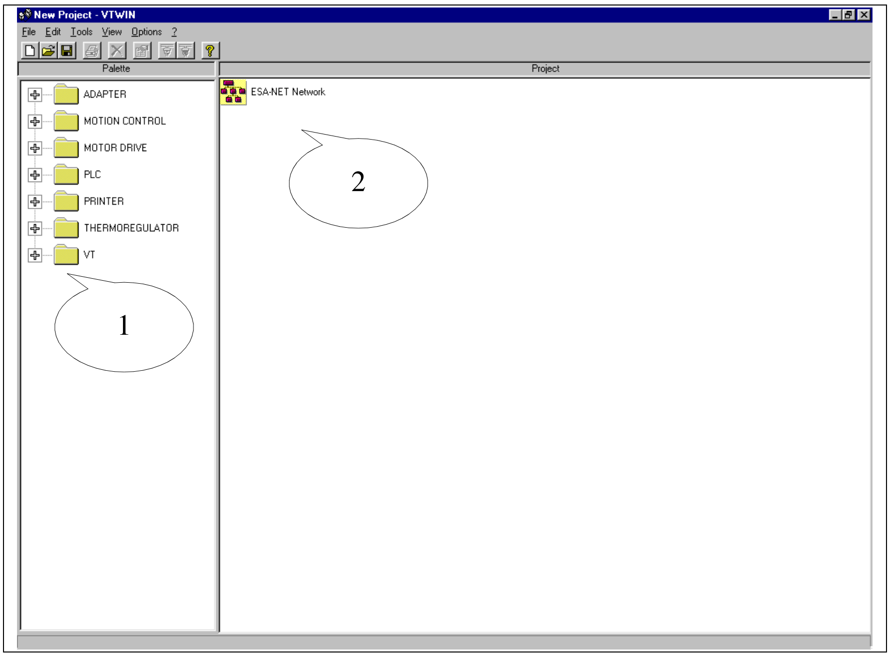
Now select the device to be connected to the PC and drag it onto the ADT (see Page 5-3).

ESA-NET network project:

After selecting the ESA-NET network icon in figure on Page 5-3, and confirmed the choice with OK, the project configurator mask shown below is recalled.

For the parametrization masks common also to the Single VT project see Page 5-3.

To know which printers can be used directly connected to the VT see "Chapter 16 -> Printers directly connectable to the VT".



1) All the devices that could be included in a project are listed

2) All the devices included in the project are listed.

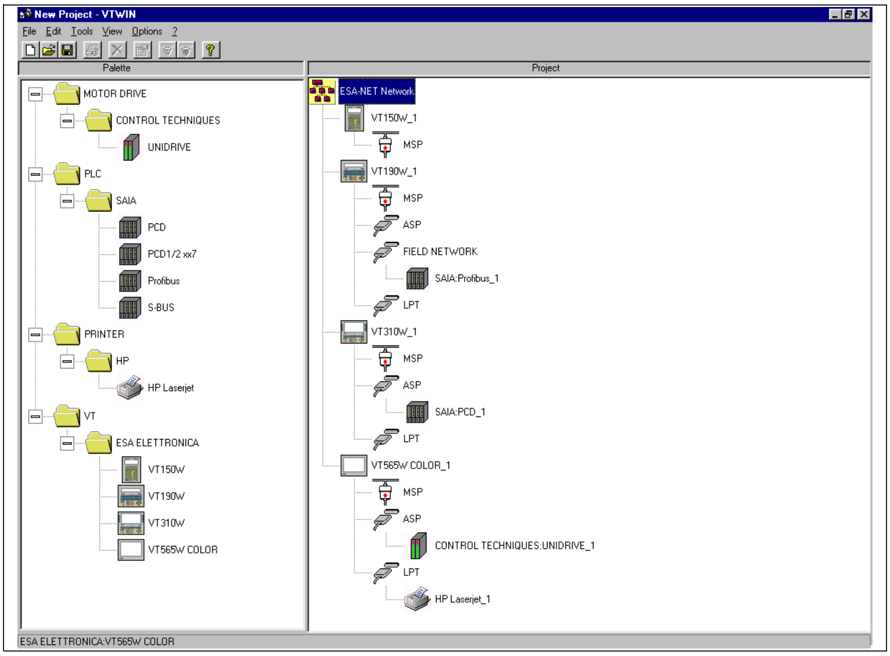
For information on the maximum number of participants in the network see Hardware Manual.

One by one select all the VTs and PCs to be included in the network.

One by one select all the devices to be connected to the VT.

Drag them from the Devices side to the ports where they are to be connected (e.g.. MSP, ASP,LPT) on the Project side.

To display and/or edit the parameters of the devices and/or communication ports of a project, see Single VT project Page 5-3.



Note that VTWIN automatically connects to the serial port marked to be preferred for connection to the ESA-NET network. (For the selection criteria see Hardware Manual, “Network connections”). If necessary, you can change the type of serial port to be used. The illustration below shows the connection of the VT190W within an ESA-NET network, using the MSP serial port.

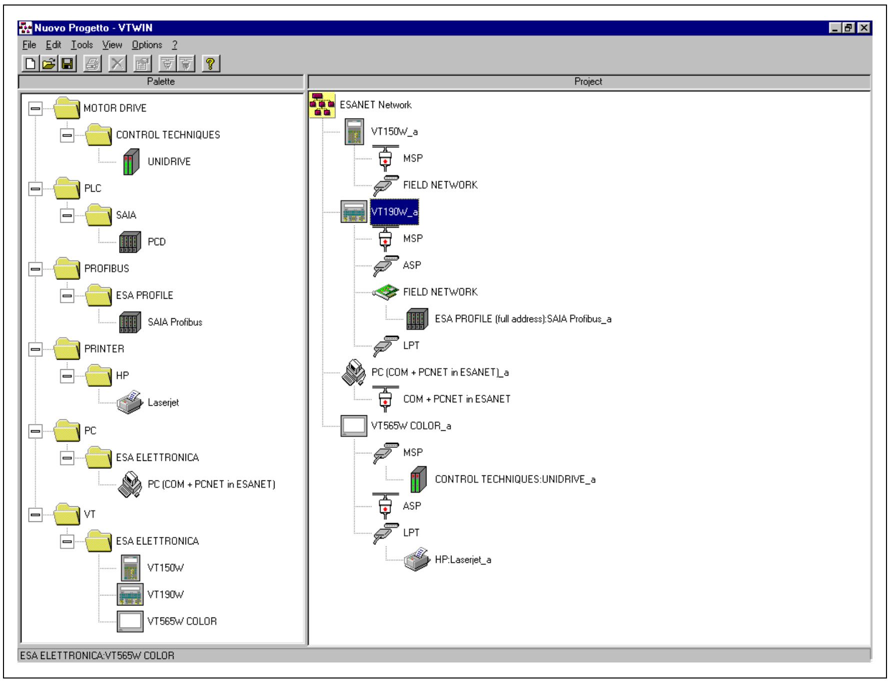
Select the FIELD NETWORK port of the VT; click on ; the network is now disconnected.

Select the device from the Devices side and drag it onto the FIELD NETWORK port.

Select the MSP port and click on .

The connection with the ESA-NET network has been re-established via the MSP serial port.

To display and/or edit the parameters of the devices and/or communication ports of a project, see Single VT project Page 5-3.

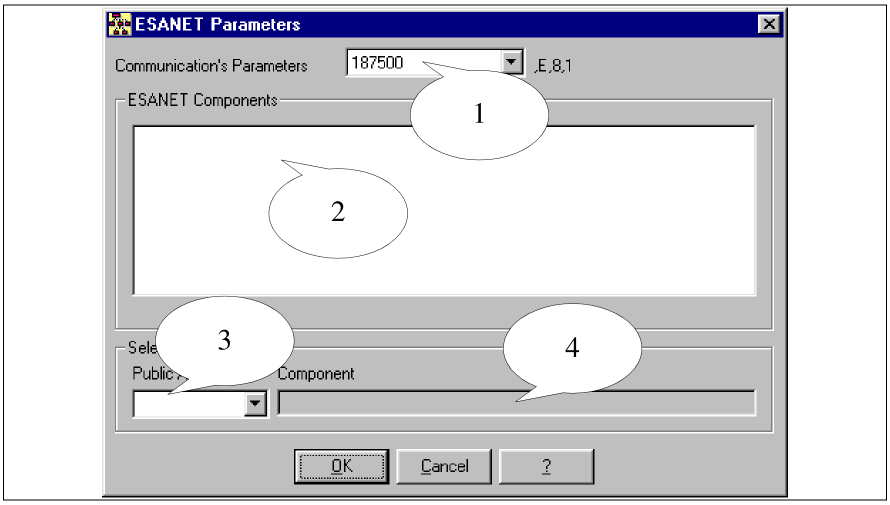


1) Display the communication speed.

2) Display the type of device connected.

3) Network address assigned (different for each participant).

4) Displays the VT assigned to a given network address.



ESANET Parameters

Communication's Parameters: 187500 ,E,8,1

Open…

Used to call up an existing project.

Versions…

Used to creare several versions of the same project automatically or on command.

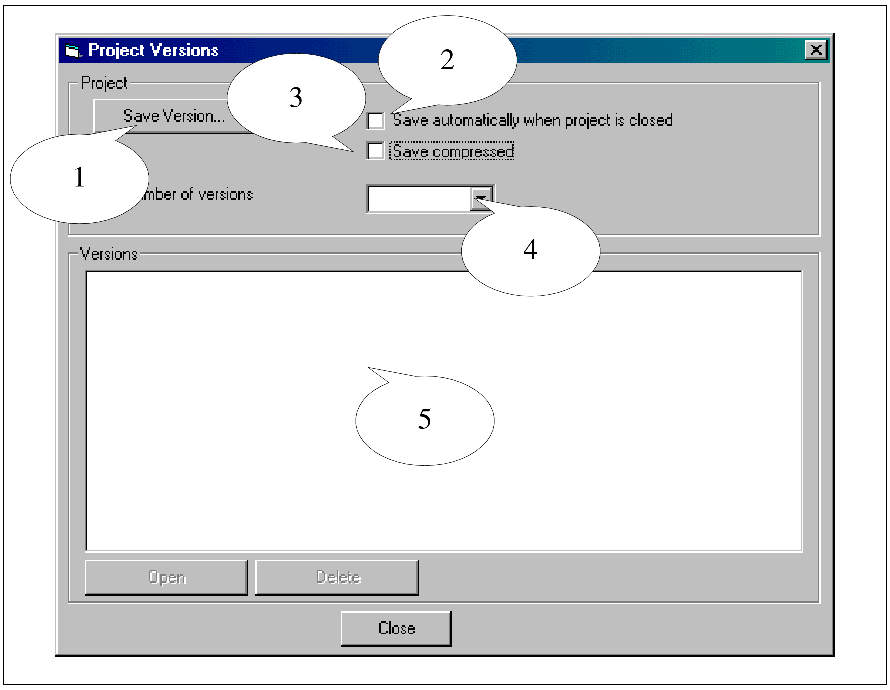
The versions of the project are put in a sub-folder of the folder of the project original called by the same project name but withouot the extension.

A prefix “REVxx_” is added to the beginning of the file, where xx is a progressive number indicating the versions; 01 is always used to identify the most recent version.

Each version of the file creates a text file in which a description can be inserted.

The files of the versions can be generated in compressed form. A “z” is added to the extension (.VTSz - .VTNz) of this type of file.

The selection of the Versions... causes the following mask to open:



1) Used to create a version of the project.

2) Used to set the automatic creation of a version each time the project is closed.

3) Used to determinare whether the project must be compressed.

4) Used to set the maximum number of versions allowable.

5) Used to display and manage the versions of the project.

Save

Used to save the project currently being displayed onto a disk.

Save as…

Used to save the project currently being worked on with a different name.

Exit

Used to abandon VTWIN.

Edit

Properties…

Used to display and/or edit the properties (name and comment) of a VT within the project.

Edit

Used to access the project editor where it is possible to create and edit a project.

Rename

Used to re-name the project(s).

Delete

Used to delete a given project from the archive.

⚠ **The project is IRREVOCABLY deleted.**

Tools

Convert…

Used to convert one type of VT, contained in the project, into another (E.g. from VT170W to VT190W).

⚠ **Changing the type of VT or the type of device connected may lead to the loss of data or graphics.**

Import...

Used to import a Single VT project into an ESA-NET network project and/or an Adapter project.

Export...

Used to export a Single VT project from an ESA-NET network project and/or an Adapter project.

⚠ **An Adapter project can only be used to be imported into another ESA-NET network project. It cannot be used as a single Adapter project.**

Create disk for updating operator terminal

See “Chapter 10 -> Creating and printing documentation“

BOOT update

See “Chapter 14 -> BOOT update“

Print…

See “Chapter 10 -> Creating and printing documentation“.

View

List labels

Used to activate or deactivate the display of the labels bar.

Devices

Used to activate or deactivate the display of the devices column.

Status bar

Used to activate or deactivate the display of the status bar.

Options

VTWIN Language

Used to display the mask containing the various languages in which VTWIN can be displayed.

⚠ **When display in Chinese is selected, Chinese must also be set in International Options of PC (see also** “Chapter 15 -> Using a TTF Extended Font“**).**

?

Contents

Used to call up the index of all the subjects contained in the Help on Line.

Search Help For...

Used to call up a mask for finding a particular subject.

About...

Used to call up a mask where ***System information*** and ***Installation control*** can be found: the former gives information on the machine where VTWIN has been installed, the second has information on the installation of VTWIN.

Chapter 6

Editor menu in detail

This chapter consists of a total of 36 pages.

Project

Update Public data

Used to save public data on disk making them available to other participants in the ESA-NET network.

Close

Used to quit the project currently being displayed, saving is left to the operator’s choice.

Tools

Compile project

See “Chapter 9 -> Compiling and transferring a project“

Download project

See “Chapter 9 -> Compiling and transferring a project“

Download with modem

See “Chapter 9 -> Compiling and transferring a project“

Font editor

See “Chapter 12 -> Defining the fonts“

Backup/Restore

Used to recover (Backup) from a VT the recipes and/or the project and to transfer (Restore) them/it to one or more panels. The files are saved on disk; the recovered files cannot be edited.

Backup/Restore with Modem

Function that is analogous to a Backup/Restore with the difference that in this case there is a check that the Modem is present.

Export to file

Allows a series of items of information (for details see individual menus listed below) to be exported using a formatted text file (TXT) or using a file formatted as Comma Separated Value (CSV). The former is editable with a text editor, the latter can be also edited with programs for handling electronic pages (E.g. Excel). For how to format fields see “Appendix B“).

⚠ **The separator in file CSV depends on the language configured in**

International Settings. We therefore advise keeping the same settings when exporting and subsequently importing (see also “Chapter 15 -> Multilanguage support“**).**

This menu contains the following sub-menus.

Export translations

Allows all texts requiring a translation to be exported from a VT. This way translations required can be inserted without the help of VTWIN (see also “Chapter 15 -> Multilanguage support“).

To generate the file all that is needed is to have set a project language, but to assign translations you must set al least two languages (See Page 6-13).

The reference language in the file must never be changed, otherwise the file cannot be imported.

The exported file must then be reimported into the same project.

In the editing phase be careful that the format of the file is not ruined, otherwise certain texts may be lost or it may become impossible to reimport the file.

Variables

Allows all variables contained in the project to be exported. This way information can be exchanged between different programs and/or variables modified/inserted without the help of VTWIN.

The elimination of one or more variables in a file does not entail its/their elimination once the file has been re-imported into VTWIN.

Import from file

This menu contains the following sub-menus.

Import translations

This function allows all the texts that have been previously exported using a formatted file to be imported to a VT.

The file must be generated using the Export translations function and then modified.

Files exported with other projects cannot be imported.

Variables

Allows variables to be inserted in a project importing them from one or more files external to VTWIN. These files may be those previously exported and then edited (TXT or CSV) or files generated by the program for handling the project of the device connected to the VT (AWL and/or ASC) and must be acquired using VTWIN (see “Chapter 5 -> Project for Single VT:“).

AWL and ASC files are imported whole without being able to differentiate between the variables contained in them (for how to import individual variables see “Chapter 4 -> Importing variables:“**).**

When importing it is important to compile all the parameters required to make them compatible with the file being imported (E.g. Unicode or ASCII, etc.).

Variables whose parameters are wrong after the CSV and/or TXT files have been imported are marked with an asterisk (*) before the name of the variable and in these erroneous parameters are listed in the Information window.

If variables have the same name they are automatically overwritten.

Import/Export from user DB

This menu contains a secondary menu which is set out below.

Header/Footer:

Used to import or export a header and footer into/from a project, so as to be able to use them in other projects.

Setting external editor

Allows the user to define which editor external to VTWIN will be used to display/edit the variables when the *External Editor* is used. VTWIN automatically sets the editor indicated in the configuration register of the PC.

The editor can be modified/defined at any point if the following fields are compiled.

Application:

Allows you to define which application is to be used.

Format:

Allows you to define which format is to be used to open the list of variables.

To activate the external editor, go to the Components window of the project, select the option Variables and click on the External Editor button.

To make the changes in VTWIN effective, the file must be saved using the name proposed by VTWIN.

Object

The content of this menu is explained on “Chapter 7 -> Meaning of the Configurator menu icons“, where the corresponding VTWIN icons are also shown.

Fields

The content of this menu is explained on “Chapter 7 -> Meaning of the Configurator menu icons“, where the corresponding VTWIN 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 7 -> Meaning of the Configurator menu icons“, where the corresponding VTWIN icons are also shown.

Create macro field

This allows several fields to be grouped together as if they were one. This function applies to numeric fields. To be able to create a macro field the individual component fields must be settable; the read-only fields are automatically excluded from the selection. The purpose of the macrofield is to write the fields in the device simultaneously.

An example.

Suppose we want to create a macro field composed of 3 individual fields 1-2-3; the fields 1 and 3 are writable while field 2 id read-only. If all three fields are selected then field 2 although selected will not form part of the macro field.

Selection
Field 1
Field 2
Field 3

->

Macro field
Field 1
Field 3

Delete owner macro field

When an individual field is selected, the macro field can be broken up into its component individual fields.

Select owner macro field

When an individual field is selected, the other individual fields that form part of the same macro field are highlighted.

Page

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.

Internal 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. (See also “Chapter 4 -> Touch button“).

External keys definition

As for global internal □□. The external □□ are called E □□.

Copy to clipboard

Used to save the graphic or text page being displayed in VTWIN onto the Windows clipboard so that it can later be pasted into another image editor program (e.g. Paint).

Axes origin

This menu contains a sub-menu set out below.

> ***Set:***
>
> Used to define and display (by means of a intersecting co-ordinates) the X, Y coordinates of the zero point relative to the page's absolute zero.
>
> ***Reset:***
>
> Used to reset the relative zero point, setting the X, Y coordinates to coincide with the absolute zero of the page.
>
> ***Visible:***
>
> Used to enable or disable the function *Axes origin* without losing the co-ordinates of the relative zero.

Configure

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 VTWIN version
- Version of firmware necessary

The listed below can be set by the user:

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

Password

The VT offers 10 *Password* levels from 0 to 9, used to restrict access to the system to a limited number of operators.

Each *Password* is composed of a maximum of 6 numerical characters.
It is not necessary to compile the *Passwords* in order of level; they can be programmed at the user's discretion (for example, level 0, 4, 9 while the

others are not programmed).

Identical *Passwords* cannot be entered; level 0 gives the highest level of security (supervisor).

The *Password* has various parameters that must be compiled; some of these are obligatory (✤), others depend on what the user needs to have represented. The parameters are listed below.

Logout timeout:

> This indicates the time (expressed in minutes) after which, if no ▢ has been pressed, the VT abandons the password level that has been activated. The time is the same for all levels. This parameter is enabled when the time is set to any value other than zero.

Logout page/sequence:

> Identifies which page (in touch screen models) or which sequence (in non-touch-screen models) to display after carrying out the logout operation. This parameter can be enabled by entering a value in the appropriate box.

Password levels:

> Used to select the level to which you wish to assign a password.

Name (✤):

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

Password (✤):

> This is the numerical code that identifies the password.
> Giving the appropriate command you can change the passwords directly using the VT. (See “Chapter 4 -> Internal Command: PASSWORD - Modify password“.)

Timer

Used to define timers internal to the terminal that are NOT dependent on the device connected that can be used as Event generators in the Automatic Operations (see “Chapter 4 -> Automatic operations“).

The value of the count is between 100ms and 25sec (not modifiable by the

VT) and not directly displayable on the page.

The value of the count by the internal timers is NOT retentive, thus, if the terminal is switched off, the values are lost even if there is a battery.

The timers, if defined, are always active and they restart automatically when the value set has expired.

The number of timers depends on the terminal being used (see Hardware Manual).

The *Timers* have associated to them various parameters that must be compiled; some are obligatory (✤), others depend on the representational needs of the user. The parameters are listed below.

Timer:

Identifies which timer must be enabled.

Name:

Name used to definire the timer. It is advisable to assign a name that makes it easier for the programmer to recognize and understand its significance.

Value:

Used to assign the timer counting value with fixed 100ms-intervals.

Internal keys global definition

Used to define the association between F ▢ and function, that will be valid for the entire project irrespective of the page being displayed. This correspondence remains valid so long as the ▢▢ 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 ▢ shown in the list. (See also "Chapter 4 -> Touch button").

External keys global definition

As for Internal ▢▢ global definition. The External ▢▢ are referred to as E ▢▢.

System messages

In certain situations the various messages displayed by the VT can be changed (For example: if the password entered is incorrect, the message

“Password not correct”). System messages are multi-language texts and can be translated.

Field keydoard

The touch screen VTs have a series of keyboards there displayed in the phase in which the dynamic fiilds are set. Using this menu option you can choose the type of keyboard to use from those available in VTWIN.

The type of keyboard selected will apply to all a project’s settable fields.

The *Field Keyboards* option has various parameters assigned to it which must be compiled; some are mandatory (✤), others depend on the representation needs of the user. The parameters are listed below.

Autorepeat delay:

Identifies the time (expressed in milliseconds) beyond which the VT will repeat the same character, if a ▢ is held down.

Example.

If this parameter is set at 1000ms (t1=1Sec.), pressing ▢ “A” and holding it down, another “A” will be written once 1 Sec. has elapsed.

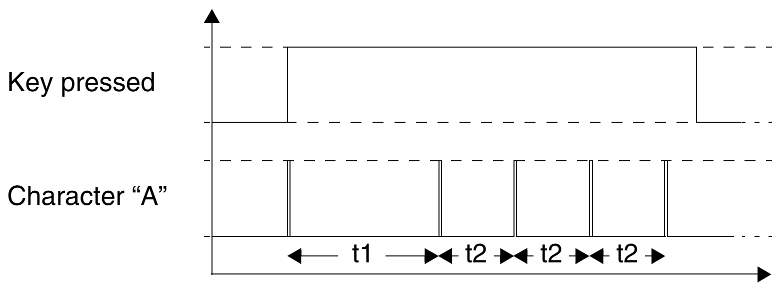
Autorepeat interval:

Identifies the time (expressed in milliseconds) beyond which, once the “Autorepeat” time has elapsed, the VT will repeat the same character, if a ▢ is held down.

Example.

If this parameter is set at 500ms (t2=0.5Sec.), pressing ▢ “A” and holding it down, another “A” will be written once 0.5 Sec. has elapsed.

Thus, going back to the example of the “Autorepeat delay” parameter the following will obtain:



Keyboard type:

Lets you select the type of keyboard to use.

Selected keyboard:

Enables you to display the keyboard chosen according to the field to be input.

Image memory area

Used to display the value of the total graphic memory of the panel being used and the value of the part currently occupied by the images; in addition it can be used to determine *a posteriori* whether the images contained in VTWIN are to be sent to the panel in compressed format or not. For the criterion governing the choice of compression see “Chapter 4 -> Images“.

Configuring languages

This allows the user to create a language set called *Configurations*.

These *Configurations* make it possible to extend the number of languages for the project to 32.

The number of languages handled simultaneously in the terminal remains as defined in the characteristics of the VT being used (see Hardware Manual).

If the languages used in the project do not exceed those handled by the terminal there is no need to create the *Configurations*.

As soon as one *Configuration* is created, the unused languages are marked with an asterisk (*) and cannot be handled.

The *Configuration* that is transferred to the terminal must be indicated in the compilation phase of the project (see Page 6-2 -> “Compile project“).

The languages transferred to the terminal are those included in the configuration chosen, so if you also want your native language (see Page 6-13 -> “Project language“) transferred, it must be included as the first in the list.

The terminal is triggered with the language that heads the Configuration transferred.

An example:

Let us suppose we are using a terminal supporting 4 project languages and want to create in one case a project with 4 languages, and in another one with 10 languages.

First project - 4 languages without using Configurations: only the languages are defined

Configurations	Languages	
None	1	Language 1 (Native Language)
	2	Language 2
	3	Language 3
	4	Language 4

First project - 4 languages with the help of configurations: two configurations are defined, the first including the native language and the second without. The terminal will be triggered with Language 1 or else Language 2 depending on which is transferred.

Configurations	Languages	
Configuration 1	1	Language 1 (Native Language)
	2	Language 2
	3	Language 3
	4	Language 4
Configuration 2	1	Language 2
	2	Language 3
	3	Language 4
	-	-

Second project - 10 languages: A number of configurations must be defined sufficient to contain the 10 languages. In all the configurations the terminal needs to be triggered with Language 1.

Configurations	Languages	
Configuration 1	1	Language 1 (Native Language)
	2	Language 2
	3	Language 3
	4	Language 4
Configuration 2	1	Language 1 (Native Language)
	2	Language 5
	3	Language 6
	4	Language 7
Configuration 3	1	Language 1 (Native Language)
	2	Language 8
	3	Language 9
	4	Language 10

Project language

As already mentioned, VTWIN 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 (1: Language) is taken as the mother tongue (M.-t.), that is the language in which things are normally displayed and which determines the maximum number of characters for the translations of the other languages (see “Chapter 4 -> Multilanguage text“). 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. Two families of characters are available: the first uses fonts supplied by ESA (we will continue to call these native fonts) with VTWIN; the second, to create an font image in VTWIN, uses as a point of departure the fonts contained in Windows (we shall call these Windows-based fonts).

Windows-based fonts can be handled in Standard (256-character) format or in Extended (65536 character) format, e.g.. Unicode or any other format of over 256 characters).

The extended font mode is supported exclusively by Windows 2000 / XP / NT 4.00.

Use of fonts in extended mode is accompanied by a series of warnings (see “Chapter 15 -> Multilanguage support“**).**

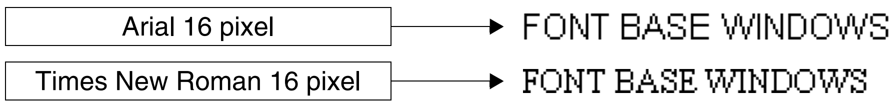
To activate the handling of Extended fonts, the International settings of the Operative System being used must be configured (see Operative System Manual and “Chapter 15 -> Multilanguage support“**).**

Native fonts alone have the advantage of being customizable by the user (see “Chapter 12 -> Defining the fonts“) but their resolution is low, whereas Windows-based fonts have the advantage of both high visual quality and an extensive range of fonts.

Windows-based fonts should not be of excessively small size otherwise there will be a loss of quality.

Families of fonts cannot be mixed in a project: either only native fonts are used or only Windows-based fonts.

The number of characters per line that can be insered using Windows-based fonts does not depend on the type of terminal (see Hardware Manual) but rather on the dimensions of the font used. Two different types of font may have different lengths even with the same dimensions and number of characters.



The number of characters depends on the surface measurement in pixels of a multilanguage label using native fonts with a x 1 dimension (see Hardware Manual).

Example.

In a VT585W a multilanguage label can contain up to 80 characters (using native fonts). Given that a character occupies 8x16 pixels, the area occupied is 8 pixels by 80 characters (640 pixels) wide and 16 pixels by 1 row (16 pixels) high. Using the same criterion, a similar calculation can be made for all terminals.

With Windows-based fonts functions X1, X2 and X4 have no effect.

In accordance with the type of VT used, VTWIN creates a list of the fonts it has available.

The number of fonts depends on the type of panel used and on the family of fonts chosen. Only 1 font can be chosen with text panels using native fonts (Windows-based ones are not supported); for graphic panels see the following table.

Table 6.1: Fonts correlated to the hardware revisions of the VT with which they can be used.

TERMINALS		FONTS	
Modello	**Revision**	**Native**	**Windows-based**[1]
VT300W	1	4	16
VT310W	1	4	16
VT320W	1	4	16
VT330W	1	4	16
	2	4	16
	3	4	16
	≥4	4	64
VT155W	1	4	4
VT185W	1	4	4
VT505H	1	4	8
VT505W	1	4	8
VT525H	1	4	4
VT515W	1	4	8
VT525W	1	4	4
VT555W	1	4	16
VT560W	1	4	64
VT565W	1	4	16
	2	4	16
	3	4	16
	≥4	4	64
VT575W	1	4	64
VT580W	1	4	64
VT585W	1	4	16
	2	4	16
	3	4	16
	≥4	4	64
VT585WB	1	4	64
VT595W	1	4	64
Notes: 1 - Depends on Font Windows memory available			

The fonts chosen become the active project fonts and can be used for editing texts in various languages. If no font is selected, the font used is that imposed by the system itself.

Example:

Let us suppose that our intention is to create a project in three languages (Language 1 to Language 3) and that the VT 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 6.2: Font.

Fonts available	Display
System font	ABCD abcd 1234
Font 1	**ABCD abcd 1234**
Font 2	ABCD abcd 1234
Font 3	*ABCD abcd 1234*
Font 4	*ABCD abcd 1234*
Font 5	***ABCD abcd 1234***
Font 6	ΑΒΧΔ αβχδ 1234
Font 7	***ABCD abcd 1234***
Font 8	ABCD abcd 1234
Font 9	ΑΒΧΔ αβχδ 1234
Font 10	***ABCD abcd 1234***

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:

		Project font		Language font
Lang. 1 (M.-t.)	->	Font 1	->	Font 1
		Font 6		Font 6
		Font 7		Font 7
		Font 10		Font 10

		Project font		Language font
Language 2	->	Font 1	->	Font 7
		Font 6		Font 10
		Font 7		Font 1
		Font 10		Font 6

Language 3	->	Project font	->	Language font
		Font 1		Font 1
		Font 6		Font 10
		Font 7		Font 6
		Font 10		

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.

A variation in the order of the language fonts or the elimination of font will change what is displayed with regard to all the translations/labels in all the languages.

1° Font	Lang. 1 -> Font 1 `ABCD abcd 1234`	Lang. 2 -> Font 2 ***ABCD abcd 1234***	Lang. 3 -> Font 1 `ABCD abcd 1234`
2° Font	Lang. 1 -> Font 6 ΑΒΧΔ αβχδ 1234	Lang. 2 -> Font 10 ***ABCD abcd 1234***	Lang. 3 -> Font 10 ***ABCD abcd 1234***
3° Font	Lang. 1 -> Font 7 ***ABCD abcd 1234***	Lang. 2 -> Font 1 `ABCD abcd 1234`	Lang. 3 -> Font 6 ΑΒΧΔ αβχδ 1234
4° Font	Lang. 1 -> Font 10 ***ABCD abcd 1234***	Lang. 2 -> Font 6 ΑΒΧΔ αβχδ 1234	Lang. 3 -> System font `ABCD abcd 1234`

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.

Language (✤):

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 fonts (✤):

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

Language fonts (✤):

Used to add project fonts. When ▫ is clicked you can access other functions, listed below.

Type of font:

Used to choose the family of font, native or Windows-based.

Project fonts:

Used to create the list of project fonts. If the Based on Windows option has been selected, click on the *Aggiungi* button to access the mask for creating fonts based on Windows. The mask is set to the parameters listed below.

Name (Based on Windows):

Name by which the font is defined. You are advised to assign this in such a way as to help future recognition.

Font base (Based on Windows):

This is the type of font (present on a Personal Computer) that is used to define the font in VTWIN.

Bold (Based on Windows):

Used to activate/deactivate an attribute.

Underlined (Based on Windows):

Used to activate/deactivate an attribute.

Italic (Based on Windows):

Used to activate/deactivate an attribute.

Barred (Based on Windows):

Used to activate/deactivate an attribute.

Extended font (Based on Windows):

Active only if the operative system loaded in the Personal Computer (Windows 2000 / XP / NT 4.00) allows the handling of fonts with over 256 characters (used typically for oriental languages) to be activated/deactivated.

⚠ **To use this function, the International Settings of the Operative System being used must be configured (see Operative System Manual and** "Chapter 15 -> Multilanguage support"**).**

Height (pixels) (Based on Windows):

Used to define the font dimensions (expressed in PIXELS).

Available fonts:

Displays the fonts that can be used as project fonts.

Preview:

Displays the form of the font selected.

Project settings

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

🗁 General options.

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.

Start page:

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

Beep when screen/key is pressed:

Used to enable an acoustic signal when the Touch Screen or a ▢ (with keyboard models) is pressed.

Enable screen saver:

The CCFL backlit display lamp of the VT automatically switches off after a given period; to switch on again touch the Touch Screen display or (in the case of keyboard models) a ▢.

Screen saver delay:

Determines the time after which the display lamp automatically goes off, if the screen saver has been activated.

Use 3D look for editable fields:

Used to choose whether throughout the project the modify enabled fields should be displayed three dimensionally.

🗀 Alarms.

Automatic change of context:

Used to move automatically into the alarms display page when an alarm is detected.

Automatic scrolling:

All the alarms present are displayed automatically in rotation.

Autoscroll delay:

Determine the interval between displaying one alarm and the next.

Beep on new alarms:

Used to make the VT emit an acoustic signal to announce the arrival of new alarm.

Ordering alarm buffer starting from most recent alarm:

Allows the display of the events contained in the history buffer for alarms in chronological order from the most recent to the earliest.

Alarm history:

The user can define how alarms in excess of the length of the buffer should be treated, FIFO (First In - First Out) or Ignore exceeding alarms.

Alarm signal position:

Used to determine where on the display the alarm signal should appear. (Touch Screen models only).

Date format:

The user can impose the format of the date to be displayed with alarms.

Time format:

The user can impose the format of the time to be displayed with alarms.

🗁 Messages.

Automatic change of context:

Used to move automatically into the messages display page when an alarm is detected.

Automatic scrolling:

All the messages present are displayed automatically in rotation.

Message signal position:

Used to determine where on the display the message signal should appear. (Touch Screen models only).

Show Time/Date:

When this check box is enabled the date and time display is activated. (VT505W only).

Date format:

The user can impose the format of the date to be displayed with messages.

Time format:

The user can impose the format of the time to be displayed with messages.

Alarm history and trend buffers.

The parameters set out below allow you to define the save criteria of the alarm history and trend buffers in the case of those terminals that have no buffered batteries. Saving the history allows the events to be saved into the non-volatile memory of the terminal, in such a way that they are not lost when the VT is switched off. The function cannot be deactivated.

Alarm history save interval (expressed in hours):

This determines the time interval after which the alarm history is no longer saved into non-volatile memory. The settable value of the count ranges from 2 to 12 hours with increments of 2 hours.

Preset alarm history save schedule:

Makes it possible to set the time at which the alarm history will be saved into the non-volatile memory.

Enables the second save schedule:

Makes it possible to set a second time at which the alarm history will be saved into the non-volatile memory.

Print settings

These allow the user to configure the print parameters valid for the entire project. This is active only if a printer has been declared in the project.

The parameters must be confirmed at least once to activate them.

Printer:

Used to select a printer out of those declared in the project.

Send Form-Feed at the end of the page/footer:

When this function is enabled, the sheet is expelled even if incompletely filled.

Rows per page:

This indicates the number of rows that make up a page.

Columns per page:

This indicates the number of columns that make up a page.

Left margin:

This indicates the number of columns to leave blank starting from the left of the sheet.

Header:

Allows the user to choose one out of those declared in the project.

Footer:

Allows the user to choose one out of those declared in the project.

Use default settings:

Used to establish whether the project’s global parameters will be used or new ones entered for the specific application.
(This 🕮 is contained in the 🗀 Alarms/Messages, Alarm history buffer, Hardcopy, Report).

Data memory structure

To explain what the data memory is it is necessary to give some examples.

Imagine we need to have a production cycle involving a range of different products. In such a case, each product differs from another because, although they have the same process parameters (ingredients), their quantities differ.

A set of parameters together with the quantities related to a given product are referred to as a Recipe. Different process parameters are called "recipe data". Let us look at an example of hot pressing.

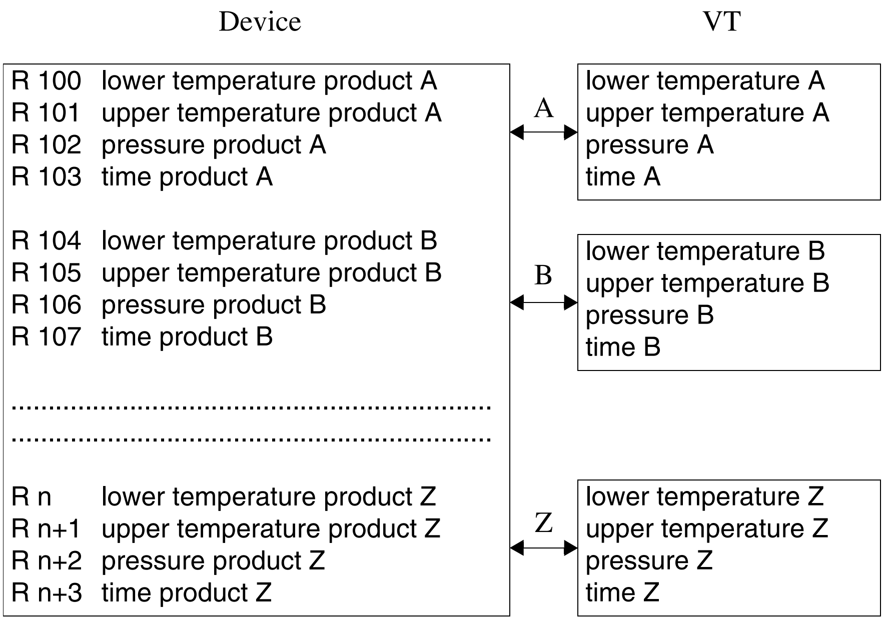
Parameters affecting the process (ingredients):

- temperature of upper die
- temperature of lower die
- pressure at bite
- bite time

The situation is described without the aid of the data memory.

In the VT various pages (one for each product) are programmed, each of which contains the values relating to the product.

The number of bits we reserve in the device connected is equal to the number of ingredients multiplied by the total number of products.

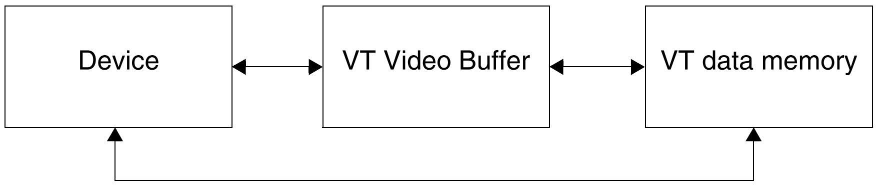


Note here the considerable quantity of data memory used and a certain degree of complication in creating the program for the device used to handle the pointers for selecting on the basis of the product concerned the part of the data to refer to.

The data memory of the VT, however, enables you to save the various quantities of ingredients for each product and reduce the number of data used in the device to the same as the number of ingredients.

In the VT you can program a single page containing the data to be stored in the data memory. This data refers to the ingredients used. The association of the set of quantities of the ingredients for each product is effected via a recipe identifying code.

Access to the VT data memory is given via a special instrument called “video buffer”: the data can be displayed within the page or accessed directly by the device.



The data memory in the VT is maintained by a battery. In those terminals where the use of a battery is not envisaged (see Hardware Manual) the recipes are normally managed and saved into the volatile memory. Saving them into the remanent memory is thus the responsibility of the user (see Page 6-32 -> “Automatically copy recipes into remanent memory:“ and/or “Chapter 4 -> Touch Button“ and/or “Chapter 4 -> Command area:“).

⚠ **In these types of terminal ALL the recipes in the volatile memory are lost when the terminal is switched off if they are not saved into the remanent memory.**

Now that we have explained the significance of the concepts data memory and recipe, we can proceed to introduce two examples; the first shows the minimum structure of a recipe, the second shows a recipe uses the extended structure.

Example of a recipe with a minimum structure.

For a recipe to have sense, it must be composed of an alphanumeric variable that unequivocally identifies it, which, as we have already see, is known as the Code (no two recipes can be introduced with the same code), and a numerical or alphanumeric variable that identifies the “ingredients” of the recipe.

The recipe code can be made up of from 2 to 40 characters.

Code	->	String Variable	->	2 - 40 Characters
Data 1	->	Numerical Variable or String	->	Data

If the recipe code and the data are substituted with values the result is the recipe.

CODE	MIX01	String Variable 20 Characters
Acid 01 Grams	10	Numerical Variable

Example of recipe with extended structure.

(A device with 32-bit registers is used. 12 data to be displayed in decimal are inserted in the recipe.)

The recipe created in the preceding example can be completed with other elements that make it richer in information for the user and that make it possible to permit the flow of data in the course of transfer to the device.

Besides the code an alphanumeric variable for the recipe comment is introduced.

Too comment too, like the code, may be composed of from 2 to 40 charac-

ters.
A recipe can contain the date and the time, in BCD format, of the last edit (only if the VT used has a clock and calendar), the checksum or control sum (calculated by taking the binary sum of all the bytes making up the recipe structure, excluding the checksum) and the number of data sent.

The recipe can be transmitted in synchrony with the device (see Page 6-29).

The 📖 described above might not be assigned to a variable of the device, but defined only in the VT. The definition in the VT must be used only if there is no need to give the information to the connected device.
Besides the variables listed above, the number of recipe ingredients increases.

A project may contain just one recipe structure; the maximum length depends on the type of VT (See Hardware Manual).

Example of a recipe structure.

Code		->	String Variable	->	2 - 40 Characters
Comment		->	String Variable	->	2 - 40 Characters
Data 1		->	Numerical Variable or String		
Data 2		->	Numerical Variable or String	->	Data
Data n		->	Numerical Variable or String		
DD	MM	->	Numerical Variable	->	Date of last edit 16Bit
YY	YY	->	Numerical Variable		
HH	MM	->	Numerical Variable	->	Time of last edit 16Bit
No. of data sent		->	Numerical Variable	->	No. of data sent 16 Bit
CKLow	CKHigh	->	Numerical Variable	->	Checksum of data 16 Bit

A recipe is always made up of a fixed part and a variable part The variable part is always that containing the data or ingredients of the recipe.

Name			
Comment			
DD	MM		
YY	YY	->	Fixed part
HH	MM		
No. of data sent			
CKLow	CKHigh		

Data item1		
Data item2	->	Variable part
Data n		

To continue with the example: the values that the various elements of the recipe can assume are inserted.

CODE	MIX01	String Variable 20 Characters
COMMENT	EXPERIMENTAL	String Variable 20 Characters
Acid 01 Grams	10	Numerical Variable
Acid 02 Grams	13	Numerical Variable
Acid 03 Grams	0	Numerical Variable
Substance 01 Grams	0	Numerical Variable
Substance 02 Grams	123	Numerical Variable
Substance 03 Grams	4	Numerical Variable
Additive 01 YES/NO	YES	String Variable 3 Characters
Additive 02 YES/NO	NO	String Variable 2 Characters
Additive 03 YES/NO	NO	String Variable 2 Characters
Additive 01 Grams	12	Numerical Variable
Additive 02 Grams	1	Numerical Variable
Additive 03 Grams	190	Numerical Variable
DDMM	2812	Numerical Variable
YYYY	1999	Numerical Variable
HHMM	2250	Numerical Variable
DATA SENT	12	Numerical Variable
CHECKSUM	3458	Numerical Variable

Example of Checksum calculation.

Make the binary sum of all the bytes except those of the variable containing the checksum value.The recipe code is composed of a string variable up to 20 characters in length; the content of this variable is MIX01, so to make the calculation you have to add up all the bytes.

CODE

MIX01	String Var.ble 20 Characters	20 x 1Byte = 20Byte

M	I	X	0	1	nil	nil	nil	nil	nil	nil	nil	nil	nil	nil	nil	nil	nil	nil	nil	Ascii
4D	49	58	30	31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hex

14F -> Sum total in Hex

We proceed in the same way for the comment. The comment of the recipe is also composed of a string variable up to 20 characters in length. The content of this variable, in our example, is EXPERIMENTAL; in this case too you have to find the sum of all the bytes.

COMMENT

EXPERIMENTAL	String Var.ble 20 Characters	20 x 1Byte = 20Byte

E	X	P	E	R	I	M	E	N	T	A	L	nil	nil	nil	nil	nil	nil	nil	nil	Ascii
45	58	50	45	52	49	4D	45	4E	54	41	4C	0	0	0	0	0	0	0	0	Hex

38E	-> Sum total in Hex

We proceed with the calculation of data 1. Data 1 of the recipe is composed of a 32-bit numerical variable, the content of this variable is 10; in this case too you have to find the sum of all the bytes.

DATA 1

10	Numerical Variable 32 Bits	32Bit / 8Bit = 4Bytes

10	␢	␢	␢	Dec
A	20	20	20	Hex

6A	-> Sum total in Hex

The same calculation must be carried out for the entire recipe structure.

MIX01	4D	49	58	30	31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14F
EXPERIMENTAL	45	58	50	45	52	49	4D	45	4E	54	41	4C	0	0	0	0	0	0	0	0	38E
10	A					20					20					20					6A
13	D					20					20					20					6D
0	0					20					20					20					60
0	0					20					20					20					60
123	7D					20					20					20					DD
4	4					20					20					20					64
YES	59					45					53					0					F1
NO	4E					4F					0					0					9D
NO	4E					4F					0					0					9D
12	C					20					20					20					6C
1	1					20					20					20					61
190	BE					20					20					20					11E
2812	1C					C					20					20					68
1999	13					63					20					20					53
2250	16					32					20					20					88
12	C					20					20					20					6C
3450	D7A																				D7A

Once the total has been obtained item by item calclate the sum of all the totals.

Synchronized recipe transmission:

When you decide to send a recipe from the VT to the device, it is likely that the device will not be ready to receive the recipe, and so will need a signal from the VT to get ready to receive before transmission starts. This procedure is called *Synchronized Recipe Transmission.*

The synchronization is brought about by what might be called a typical “Handshake” between the VT and the device.

A “Handshake” is the creation of synchronism between two intelligent inter-connected units. It implies the step-by-step execution of certain operations. The execution of each individual step is subject to reciprocal confirmations that, if not acknowledged by both parties, make it impossible to carry out the successive steps.

The synchronized recipe transmission function is directed by bit-organized handshake.

The cross-over confirmations between the terminal and the device connected thus occur as a function of the status of certain bits present in the data exchange areas which must of necessity have been defined in the user project.

The use or otherwise of the synchronized transmission option is at the discretion of the user.

If the synchronized recipe transmission option is NOT enabled, the VT will ALWAYS be able to operate the recipe transmission INDEPENDENTLY of whether the device is ready or not.

Example:

An plastic injection molding plant can press out 5 different details whose process parameters are controlled by 5 recipes. The plant has two levels: MANUAL and AUTOMATIC.

With the MANUAL level the plant preset can be operated before launching the working of a given detail.

With the AUTOMATIC level working on a previously preset detail can be started.

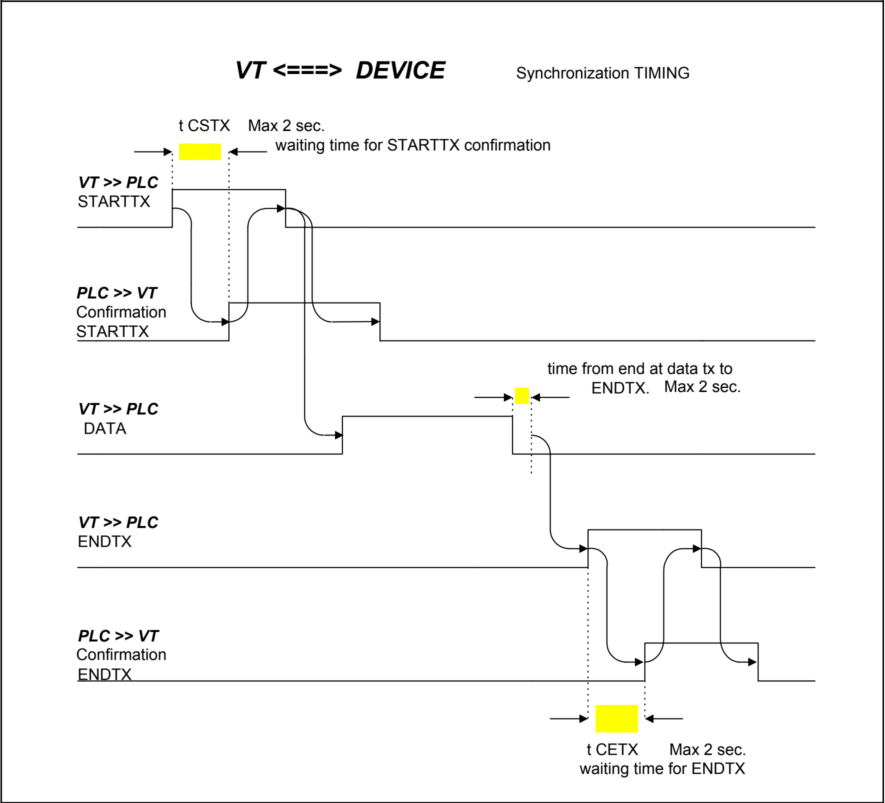
If the synchronized transmission option has been selected, the device can be programmed so as to prevent the reception of any recipe sent by the VT terminal while the productive process is in progress (AUTOMATIC).

If, on the other hand, the synchronized transmission option has been selected, it will STILL be possible (even in AUTOMATIC mode) to send a recipe from the VT to the device; this operation could prove somewhat dangerous as we are transferring process parameters that are completely incompatible with the detail currently being worked!!!

It is entirely at the discretion of the programmer whether to define the type of transmission on the basis of the type of plant being controlled. When the synchronization option is not selected it is the responsibility of the programmer to prepare the device in such a way as to preclude the risk of inappropriate actions on the part of the user.

Example of the Handshake for operating the synchronized recipe transmission.

Time chart for synchronized transfer



- STARTTX = bit to start transmission at 1.
- Confirm STARTTX = command SINCRORICETTA, bit to confirm start of recipe transmission at 1.
- ENDTX = bit to end recipe transmission to device at 1.
- Confirm ENDTX = command SINCRORICETTA, bit to end recipe transmission to device at 1.

Synchronized transmission involves 6 phases, starting from the moment the VT gets ready to transfer a recipe to the device. (See relevant Hardware Manual to determine the number of bits involved.)

Phase 1:

> The VT which is in recipe status area puts the bit to start transmission at 1.

Phase 2:

> The device realizes that the VT is ready to transmit, sends the command Synchronize recipe to the VT and puts the bit confirming the start recipe transmission at 1. The bit is in the word parameter 1.

Phase 3:

> The VT is aware of the reply by the device and puts the command word at 0 and the bit to start transmission at 0.

Phase 4:

> At the end of the transmission, the VT puts at 1 the bit to end recipe transmission to the device into the recipe status area.

Phase 5:

> The device registers that the VT has finished transmitting, sends the command Synchronize recipe to the VT and puts the bit for ending the transmission of the recipe to the device at 1. This bit is to be found in the word called parameter 1.

Phase 6:

> The VT registers that the device has received the recipe and puts the command word at 0 and the bit for ending the transmission of the recipe to the device at 0.

If the Handshake times set out in the previous table are not respected in the course of the transfer (phases 1 to 6), the VT puts the status of the bit for the synchronized recipe transfer time-out in the recipe status area at 1. The device responds with a command SINCRORICETTA, and status of the bit for the synchronized recipe transfer time-out is set at 1 in the word parameter 1; at this point the VT sets the status of the command word and the bit at 0. The transfer has NOT been carried out.

The recipe structures have related to them system variables; this means that the value is not related to the device but is contained within the terminal.

The variables are:

- Recipes written
- Recipes remaining
- Date recipe made
- Time recipe made

These variables can be introduced by introducing a Numerical, Dynamic, Bar-value field. (See “Chapter 4 -> Numerical field“)

To create a recipe structure the following parameters must be defined.

Recipe management enable:

Used to manage the recipes.

Automatically copy recipes into remanent memory:

Allows recipes to be saved into the remanent memory independent of the command used. The operation of saving them into the remanent memory is still the responsibility of the user, whether by means of an internal command or by means of a command from the device (see “Chapter 4 -> Touch Button“ and/or “Chapter 4 -> Command area:“).

Synchronized recipe transfer:

Used to define the type of transfer.

🗀 Code and comment.

Variable:

Used to define which variable, internal or related to the device, shall contain the recipe code.

Number of characters in code:

Indicates the maximum length of the recipe codes.

Comment enabled:

Used to assign a comment to a recipe.

Number of characters in comment:

Indicates the maximum length of the recipe comment.

Variable comment:

Used to determine which variable, internal or related to the device, shall contain the comment.

Items.

Items:

Used to list which variables shall be contained in the recipe.

Infos.

Enable hour:

Used to determine which variable, internal or related to the device, shall be assigned to the given field. (The time given is that of the last transfer).

Enable date:

Used to determine which variable, internal or related to the device, shall be assigned to the given field. (The date given is that of the last transfer).

Enable Checksum:

Used to determine which variable, internal or related to the device, shall be assigned to the given field. (See also "Chapter 6 -> Example of Checksum calculation.")

Enable No. of Items:

Used to determine which variable, internal or related to the device, shall be assigned to the given field.

Public data

Used to specify which variables in the context of an ESA-NET network must be shared with the other participants in the network. Both variables and memory areas can be made public.

The maximum length of the public objects is 60 Bytes; excess lengths will be truncated. To avoid this, you are advised to create more than one object of the appropriate length (given a length of 120 Bytes use two objects of 60 Bytes).

The maximum number of public objects depends on the terminal

(see Hardware Manual), with a total of 1024 bytes.

Windows

Horizontal arrangement

With this the active windows can be displayed horizontally.

Vertical arrangement

With this the active windows can be displayed vertically.

?

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 VTWIN...

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 VTWIN is installed, the second allowing you to get information on VTWIN installation.

Chapter 7 Using VTWIN

This chapter consists of a total of 8 pages.

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 the Configurator menu icons

The table lists all the icons of the menu together with their meanings.

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

Table 7.1: List of Configurator menu icons, respective pulldown menus and meanings

Tools Bar	Pulldown Menu	Action	Selection Required
🗋	***File > New***	Creates a new project (configuration).	--
📂	***File > Open***	Opens an existing project (configura-tion).	--
💾	***File > Save***	Saves a project on disk.	--
🖨	***Tools > Print…***	Prints the project.	Yes
✕	***Edit > Delete***	Deletes object selected in configuration.	Yes
📋	***Edit > Property***	Modifies the property of a project (name and comment).	Yes
🔌	**--**	Used to connect a VT to an ESA-NET network.	Yes
🔌	**--**	Used to disconnect a VT from an ESA-NET network	Yes
?	***? > Summary and index***	Calls up the Help on line.	--

-- Option not valid for this menu

Meaning of the Editor menu icons

The table lists all the icons of the menu together with their meanings.

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

Table 7.2: List of Editor menu icons, respective pulldown menus and meanings (Part 1 of 4)

Tools Bar	Pulldown Menu	Action	Selection Required
	Tools > Project compilation	Used to compile the project.	--
	Tools > Project transmission	Used to transfer the project to the panel.	--
	Tools > Download with Modem	Allows you to transfer the project to a remote panel using a modem.	--
✂	***Edit > Cut***	Saves a selection in the notes and deletes the object selected from the page.	Yes
	Edit > Copy	Saves a selection in the notes.	Yes
	Edit > Paste	Inserts a selection from the notes into the page .	No
✕	***Edit > Delete***	Deletes the object selected from the page.	Yes
✕	***Edit > Erase all***	Deletes all the objects in the page.	No
	Edit > Duplicate	Duplicates the object selected.	Yes
	Edit > Build library	Saves a selection in a file on disk.	Yes
	Object > library	Inserts a library in the page.	No
↶	***Edit > Undo***	Each keystroke undoes the last action performed.	No
↷	***Edit > Redo***	Each keystroke restores the action undone.	No
🔍	***Edit > Zoom up***	Increases the degree of enlargement of the page displayed.	No

-- Option not valid for this menu

Table 7.2: List of Editor menu icons, respective pulldown menus and meanings (Part 2 of 4)

Tools Bar	Pulldown Menu	Action	Selection Required
🔍	*Edit > Zoom down*	Decreases the degree of enlargement of the page displayed.	No
3D	*Modify > 3D effect*	Determines the display as having a 3D effect for the fields with the attribute Modify enabled.	Yes
🟥	*Edit > Colors*	Sets the colors of an object.	Yes
#	*Edit > Grid*	Assigns a given number of pixels as a minimum movement.	No
▨	*Edit > Show touch areas*	Displays all the touch aeas present in the work page	No
?	*? > Index*	Calls up the Help on line.	--
?	*? > Search for help on…*	Activates the “search for help” function of the help on line.	--
A	*Edit > Font choice*	Permits a character font to be chosen from among those present in VTWIN.	Yes
R	*Edit > Reverse*	Displays a label in negative.	Yes
A^1	*Edit > Dimension > X1*	Sets the font dimension as x1. Only for native fonts; not applicable to Windows-based fonts.	Yes
A^2	*Edit > Dimension > X2*	Sets the font dimension as x2. Only for native fonts; not applicable to Windows-based fonts.	Yes
A^4	*Edit > Dimension > X4*	Sets the font dimension as x4. Only for native fonts; not applicable to Windows-based fonts.	Yes
▣	*Edit > Level > Top*	Moves the object selected above any other objects.	Yes
▣	*Edit > Level > Bottom*	Moves the object selected below any other objects.	Yes
▣	*Edit > Level > Up*	Moves the object selected above the antecedent object.	Yes
▣	*Edit > Level > Down*	Moves the object selected below any preceding object.	Yes

-- Option not valid for this menu

Table 7.2: List of Editor menu icons, respective pulldown menus and meanings (Part 3 of 4)

Tools Bar	Pulldown Menu	Action	Selection Required
	Edit > Align > Leftwards	Align the objects selected with the left border of the object furthest from the center of the selection.	Yes
	Edit > Align > Rightwards	Align the objects selected with the right border of the object furthest from the center of the selection.	Yes
	Edit > Align > Upwards	Align the objects selected with the upper border of the object furthest from the center of the selection.	Yes
	Edit > Align > Downwards	Align the objects selected with the lower border of the object furthest from the center of the selection.	Yes
	Edit > Align > Vertical center	Aligns the objects selected with the central vertical axis of the selection containing them.	Yes
	Edit > Align > Horizontal center	Aligns the objects selected with the central horizontal axis of the selection containing them.	Yes
	Edit > Mirror > Vertical	Inverts the objects selected vertically.	Yes
	Edit > Mirror > Horizontal	Inverts the objects selected horizontally.	Yes
	Object > None	Puts the pointer in readiness mode.	--
	Fields > Label	Allows the insertion of a multilanguage label.	--
	Fields > Numeric	Allows the insertion of a numerical field.	--
	Fields > Ascii	Allows the insertion of an ASCII field.	--
	Fields > Dynamic	Allows the insertion of a dynamic field.	--
	Fields > Bar	Allows the insertion of a bar field.	--
	Fields > Symbolic	Allows the insertion of a symbolic field.	--
	Fields > Mobile symbolic	Used to insert a mobile symbolic field.	--

-- Option not valid for this menu

Table 7.2: List of Editor menu icons, respective pulldown menus and meanings (Part 4 of 4)

Tools Bar	Pulldown Menu	Action	Selection Required
	Fields > Date	Allows the insertion of the date and/or time.	--
	Field > Trend	Allows the insertion of a trend.	--
	Fields > Touch button	Allows the insertion of a button.	--
	Fields > Touch area	Allows a touch area to be defined.	--
	Object > Line	Permits the drawing of a line.	--
	Object > Rectangle	Permits the drawing of a rectangle and/or square.	--
	Object > Ellipse	Permits the drawing of an ellipse and/or circle.	--
	Object > Arc	Permits the drawing of an arc.	--
	Object > Bitmap image	Permits the drawing of a bitmap image.	--
	Fields > Indicator	Permits an indicator to be inserted.	--
	Fields > Sliding potentiometer	Permits insertion of sliding potentiome-ter.	--
	Fields > Sliding selector	Permits a sliding selector to be inserted.	--
	Fields > Potentiometer knob	Permits insertion of potentiometer knob .	--
	Fields > Selector knob	Permits a selector knob to be inserted.	--

-- Option not valid for this menu

Chapter 8 Creating a project with VTWIN

This chapter contains a total of 100 pages.

The first thing is to understand which functions the VT 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 (alarms, Start Page, etc.).

Although this seems obvious, often for a variety of reasons it is forgotten and what happens is that the operation of the VT 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 VT.

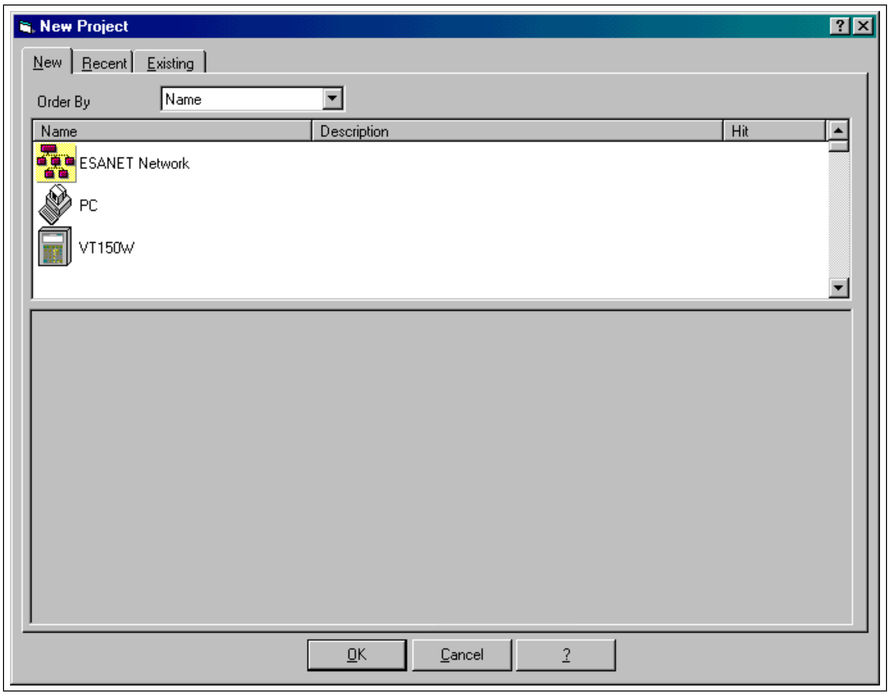
This being clear, we can move on to creating the project. It is necessary to establish the graphical structure of the project, by which we mean the look of the pages and their contents, to know which variables to use and which alarms and/or messages (if any), which data exchange area to use (if necessary). It is necessary too to have defined the sequences (in the case of non-touch screen panels) 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 a Touch Screen graphic panel connected to a SAIA PCD PLC device. Using the example of this plant, we will see how to control the temperatures and pressures; it will be necessary to control the fermentation process by means of a special page that gives complete control of the autoclaves; recipe pages will have to be created to manage the quantities of the various ingredients. There will be pages for setting temperature and pressure values for controlling the condition of the must, that is, for stopping spontaneous, “wild”, fermentation; other pages for controlling the preparation, that is the phase in which the must is prepared for fermentation. The plant will be monitored by means of alarms and messages. Passwords will be used to protect sensitive data, and the texts will be written in two languages.

Such an example will give us an opportunity of analyzing all the functions the panel offers.

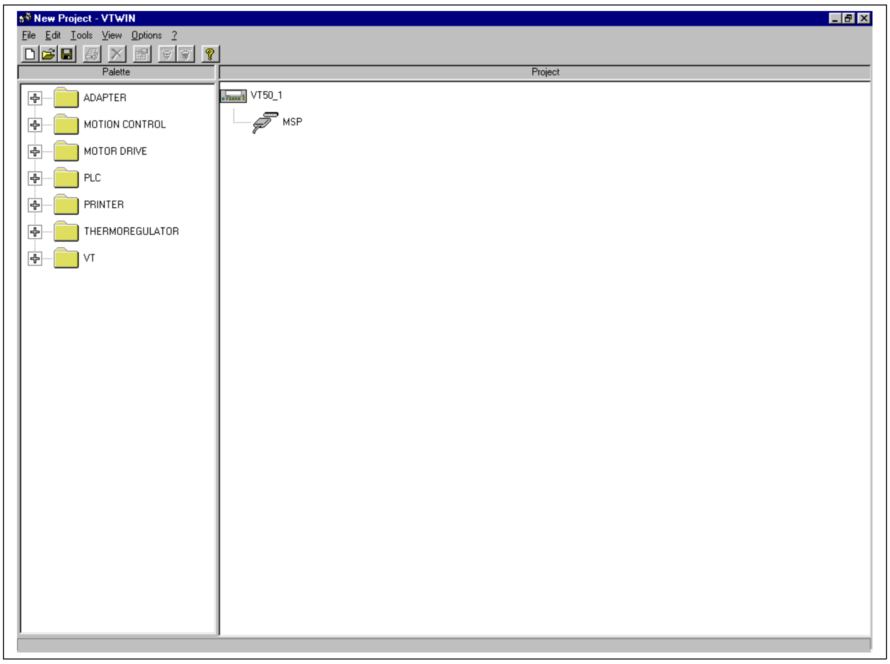
Creating the project

Click on 🕮 ***File > New*** (See “Chapter 5 -> New…“)



Select the icon Single VT.

Click on ☐ OK.



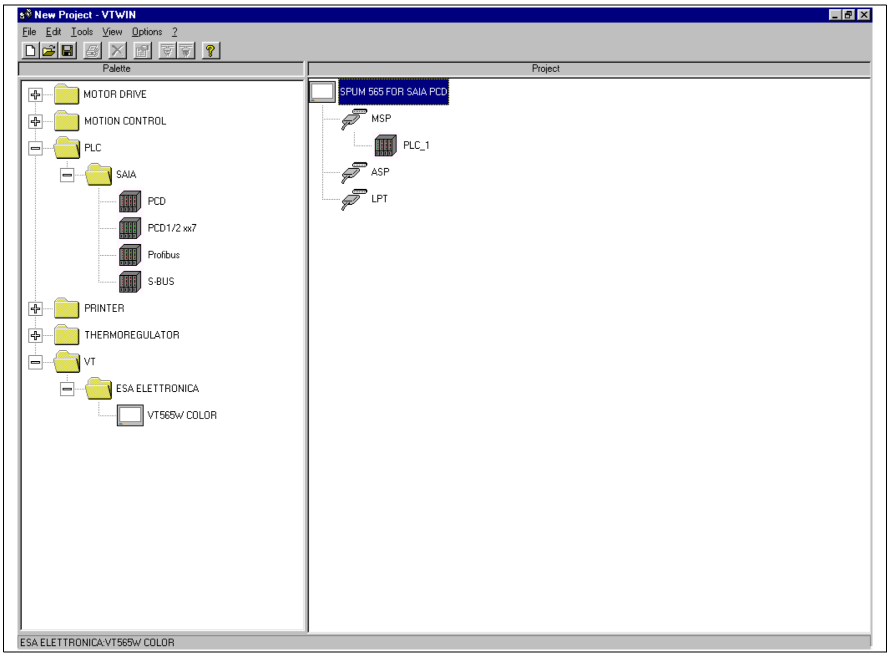
Select the icon on the Project side, then click on the 🕮 Tools>Convert, and choose the VT you want from the list.

Confirm with OK.

Rename the project by selecting the VT, then click on the Edit>Rename, and assign a new name as illustrated.

Select the device to be connected to the VT; drag it from the Device side to the MSP port on the Project side.

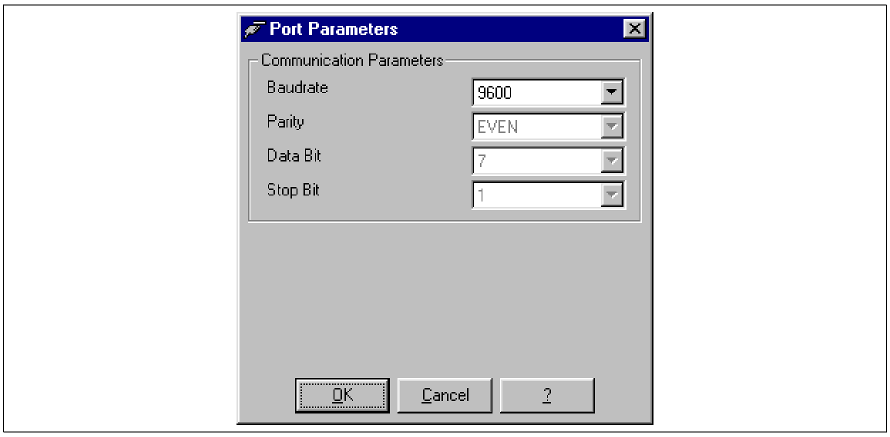
Select the MSP icon required, then click on the Edit>Property; the following mask then appears.



Set the communication parameters of the VT's serial port.

These must be the same as those of the device.

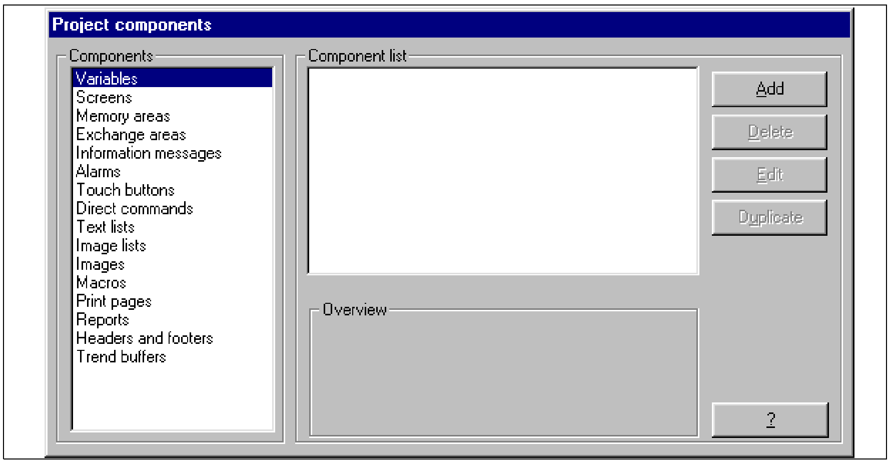
Click on the OK.


Port Parameters

Communication Parameters

Baudrate	9600
Parity	EVEN
Data Bit	7
Stop Bit	1

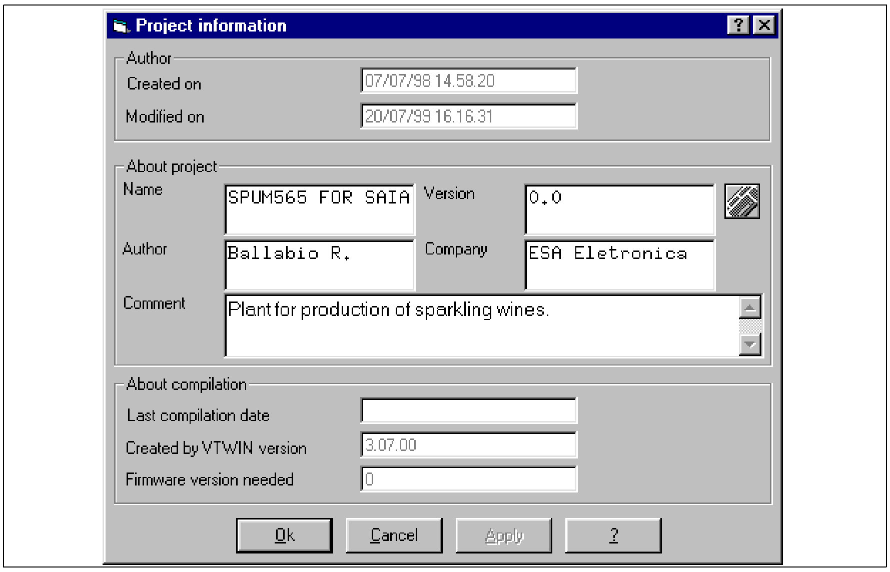
Once the parameters have been set, double-click on the VT icon on the Project side; the following mask then appears.



The project has now been opened.

Project information

Click on ***Configure > Project information*** (See “Chapter 6 -> Project information“)



Project information

Author
Created on: 07/07/98 14.58.20
Modified on: 20/07/99 16.16.31

About project
Name: SPUM565 FOR SAIA
Version: 0.0
Author: Ballabio R.
Company: ESA Eletronica
Comment: Plant for production of sparkling wines.

About compilation
Last compilation date:
Created by VTWIN version: 3.07.00
Firmware version needed: 0

Compile the 📖 required.

Click on OK.

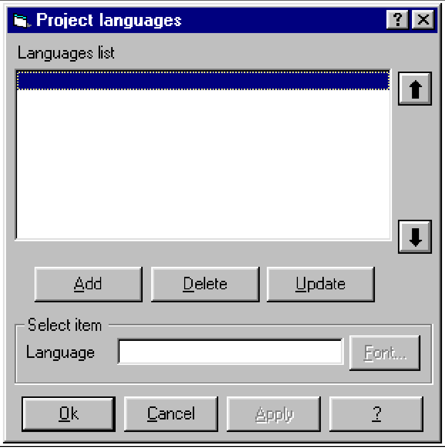
Setting project languages

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

Click on ***Configure > Project languages*** (See "Chapter 6 -> Project language")

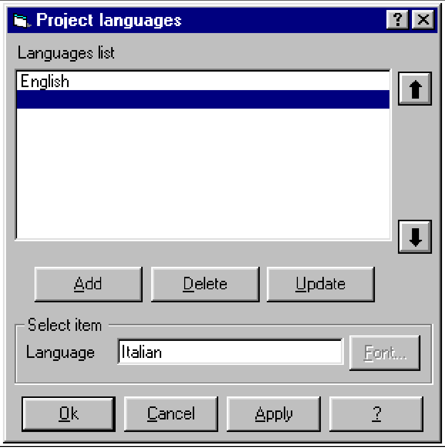
Select the box Language, and insert the mother tongue of the project; digit Italian.

Click on ▢ Add.



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

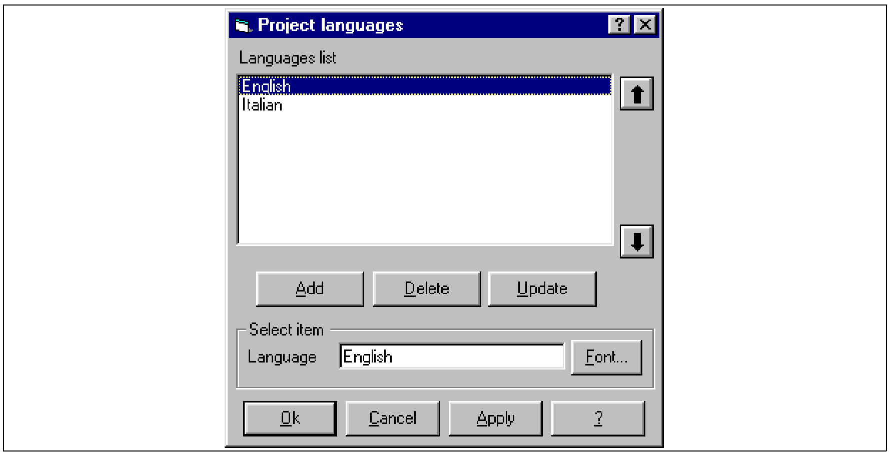
Click on ▢ Add.



English

Language: Italian

At this point the fonts of the languages are set.

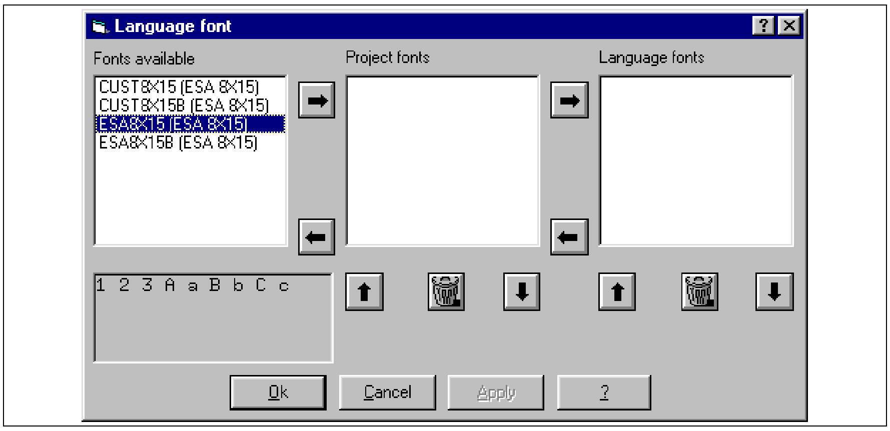


Languages list
English
Italian

Language: English

Select 🕮 Italian.

Click on ☐ Font...



Fonts available
CUST8X15 (ESA 8X15)
CUST8X15B (ESA 8X15)
ESA8X15 (ESA 8X15)
ESA8X15B (ESA 8X15)

Project fonts

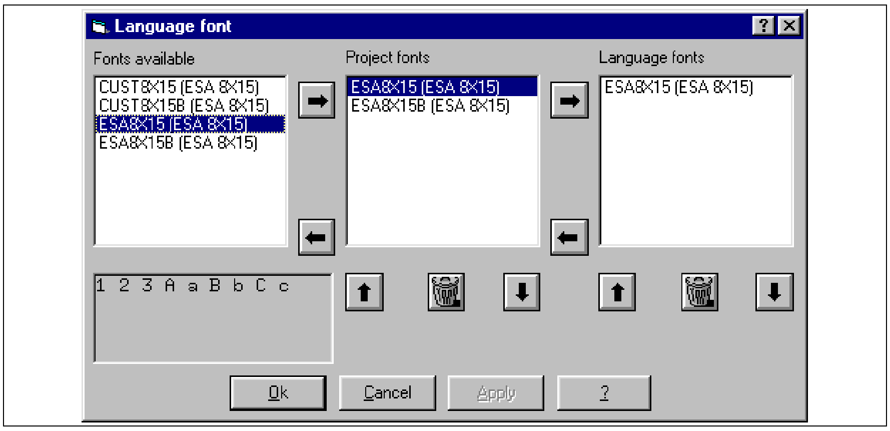
Language fonts

1 2 3 A a B b C c

Select 🕮 ESA8X15(ESA8X15) in the Fonts available list.

Click on ➡.

Repeat the procedure selecting the 🕮 ESA8X15B (ESA8X15).



Fonts available
CUST8X15 (ESA 8X15)
CUST8X15B (ESA 8X15)
ESA8X15 (ESA 8X15)
ESA8X15B (ESA 8X15)

Project fonts
ESA8X15 (ESA 8X15)
ESA8X15B (ESA 8X15)

Language fonts
ESA8X15 (ESA 8X15)

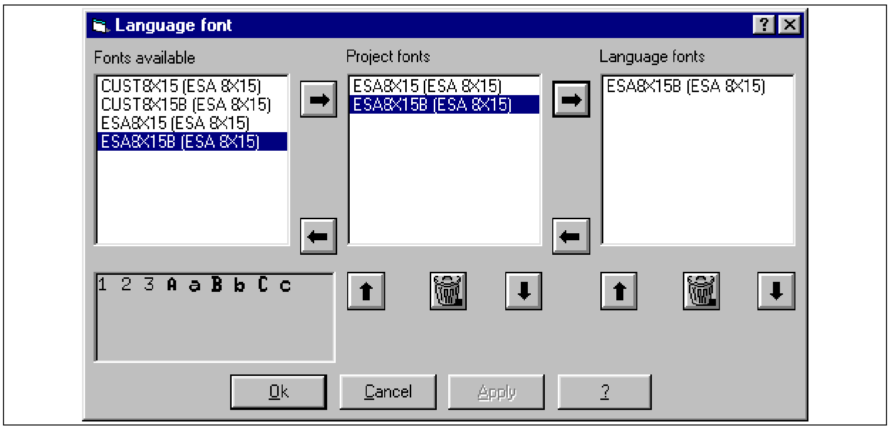
1 2 3 A a B b C c

Select 🕮 ESA8X15 (ESA8X15) from the Project fonts list.

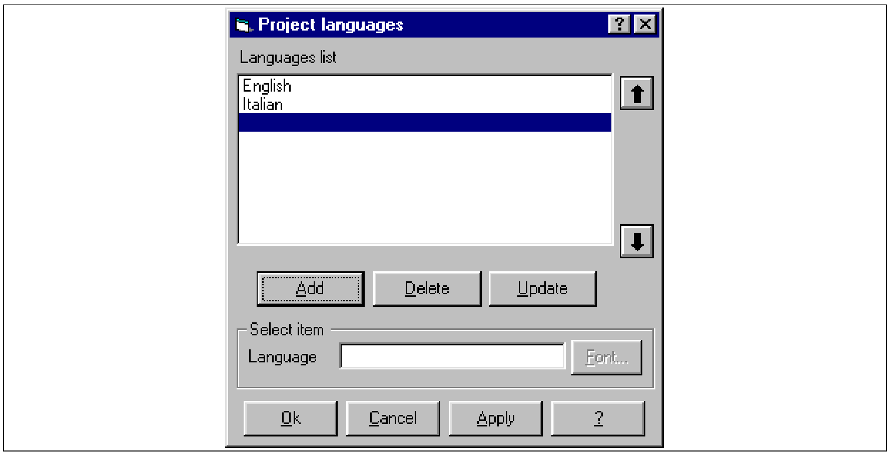
Click on ➡

Cliccare su ☐ Ok.

Repeat this operation for English, but choose it as a Font of the language ESA8X15B(ESA8X15).



To activate the language settings, 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.

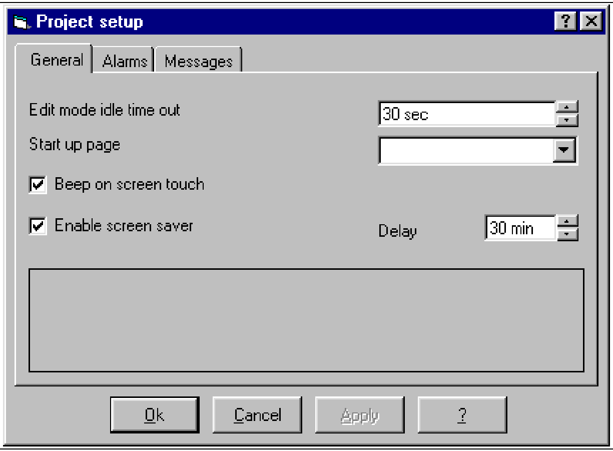
Project setup

Click on ***Configure*** > ***Project setup*** (See “Chapter 6 -> Project information“)

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

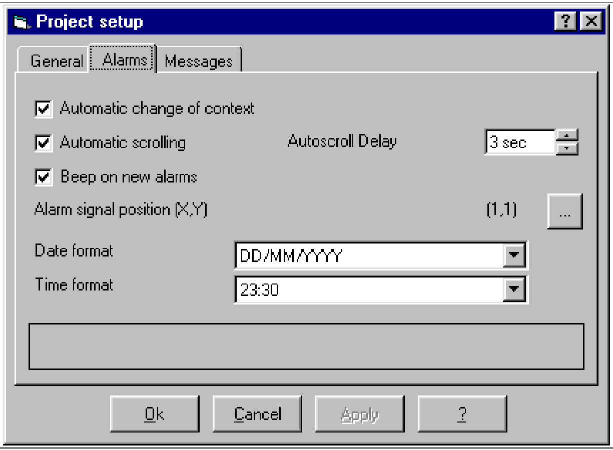
Compile as illustrated.

Click on Alarms.



Edit mode idle time out: 30 sec
Start up page:
☑ Beep on screen touch
☑ Enable screen saver — Delay: 30 min

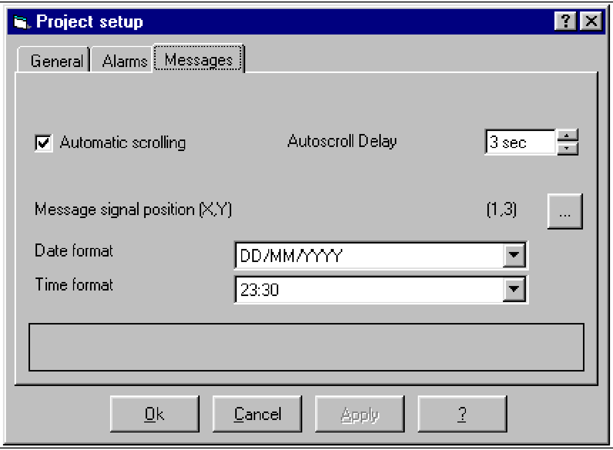
Set parameters as illustrated.



☑ Automatic change of context
☑ Automatic scrolling — Autoscroll Delay: 3 sec
☑ Beep on new alarms
Alarm signal position (X,Y): (1,1)
Date format: DD/MM/YYYY
Time format: 23:30

Click on Messages; as in the previous mask we will set the parameters as illustrated.

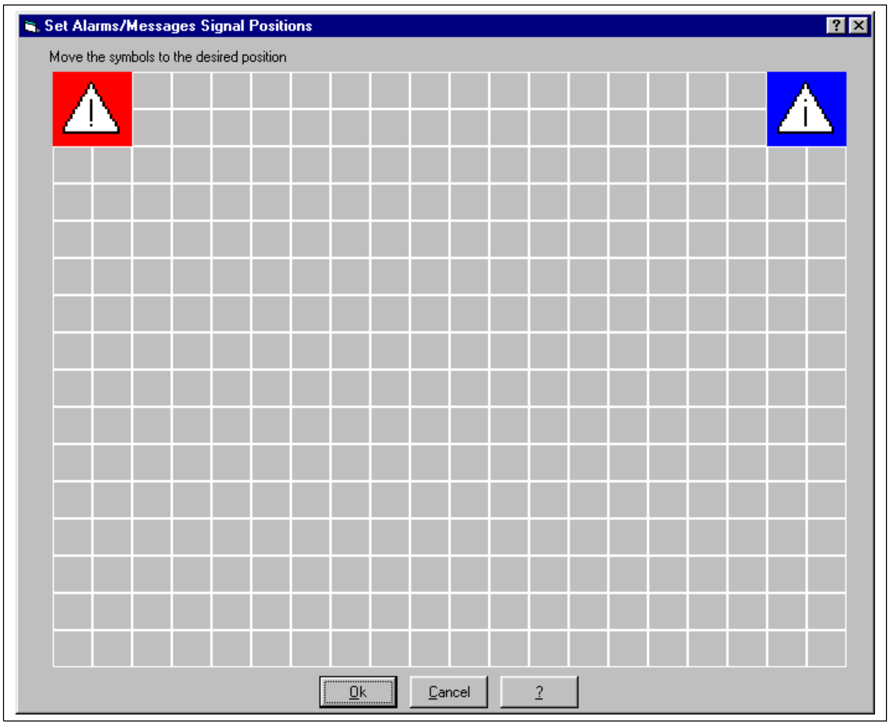
To position the symbol signaling the presence of an alarm and/or message, click on ... .



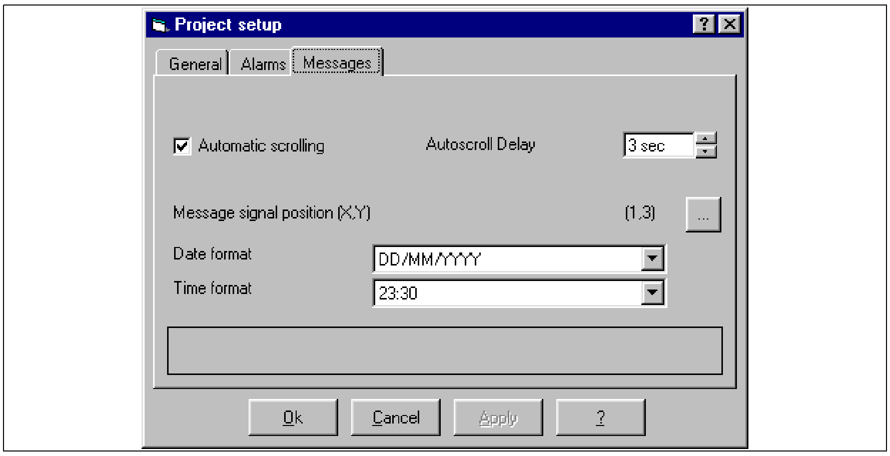
☑ Automatic scrolling — Autoscroll Delay: 3 sec
Message signal position (X,Y): (1,3)
Date format: DD/MM/YYYY
Time format: 23:30

We will arbitrarily fix the position as illustrated.

Click on OK.


Set Alarms/Messages Signal Positions
Move the symbols to the desired position
Ok
Cancel
?

Click on OK.



☑ Automatic scrolling

Autoscroll Delay: 3 sec

Message signal position (X,Y): (1,3)

Date format: DD/MM/YYYY

Time format: 23:30

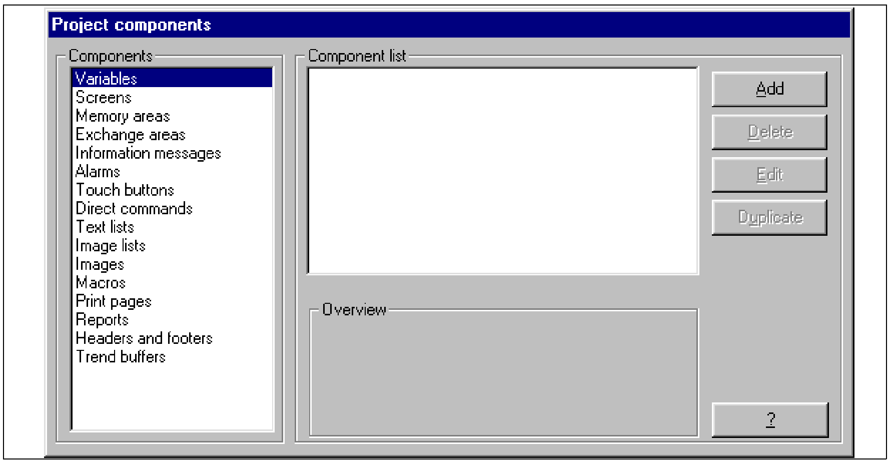
In this way all parts of the project is parameterized; now we can start to insert the elements that make up the project.

Inserting variables

You can proceed in two ways, inserting all the elements like variables, touch buttons, 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 the 🕮 Variables.

Click on the ☐ Add.

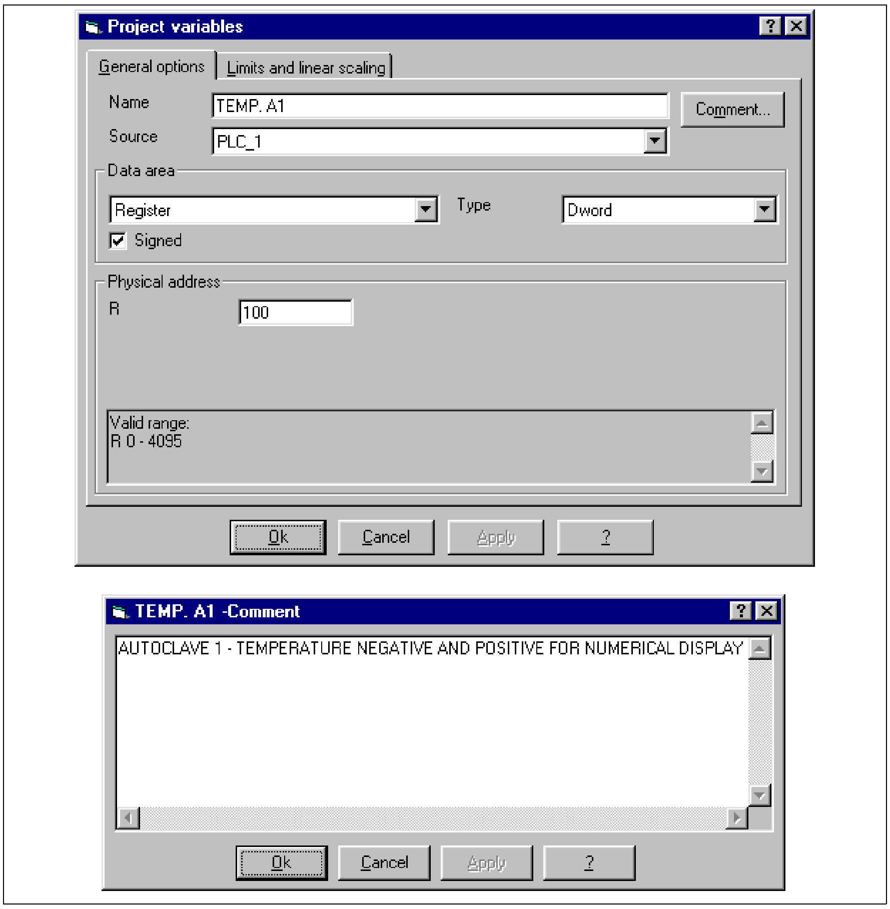


Assign a name to the variable so as to be able to recognize it easily in the list: TEMP.A1 Add a comment to the variable by clicking on ☐ Comment.

Click on OK, then parametrize as illustrated.

The comment should be as exhaustive as possible.

Click on Limits and Linear scaling.

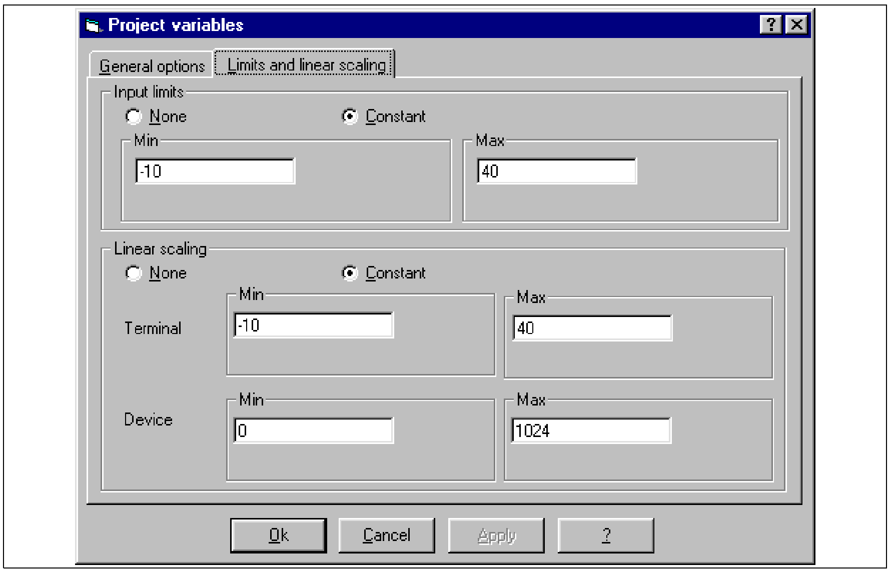


Name: TEMP. A1
Source: PLC_1
Data area
Register
Type: Dword
☑ Signed
Physical address
R: 100
Valid range:
R 0 - 4095

AUTOCLAVE 1 - TEMPERATURE NEGATIVE AND POSITIVE FOR NUMERICAL DISPLAY

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

Click on Ok.



Input limits: None [✓ Constant]
Min: -10
Max: 40

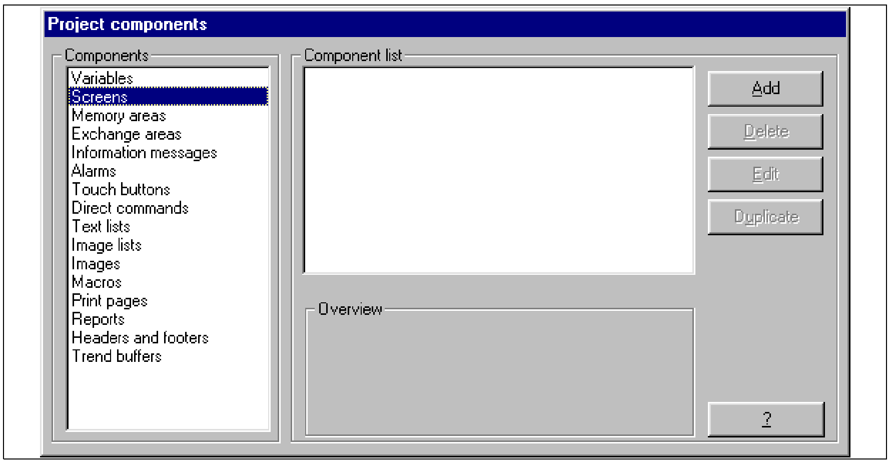
Linear scaling: None [✓ Constant]
Terminal — Min: -10, Max: 40
Device — Min: 0, Max: 1024

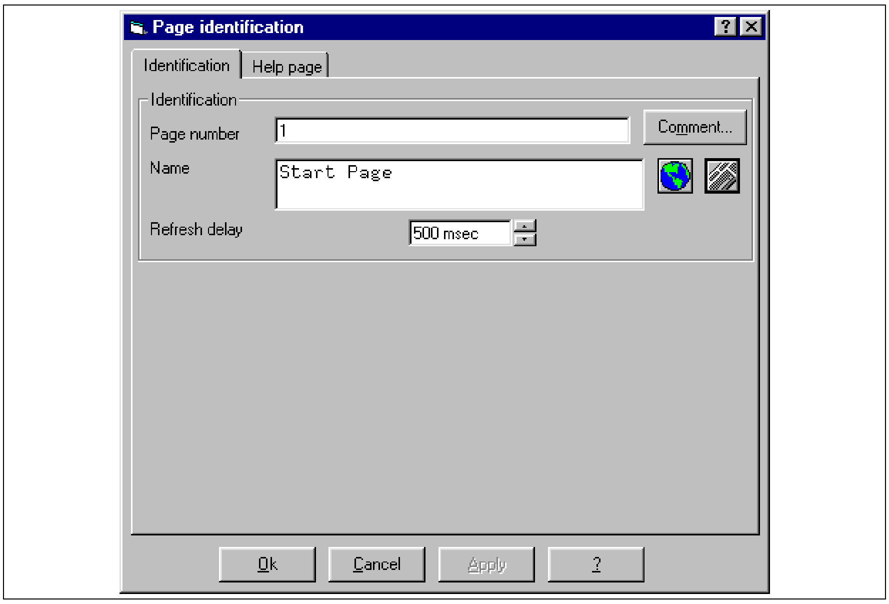
Repeat the operations described above to insert all the variables necessary (See project offered as example).

Inserting pages

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

Click on the Add.





Page identification

Identification | Help page

Identification

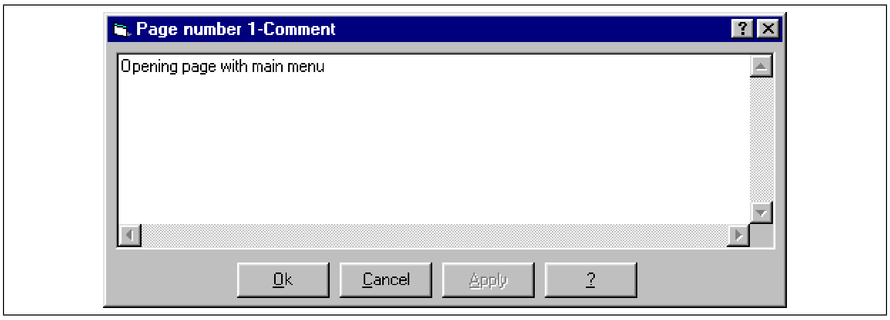
Page number: 1

Name: Start Page

Refresh delay: 500 msec

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

Add a comment to the page by clicking on ▢ Comment.

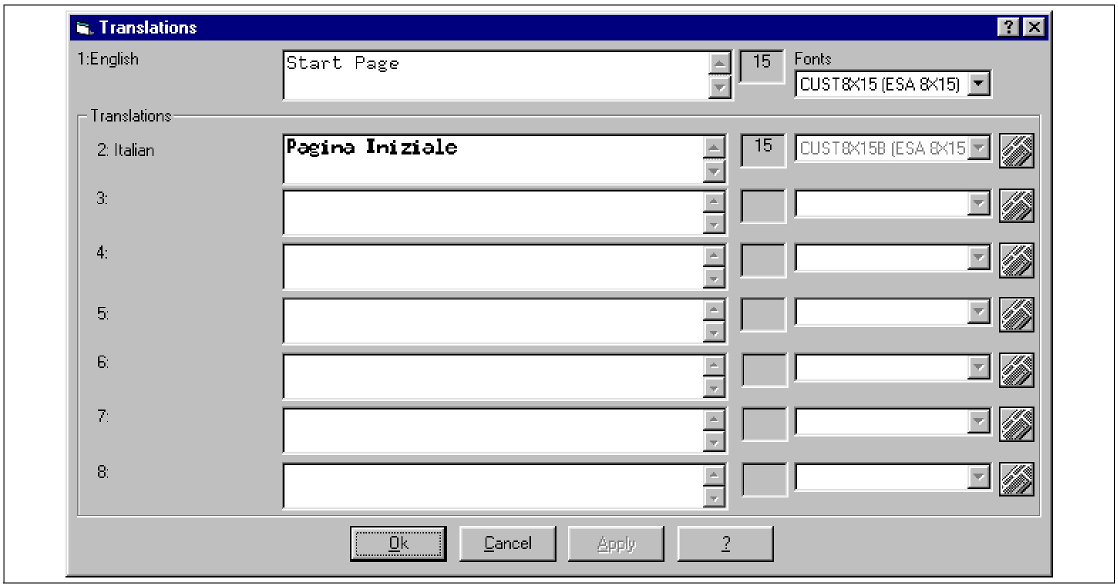


Page number 1-Comment

Opening page with main menu

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

To call up the window for translations click on



Translations

1:English — Start Page — 15 — Fonts: CUST8X15 (ESA 8X15)

Translations

2: Italian	Pagina Iniziale	15	CUST8X15B (ESA 8X15
3:			
4:			
5:			
6:			
7:			
8:			

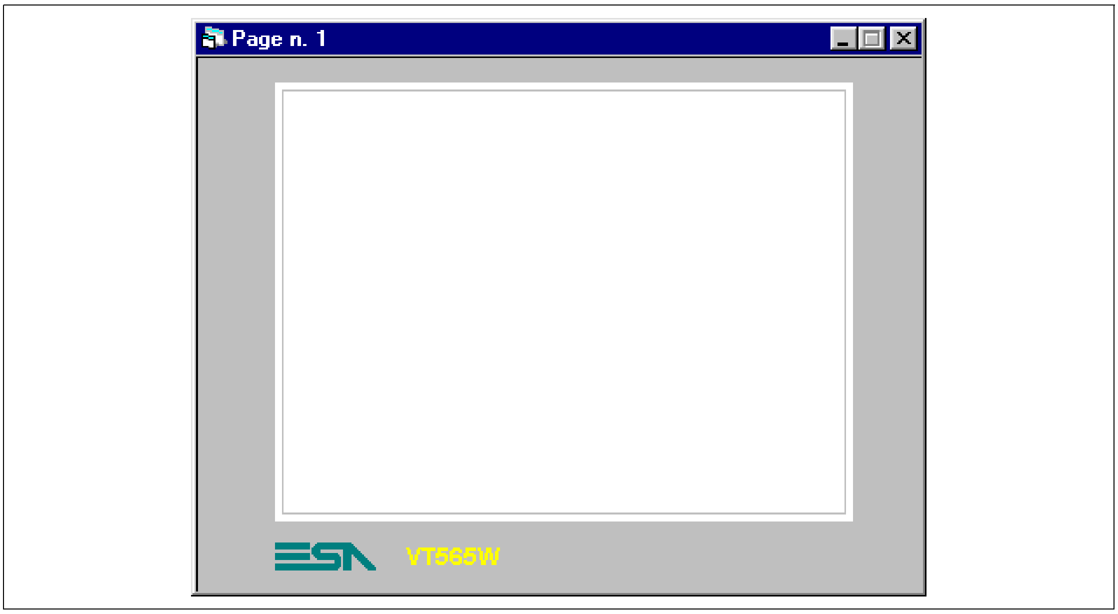
The text of the translations cannot be longer than the mother tongue text. If the translation should require more characters the mother tongue can be lengthened by adding spaces.

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

Click on OK to accept the translation and turn back to the previous mask.

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 - Start page

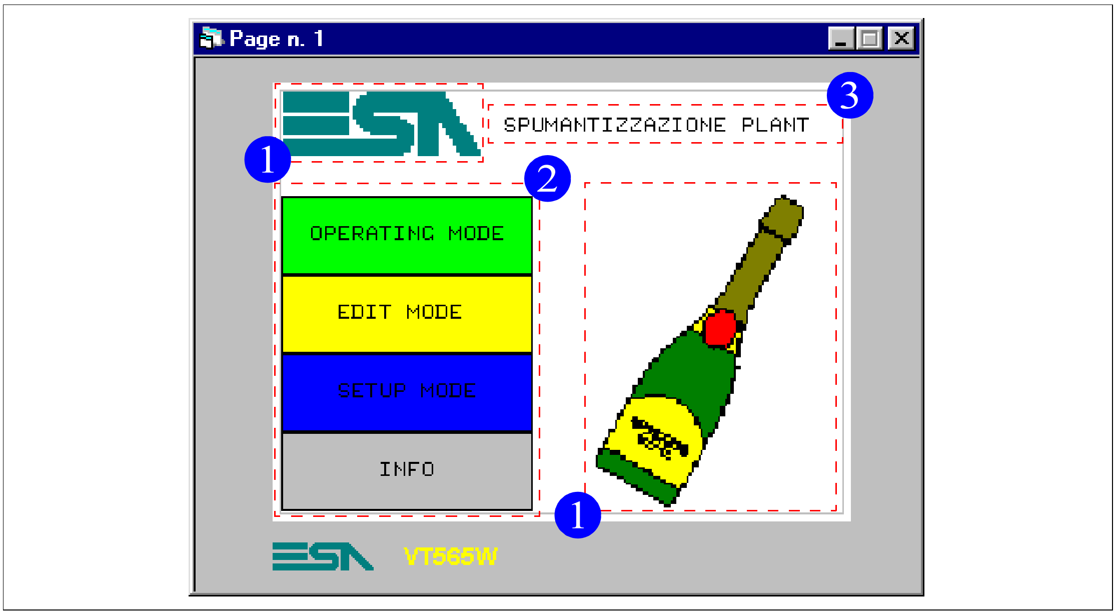
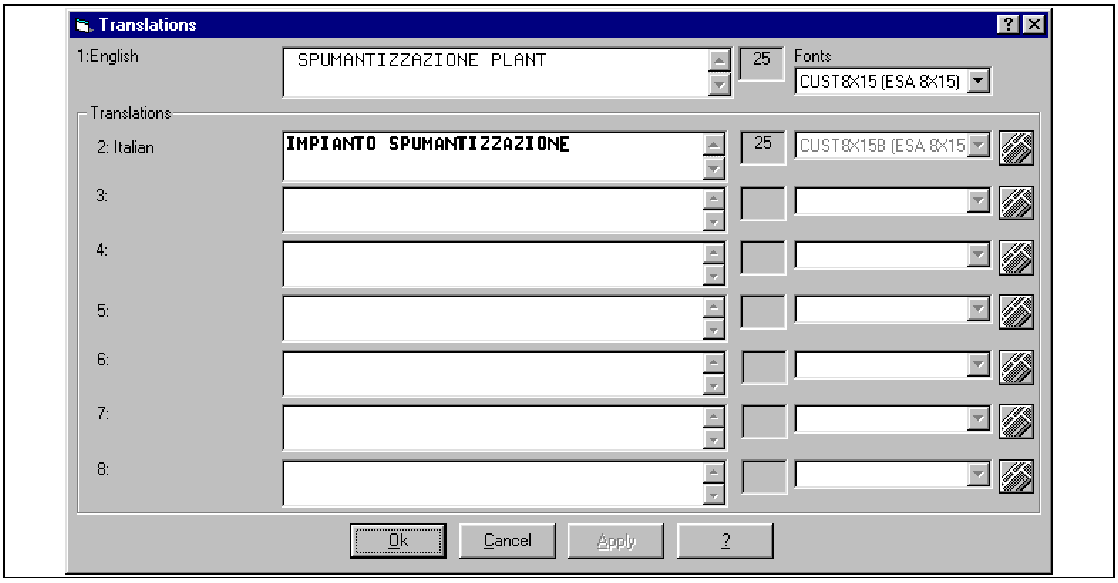


Table 8.1: Elements appearing in page no.1

IMAGES (1)	
	ESA
	BOTTLE
TOUCH BUTTONS (2)	
	OPERATING MODE
	EDIT
	SETUP MODE
MULTILANGUAGE LABELS (3)	
	SPUMANTIZZAZIONE PLANT

This page is seen when the VT 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 -> Multi-language label“). Click on AB and position the cursor on the page; edit the text and confirm,



1:English	SPUMANTIZZAZIONE PLANT	25	CUST8X15 (ESA 8X15)
2: Italian	IMPIANTO SPUMANTIZZAZIONE	25	CUST8X15B (ESA 8X15
3:			
4:			
5:			
6:			
7:			
8:			

Insert text and click on Ok to confirm.

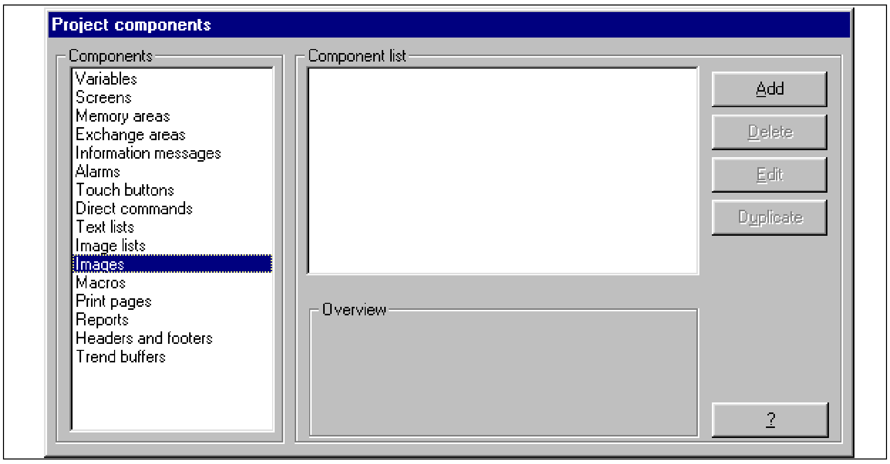
The final positioning of the label takes place later.

Now the logo is put in as a bitmap image.

The image can be created with any image editor. The image must be of 16 DOS colors and not exceed the maximum dimensions of the display. It is moreover advisable to respect the display grid. (See relevant Hardware Manual).

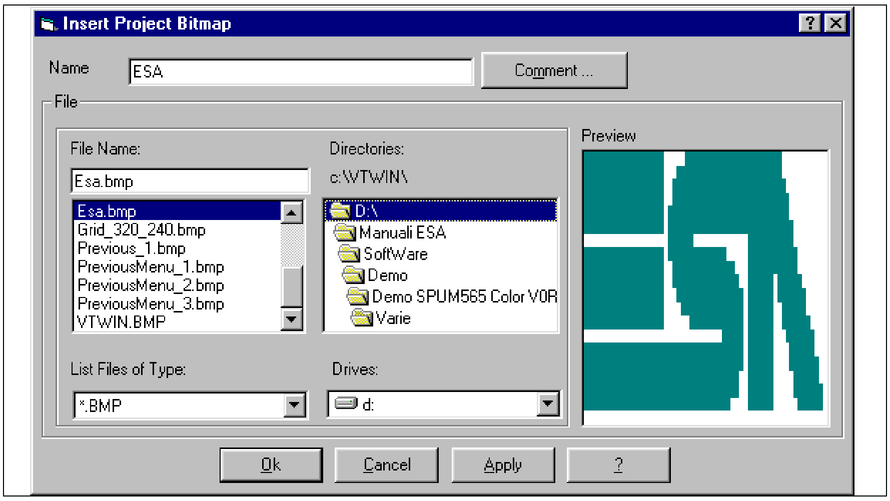
Once the image ESA.BMP has been created, it has to be inserted into VTWIN.

To insert image, select the 🕮 Images (See "Chapter 4 -> Images").


Project components

Components: Variables, Screens, Memory areas, Exchange areas, Information messages, Alarms, Touch buttons, Direct commands, Text lists, Image lists, Images, Macros, Print pages, Reports, Headers and footers, Trend buffers

Click on the ▢ Add.


Insert Project Bitmap

Name: ESA

File Name: Esa.bmp

Esa.bmp, Grid_320_240.bmp, Previous_1.bmp, PreviousMenu_1.bmp, PreviousMenu_2.bmp, PreviousMenu_3.bmp, VTWIN.BMP

Directories: c:\VTWIN\

D:\, Manuali ESA, SoftWare, Demo, Demo SPUM565 Color V0R, Varie

List Files of Type: *.BMP

Drives: d:

Select image to insert and assign a name.

Click on ▢ OK.

No comment has been added given that the name is sufficiently clear.

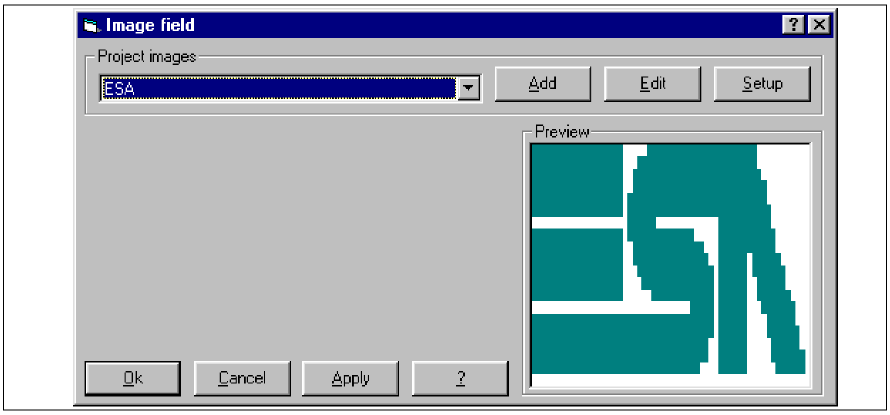
Once the image is inserted in VTWIN, it can be inserted in the page.

Insert all the images in the list (See project offered as example).

Click on , position pointer on the display area and click

Select image to insert in the page.

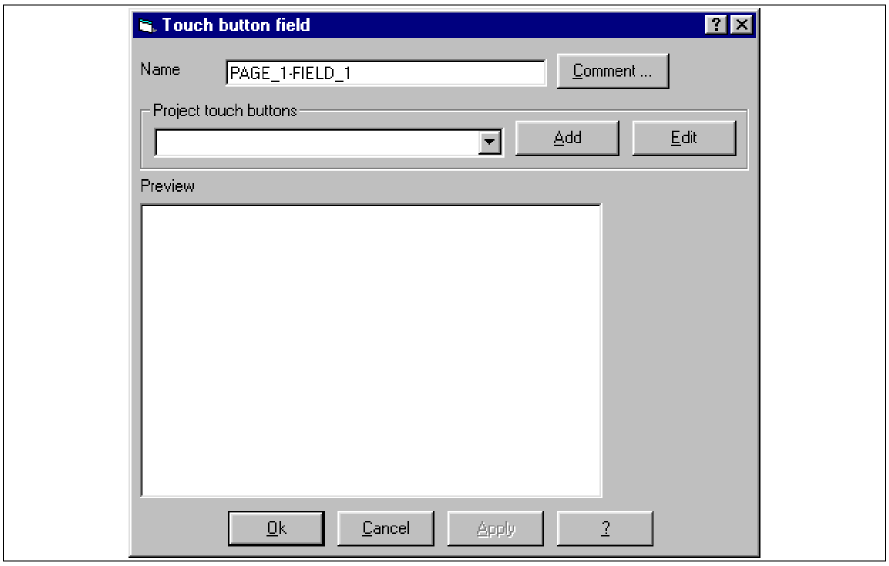
Click on ▢ OK.



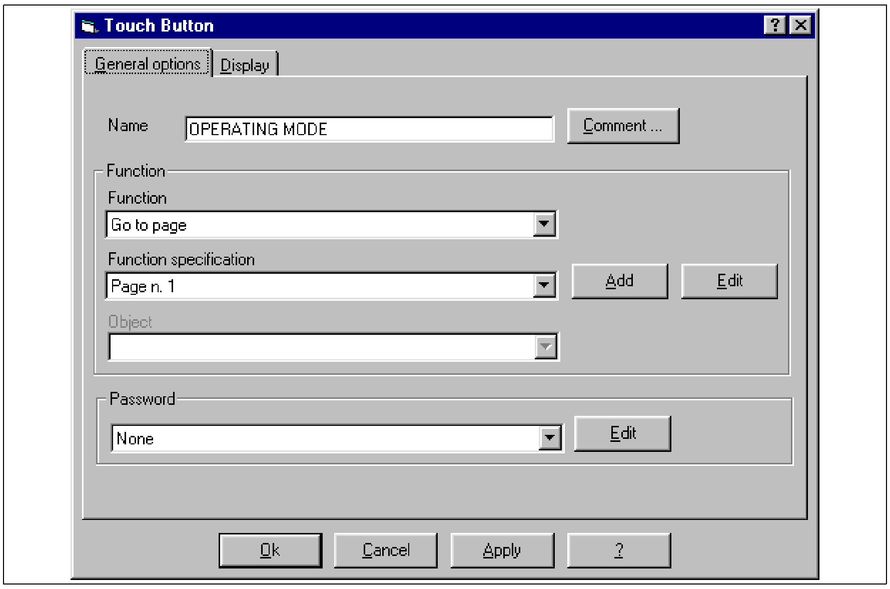
Using the same method insert the image BOTTLE.

As in the case of the multilanguage label, the final positioning will be done later.

After inserting the images, insert the touch buttons (See "Chapter 4 -> Touch Button"). Click on , position the pointer on the display area and click.



Click on the ▢ Add.



Touch Button

General options | Display

Name: OPERATING MODE

Function

Function: Go to page

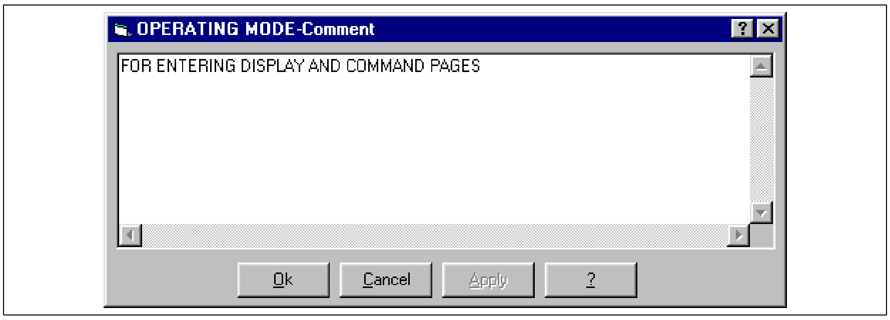
Function specification: Page n. 1

Object:

Password: None

Compile the parameters as illustrated.

Click on ▢ Comment.



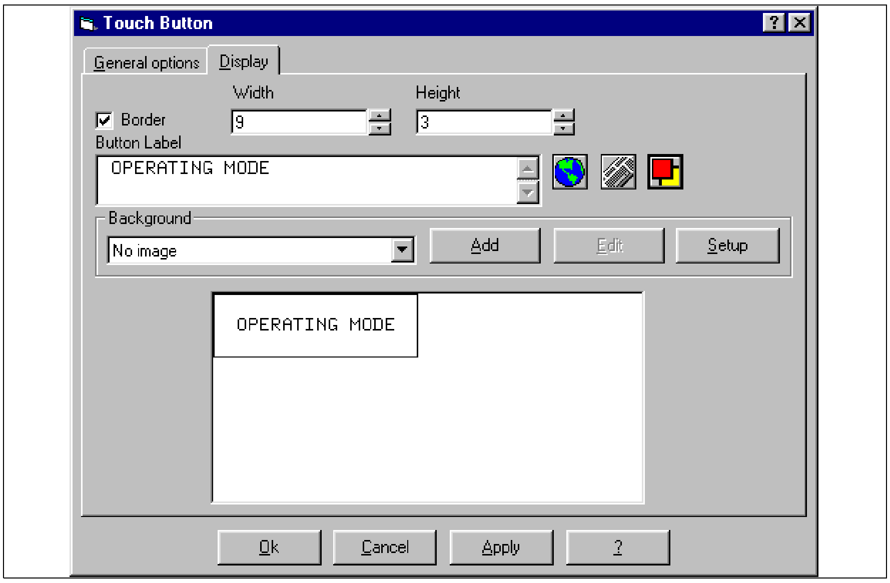
OPERATING MODE-Comment

FOR ENTERING DISPLAY AND COMMAND PAGES

Edit the comment as illustrated.

Confirm with OK returning thereby to the previous mask.

Then browse the 🗂 by clicking on ▢ Display.



Touch Button

General options | Display

[✓] Border

Width: 9

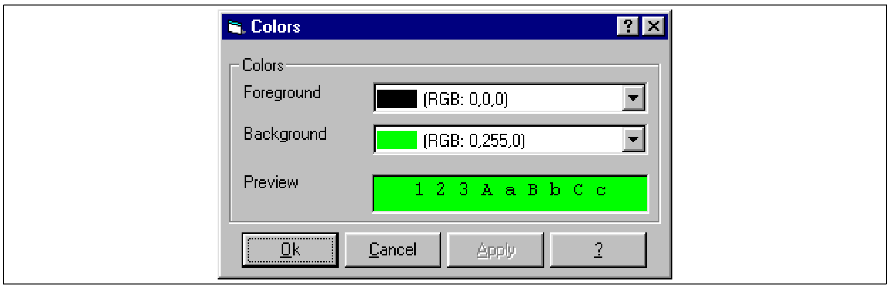
Height: 3

Button Label: OPERATING MODE

Background: No image

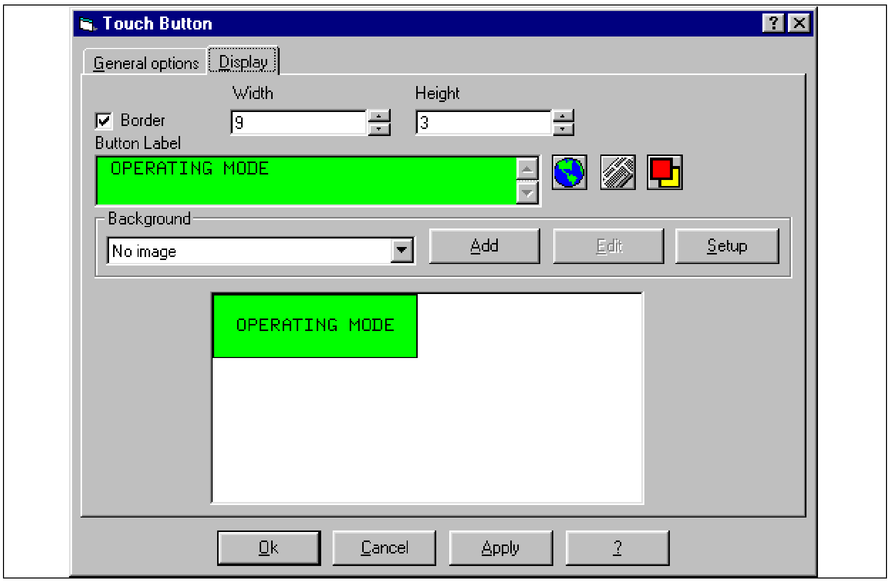
Parameterize as illustrated to create a ▢ with a border.

Click on 🟥.


Colors

Colors
Foreground: (RGB: 0,0,0)
Background: (RGB: 0,255,0)
Preview: 1 2 3 A a B b C c

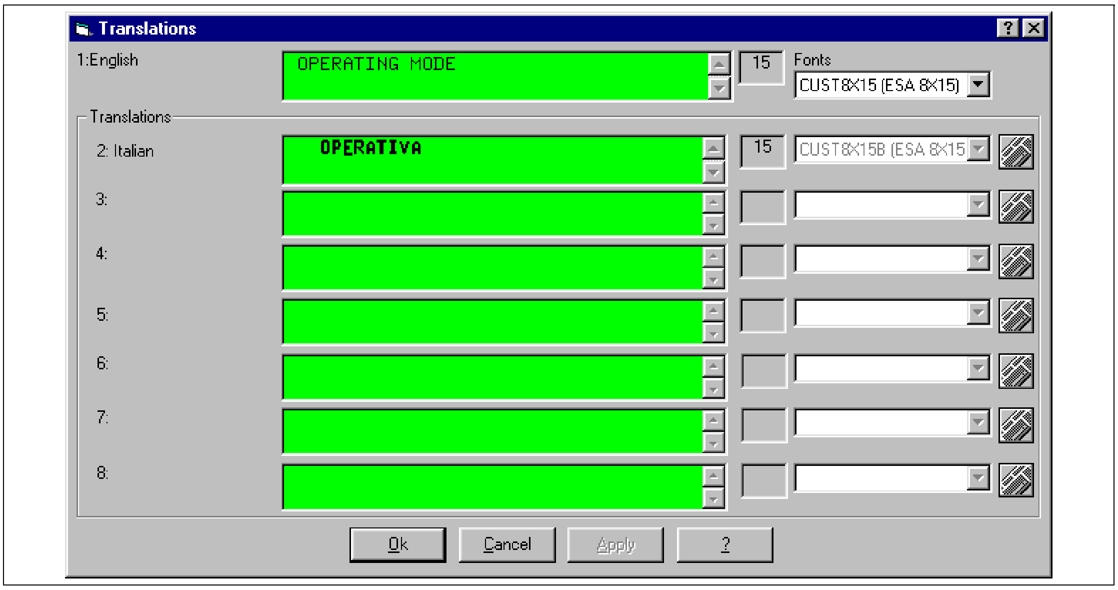
Define the colors as illustrated.
Click on the ☐ Ok.


Touch Button

Width: 9
Height: 3
☑ Border
Button Label: OPERATING MODE
Background: No image

Parameterize as illustrated to create a ☐ with a border.

Click on

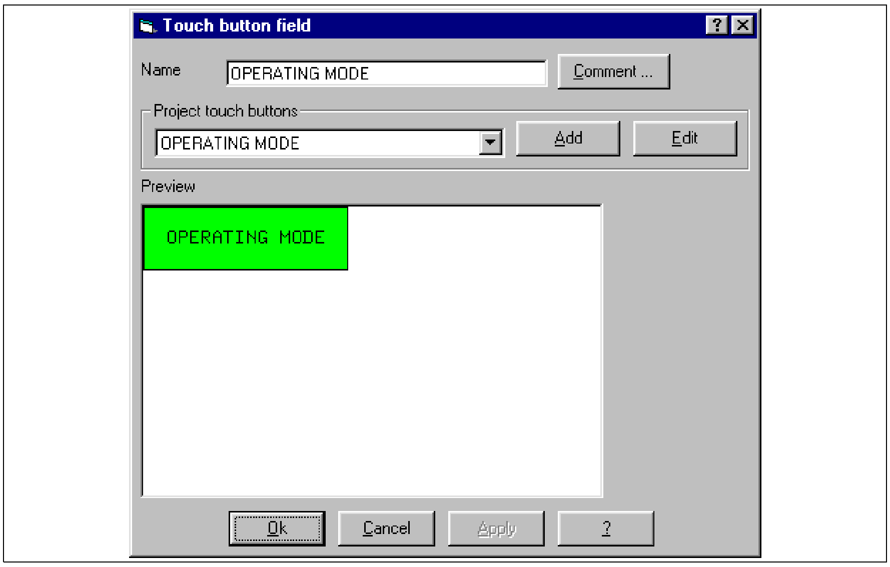

Translations

1:English: OPERATING MODE — 15 — Fonts: CUST8X15 (ESA 8X15)

Translations
2: Italian: OPERATIVA — 15 — CUST8X15B (ESA 8X15
3:
4:
5:
6:
7:
8:

Insert the translation and confirm.

Using the same criteria, insert all the touch buttons (See project offered as example).



Name: OPERATING MODE

Project touch buttons: OPERATING MODE

Preview: OPERATING MODE

Select the □ that you wish to insert; in this case OPERATING MODE.

Click on □ Ok.

Using the same criteria, insert the □ EDIT and the □ SETTINGS.

When all the elements making up the page have been inserted, we can define their positions. When clicking on the element to be moved, it becomes framed by highlighted dotted line. Click on the object, keep the □ pressed down and drag the object.

To select several objects simultaneously click on the objects while holding down the □ Shift.

Continue with the insertion of page 2. Repeat the operations already described.

PAGE 2 -> Operating Page 1

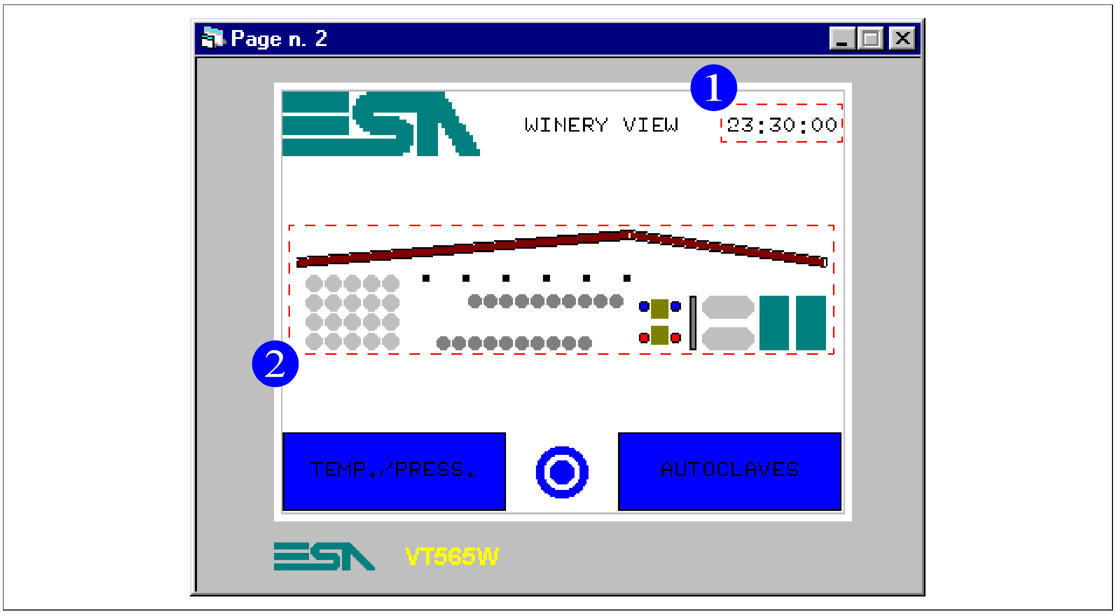


Table 8.2: Elements in page no.2

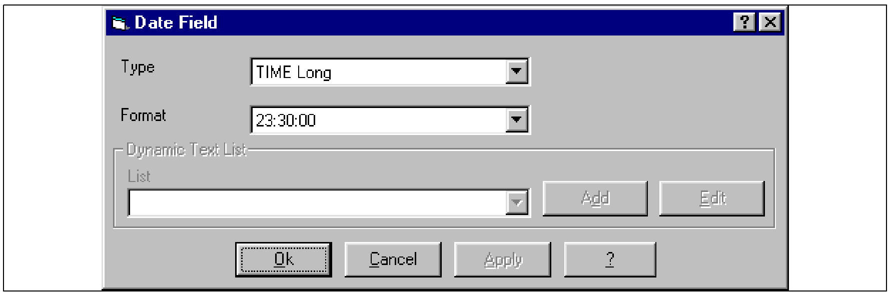
IMAGES	
	ESA
TOUCH BUTTONS	
	TEMPERAT./PRESSURE
	PREVIOUS MENU -> 1
	AUTOCLAVES
DATE/TIME FIELD (1)	
GRAPHIC (2)	

This page is displayed by pressing the touch button “OPERATING MODE” in page 1 (see Page 8-17); it shows a representation of the lay-out of the production area and may be used to call up the page with the temperatures and pressures or the autoclave control page. From this page the user can return to the principal page.

To insert the clock click on and, with the mouse, position the pointer somewhere in the display area and click (See “Chapter 4 -> Hour/Date field“).

Choosing to display the time in a 24 hour format, the seconds are also shown. Compile the parameters as illustrated.

Confirm with OK.



Date Field

Type: TIME Long

Format: 23:30:00

Return to the page being created. Using the method described before, move the clock to its definitive position.
From now on the definitive positioning of the object will be taken for granted and thus not mentioned again.

To insert the graphic click on depending on what it is you want to draw, position the mouse on the display area, click and drag. (See “Chapter 4 -> Line“ and “Chapter 4 -> Ellipse“).

Continue with inserting page 5. Pages 3 and 4 are not created but left as spares. Repeat the operations already described.

PAGE 5 -> Temp./Press. A1-2

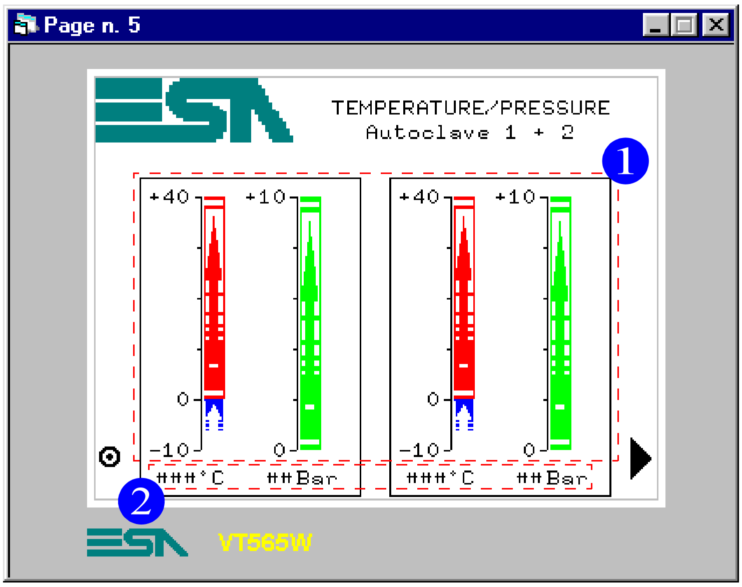


Table 8.3: Elements in page no..5

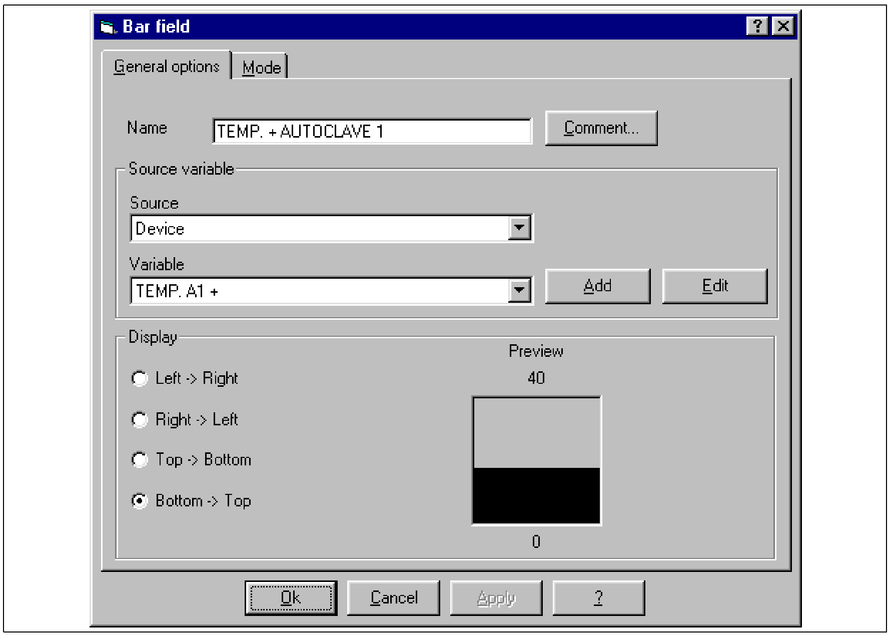
IMAGES	**NUMERICAL DATA FIELD (2)**
ESA	TEMP. A1
TOUCH BUTTONS	PRESS. AUTOCLAVE 1
PREVIOUS MENU -> 2	TEMP. A2
ARROW R -> P6	PRESS. AUTOCLAVE 2
DATE/TIME FIELD	**GRAPHIC**
BAR DATA FIELD (1)	
TEMP. A1+	
TEMP. A1-	
PRESS. AUTOCLAVE 1	
TEMP. A2+	
TEMP. A2-	
PRESS. AUTOCLAVE 2	

This page is displayed by pressing the touch button marked “TEMP./PRESS.” in page 2; it shows the temperature and pressure values for autoclave 1 and 2, shown in numerical and bar data format. From this page the user can turn back to page 2 or to the similarly titled page related to autoclave 3 and 4.

Before proceeding with further insertions, it need to make a brief aside: it will be noted that the bar data for the temperature is composed of two bars, one that runs from 0 to 40°C and the other that runs from 0 to -10°C. This is because we want to see the bar with zero as its origin with the temperature going up or down. Both bars are shown in a bottom to top direction, but the bar from 0 to -10°C is declared as a reverse bar. (See “Chapter 4 -> Bar field“).

We can now proceed with the creation of the bar data from 0 to 40°C. The first thing to do is to determine what the excursion of the bar should be and draw the graduated scale with the help of the graphical functions.

Click now on , move the pointer to the 0 point on the graduated scale just drawn, click and hold down until you reach a value of 40. An arbitrary width of 12 pixels has been given.



Name: TEMP. + AUTOCLAVE 1

Source variable

Source: Device

Variable: TEMP. A1 +

Display

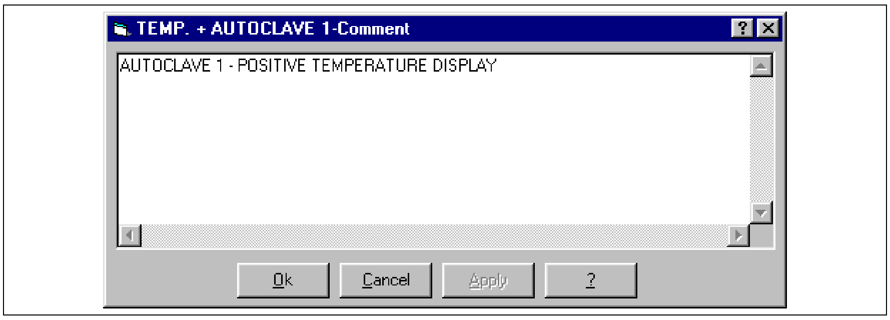
Left -> Right

Right -> Left

Top -> Bottom

[✓ Bottom -> Top]

Assign a name to the data and assign its related comment.

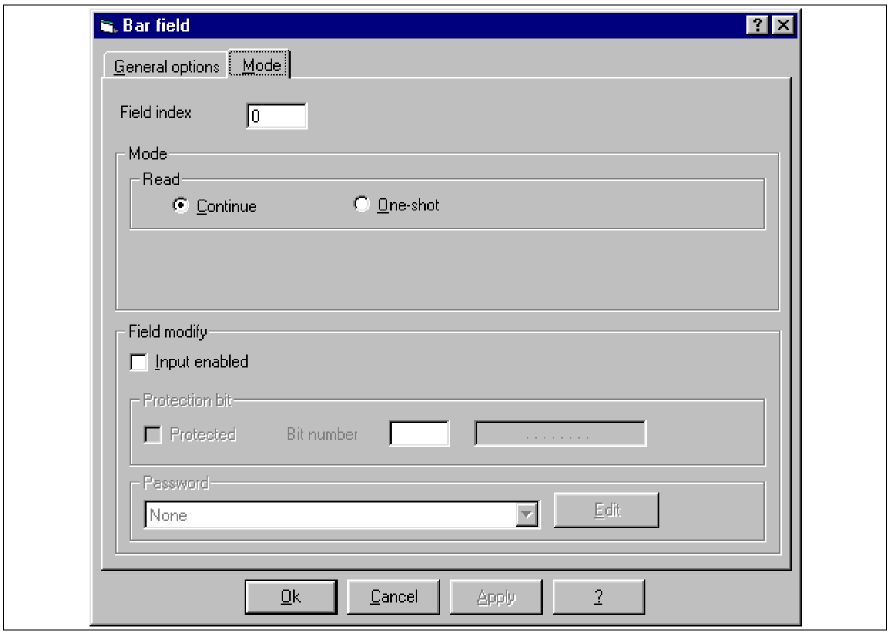


AUTOCLAVE 1 - POSITIVE TEMPERATURE DISPLAY

After assigning the comment, click on OK to confirm and return to the preceding figure, then browse the by clicking on Mode.

The variable is TEMP. A1+ (See project offered as example).
The setting is limited to a range running from 0 to 40°C and not the linear

scale; we therefore make use of the calculation converting from bit values from 0 to 1024 to temperature values.



Field index: 0

Mode
Read: [✓ Continue] One-shot

Field modify
[ ] Input enabled

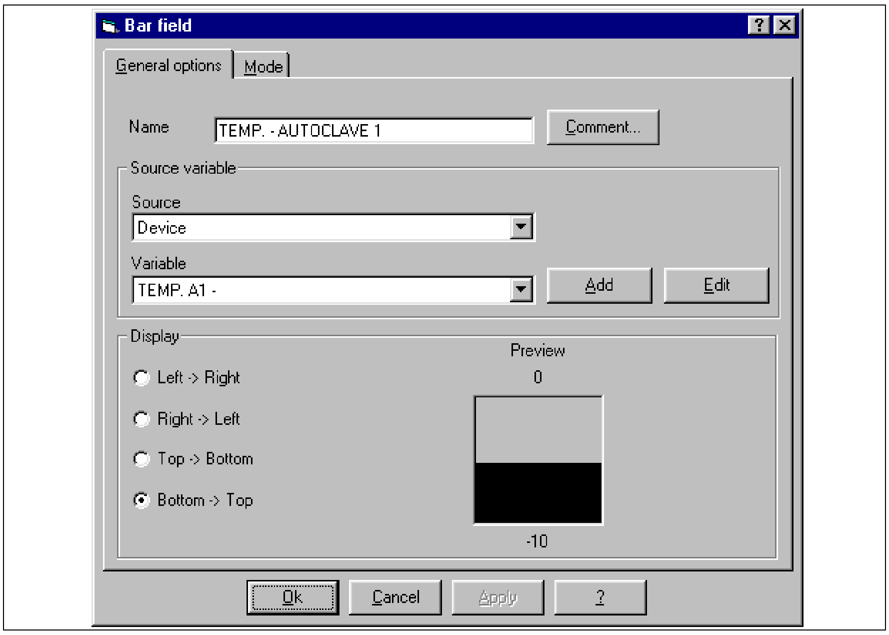
Protection bit
[ ] Protected Bit number

Password: None Edit

Compile the fields as illustrated to obtain a continuous read data which is not settable using the VT.

Click on OK.

Insert the negative bar at this point. Repeat the operations described above.



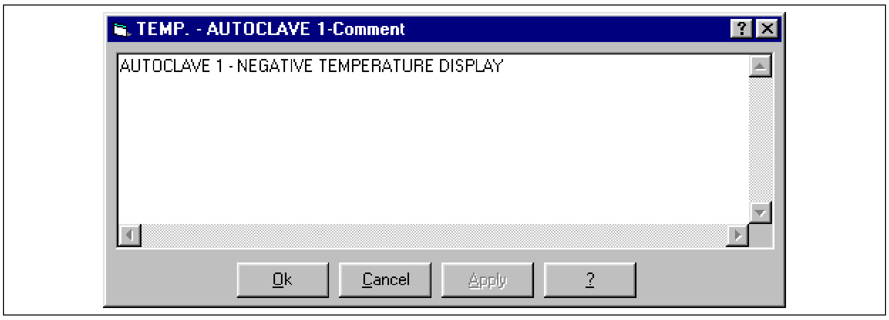
Name: TEMP. - AUTOCLAVE 1

Source variable
Source: Device
Variable: TEMP. A1 -

Display
Left -> Right
Right -> Left
Top -> Bottom
[✓ Bottom -> Top]

Assign a name to the data and the related comment.

After assigning the comment click on OK to confirm and return to the preceding figure; then browse the 🗀 by clicking on Mode.

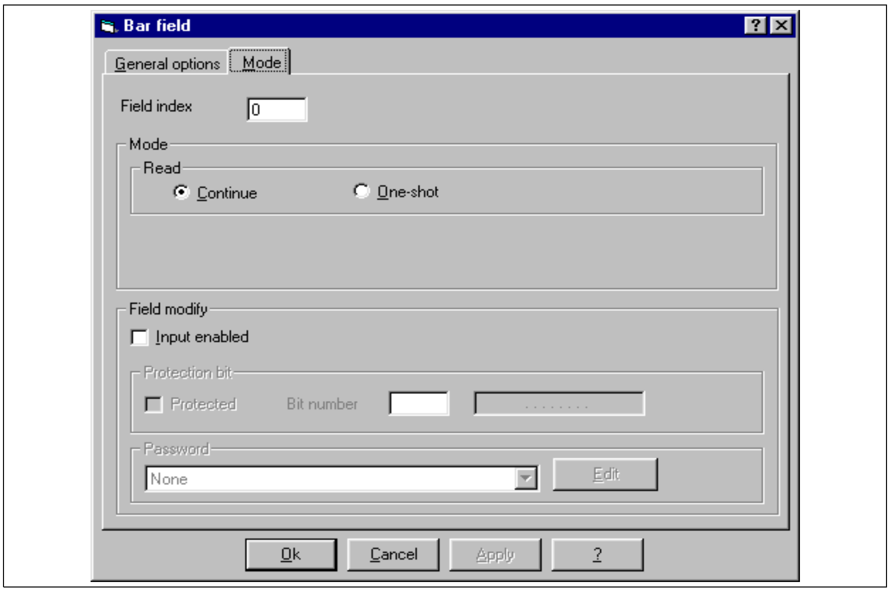

TEMP. - AUTOCLAVE 1-Comment

AUTOCLAVE 1 - NEGATIVE TEMPERATURE DISPLAY

The variable is TEMP. A1- (See project offered as example).
Only the setting is limited to from 0 to -10°C and not the linear scale; we therefore make use of the calculation converting from bit value from 0 to 1024 in temperature value.

Compile the fields as illustrated to obtain a continuous read data which is not settable using the VT.

Click on OK.


Bar field

General options | Mode

Field index: 0

Mode
Read: [✓ Continue] One-shot

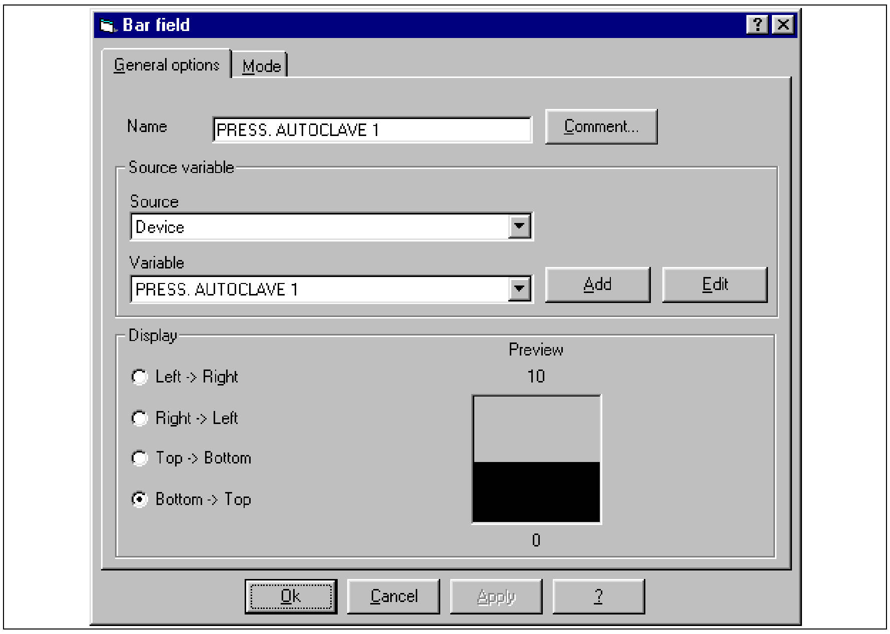
Field modify
☐ Input enabled
Protection bit: ☐ Protected, Bit number
Password: None

Once this has been done, select negative bar data and declare it as reverse.

To do this select the data and click on R.

Now insert the pressure bar data.

The variable is PRESS. AUTOCLAVE 1 (See project offered as example).


Bar field

General options | Mode

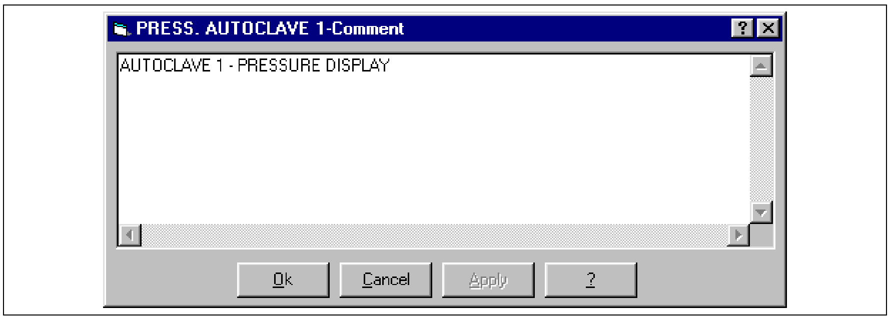
Name: PRESS. AUTOCLAVE 1

Source variable
- Source: Device
- Variable: PRESS. AUTOCLAVE 1

Display
- ( ) Left -> Right
- ( ) Right -> Left
- ( ) Top -> Bottom
- [✓ Bottom -> Top]

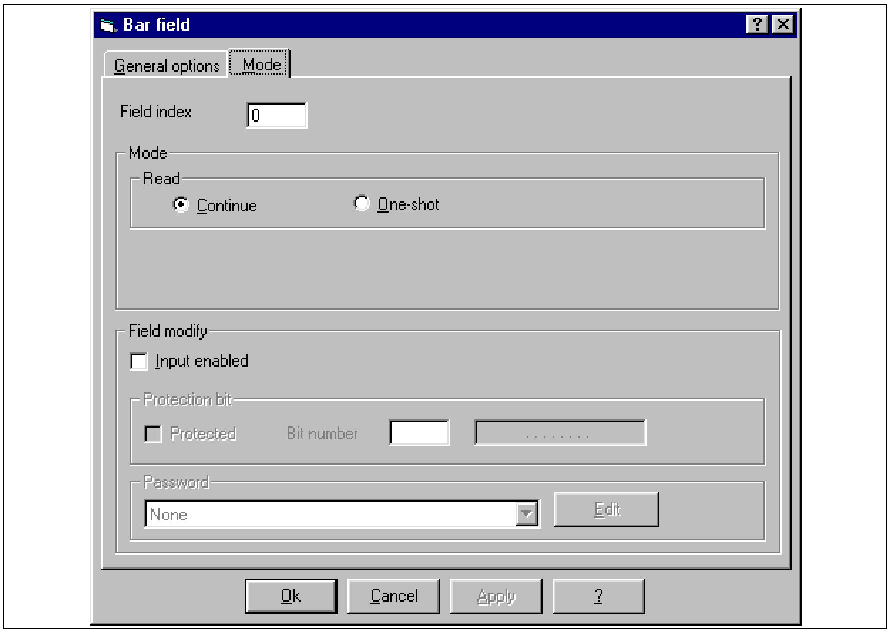
Preview: 10 … 0

Compile as illustrated and assign a comment.


PRESS. AUTOCLAVE 1-Comment

AUTOCLAVE 1 - PRESSURE DISPLAY

After assigning the comment click on OK to confirm and return to the preceding figure; then browse the 🗀 by clicking on Mode.



Field index: 0

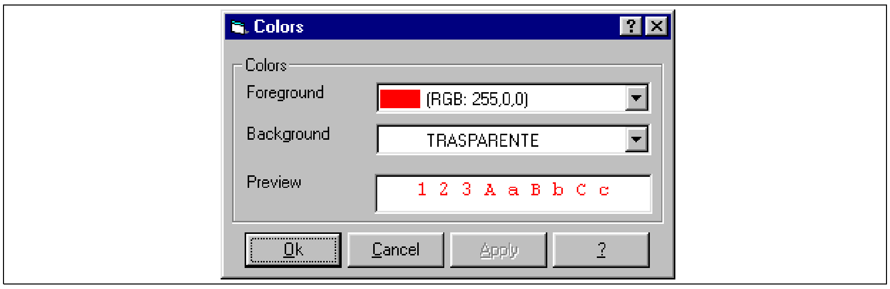
Mode
Read: (● Continue) ○ One-shot

Field modify
☐ Input enabled
Protection bit
☐ Protected Bit number
Password: None

Compile as illustrated.

Click on OK

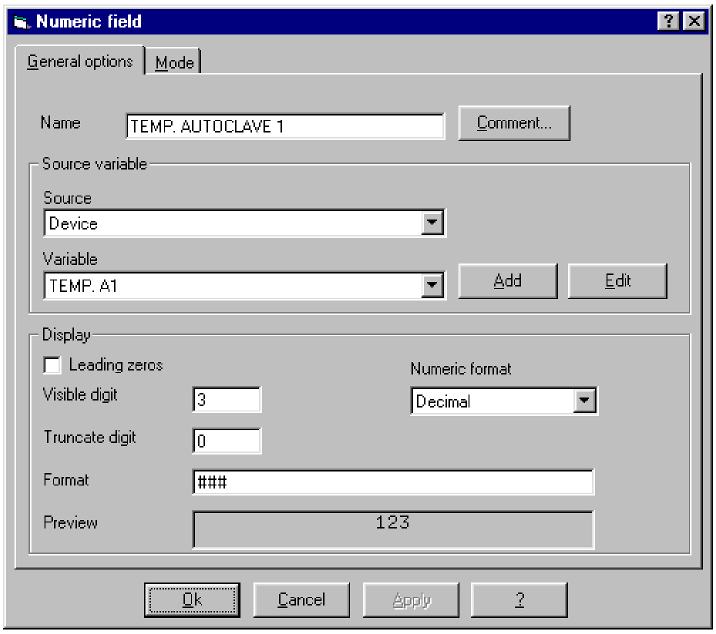
Once the data has been entered the colors for displaying them must be defined. Select the relevant data and click on 🟥. The following mask appears.



Colors
Foreground: 🟥 (RGB: 255,0,0)
Background: TRASPARENTE
Preview: 1 2 3 A a B b C c

Set the colors as illustrated.
Click on the ☐ Ok.

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


Numeric field

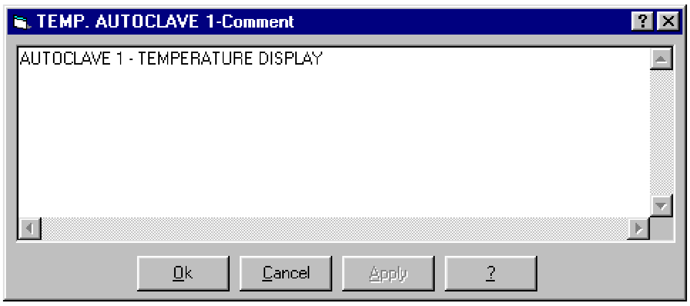
General options | Mode

Name: TEMP. AUTOCLAVE 1

Source variable
Source: Device
Variable: TEMP. A1

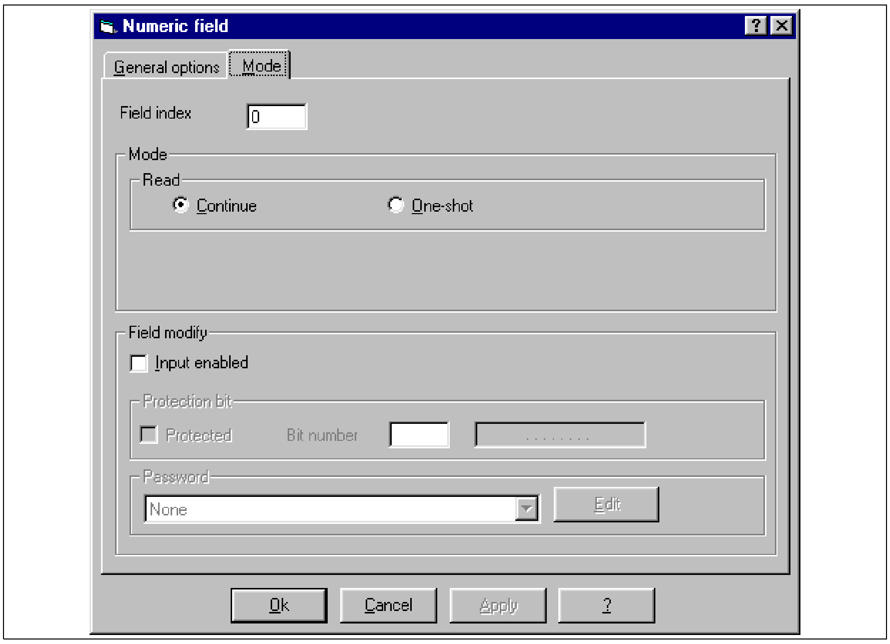
Display
Leading zeros: [ ]
Visible digit: 3
Truncate digit: 0
Format: ###
Preview: 123
Numeric format: Decimal

Compile as illustrated and assign the comment.


TEMP. AUTOCLAVE 1-Comment

AUTOCLAVE 1 - TEMPERATURE DISPLAY

After assigning the comment click on OK to confirm and return to the preceding figure; then browse the 🗀 by clicking on Mode.



Field index: 0

Mode

Read: [✓ Continue] One-shot

Field modify

Input enabled: ☐

Protection bit: Protected ☐ Bit number

Password: None

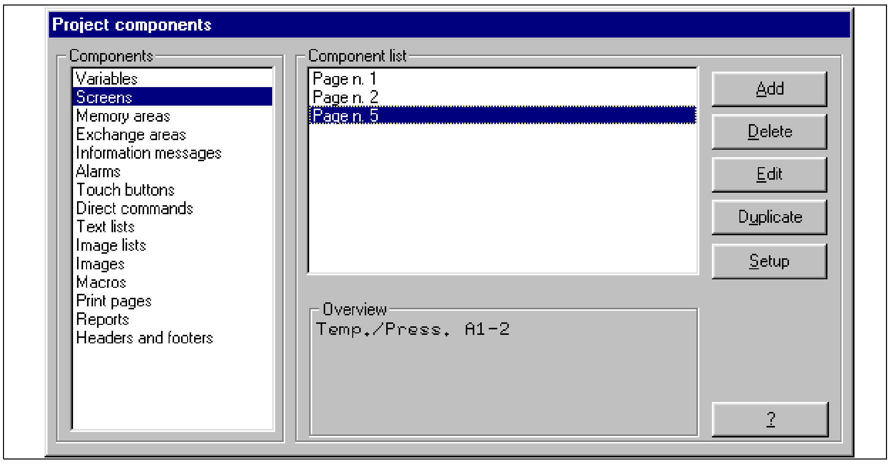
Compile the fields as illustrated to obtain a continuous read data, which is not settable using the VT.

Click on OK

Assign a Multilanguage label comment for °C; repeat this procedure for the pressure. Everything done in relation to autoclave 1 must be done for autoclave 2 as well.

Insert page 6. Two approaches are possible: repeating the operations described above, or, given the overwhelming similarity of the two pages, using the duplication function. This way a page with the same graphical content can be generated, taking advantage of the positions already assigned as well as the background graphic.

To continue duplicating go to this mask



Components: Variables, [✓ Screens], Memory areas, Exchange areas, Information messages, Alarms, Touch buttons, Direct commands, Text lists, Image lists, Images, Macros, Print pages, Reports, Headers and footers

Component list: Page n. 1, Page n. 2, [✓ Page n. 5]

Overview: Temp./Press. A1-2

Select as illustrated and click on ▢ Duplicate.

Thus the page is duplicated, but it is necessary to compile the comments and the page number desired. (When you duplicate the old comment is maintained and the page number is assigned by the system as the first free number, in this case the system assigns the number 3.)

PAGE 6 -> Temp./Press. A3-4

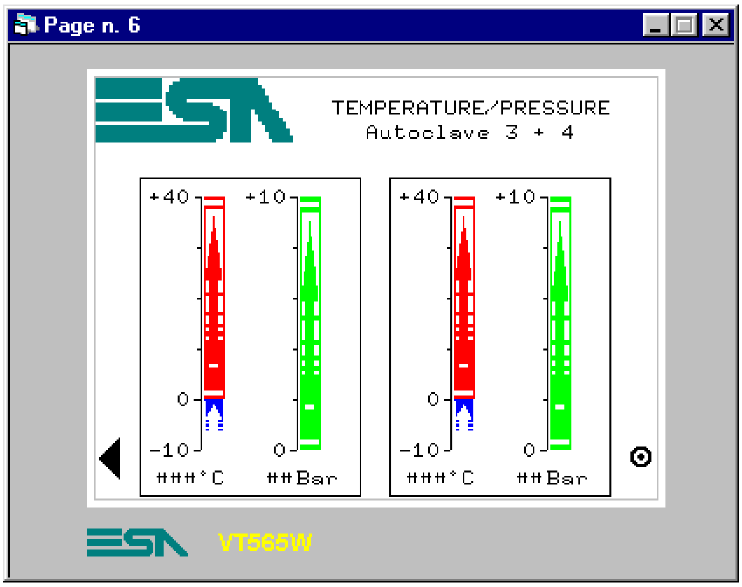


Table 8.4: Elements in page no. 6

IMAGES	NUMERICAL DATA FIELD
ESA	TEMP. A3
TOUCH BUTTONS	PRESS. AUTOCLAVE 3
PREVIOUS MENU -> 2	TEMP. A4
ARROW L -> P5	PRESS. AUTOCLAVE 4
DATE/TIME FIELD	**GRAPHIC**
BAR FIELD	
TEMP. A3+	
TEMP. A3-	
PRESS. AUTOCLAVE 3	
TEMP. A4+	
TEMP. A4-	
PRESS. AUTOCLAVE 4	

This page is displayed by pressing the touch button “>” in page 5; it shows the temperature and pressure values for autoclaves 3 and 4, displayed in bar and numerical format. From this page it is possible to return to page 2 or to the similarly titled page for autoclaves 1 and 2.

PAGE 7 -> Autoclaves

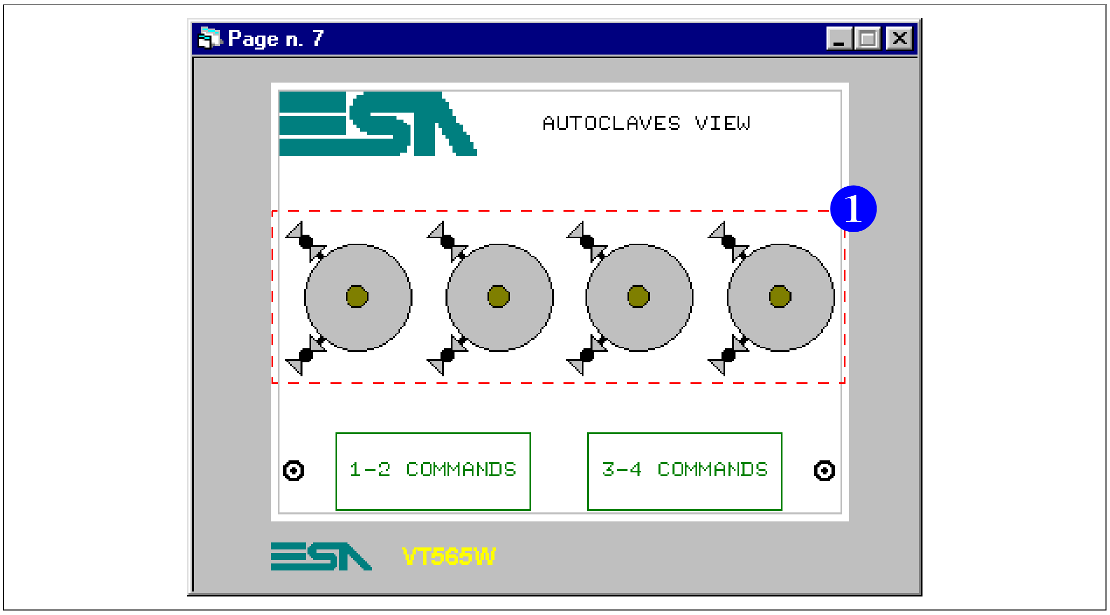


Table 8.5: Elements in page no. 7

IMAGES	
	ESA
TOUCH BUTTONS	
	PREVIOUS MENU -> 2
	COMMANDS 1-2
	COMMANDS 3-4
DATE/TIME FIELDS	
SYMBOLIC FIELD (1)	
	AUTOCLAVE E/F

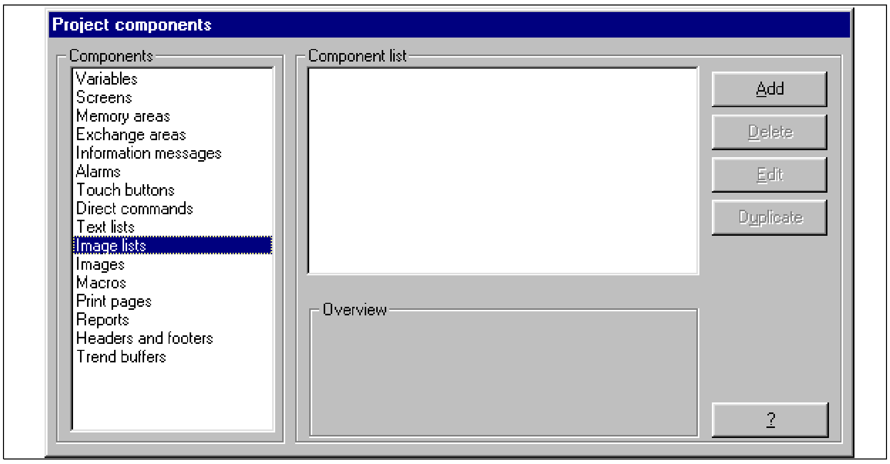
This page is displayed by pressing the touch button “AUTOCLAVES” in page 2; it gives an overview of the plant’s autoclaves and the operational status, whether or not it is automatic. In addition by pressing the touch buttons “1-2 COMMANDS” or “3-4 COMMANDS” you can go to the associated page, from which it will be possible to control the respective autoclaves. From this page it is possible to return to page 2.

Proceed inserting dynamic symbols (See “Chapter 4 -> Symbolic field“).

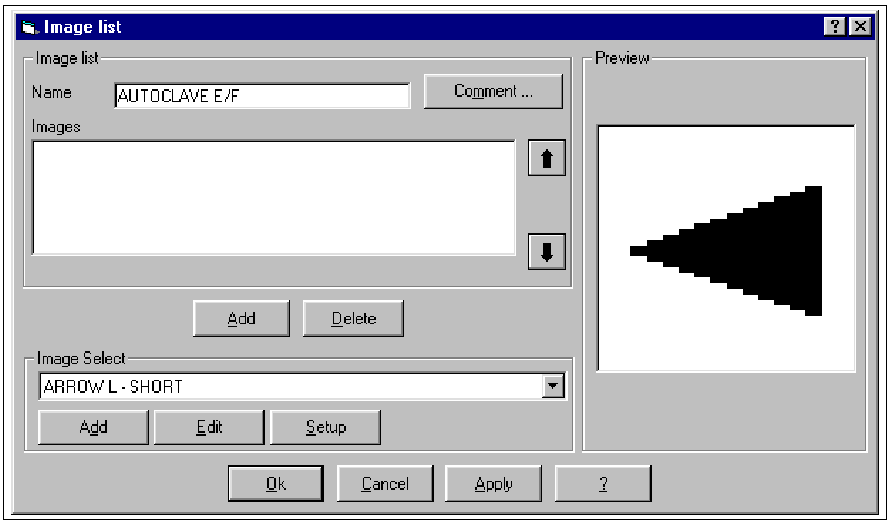
To do this you need to have created images and the variables to be assigned.

We take for granted that these operations have been done, given that the relevant procedure has already been described.

Go to this mask



Select as illustrated and click on ▢ Add.



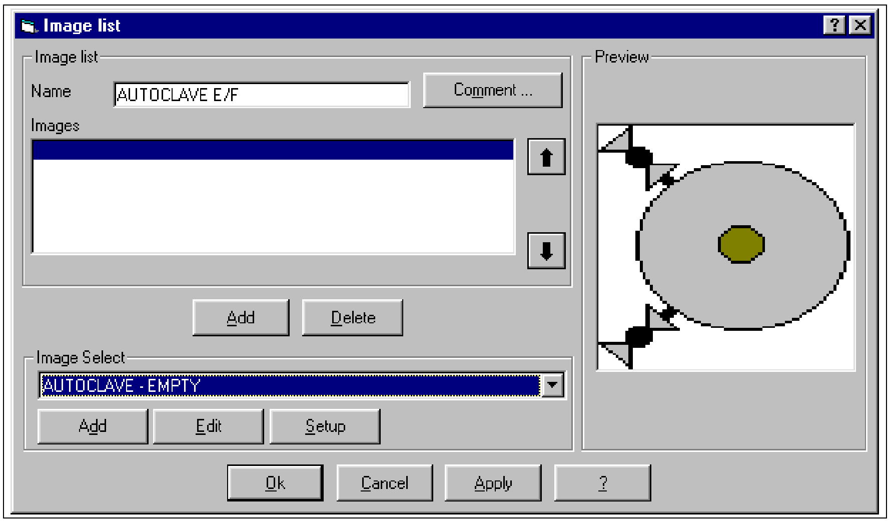
Name: AUTOCLAVE E/F

Image Select: ARROW L - SHORT

Assign a name to the list.

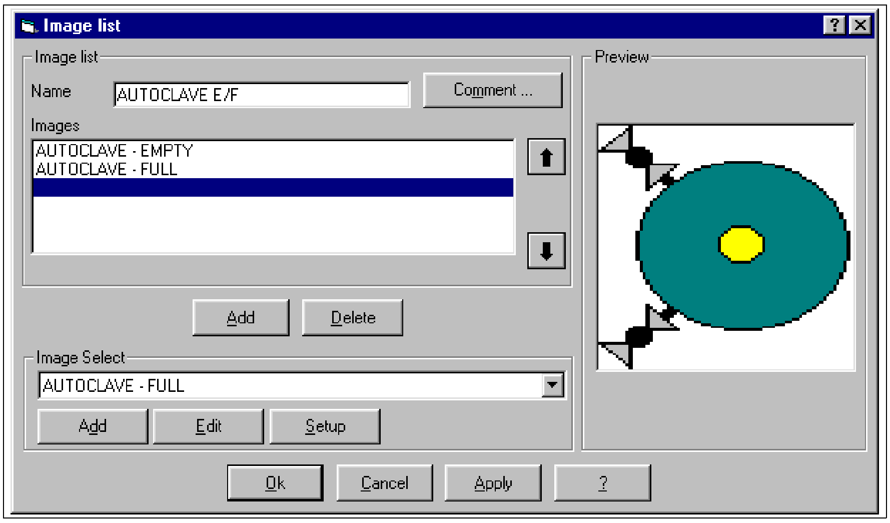
Select from among the images available. The first to insert is AUTOCLAVE EMPTY.

Click on ▢ Add under the Images window.



Select the second image in the list AUTOCLAVE FULL and click on Add.

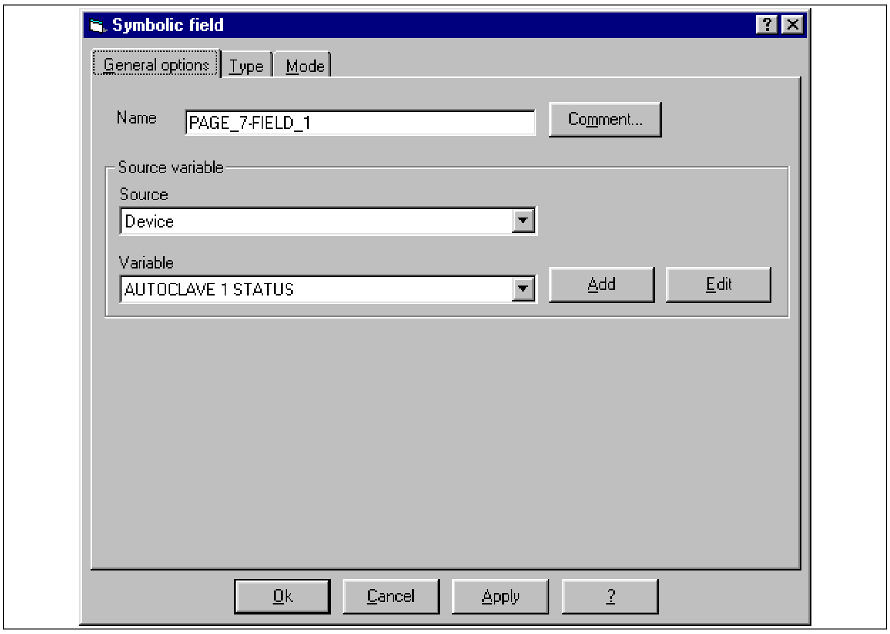
Click on OK to accept the list.



The order of insertion is not random, but relates to the display desired: the first image is that displayed when the assigned bit is set at 0.

Insert all the lists of images (See project offered as example).

Turn to the page and click on , move the pointer in the display area and click.

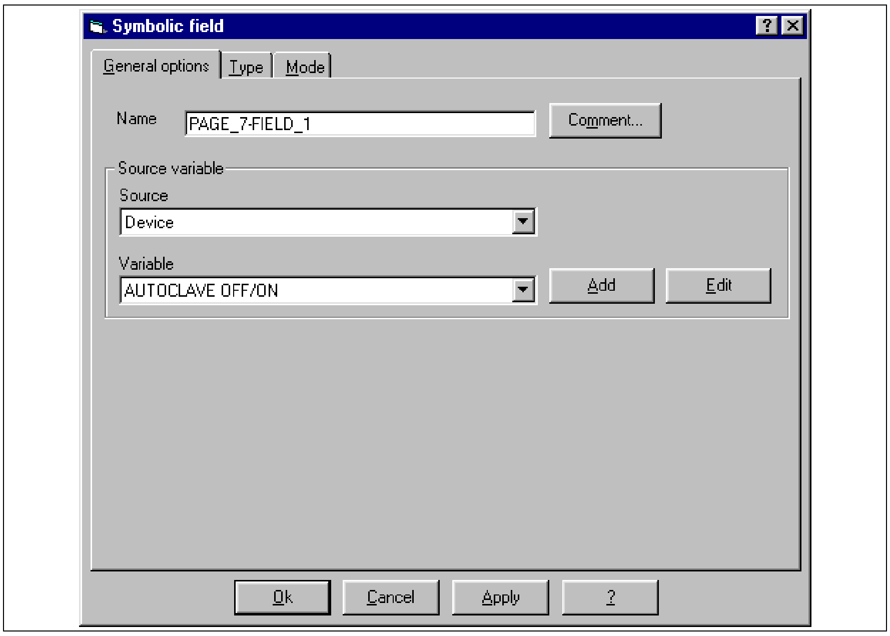


Name: PAGE_7-FIELD_1

Source variable

Source: Device

Variable: AUTOCLAVE 1 STATUS



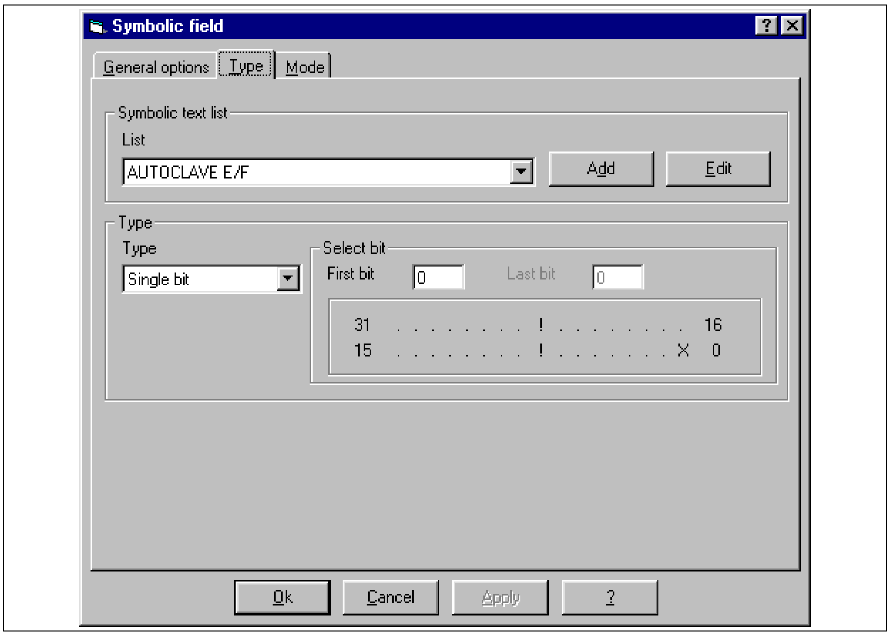
Name: PAGE_7-FIELD_1

Source variable

Source: Device

Variable: AUTOCLAVE OFF/ON

Compile as illustrated, then select the 🗀 Type.

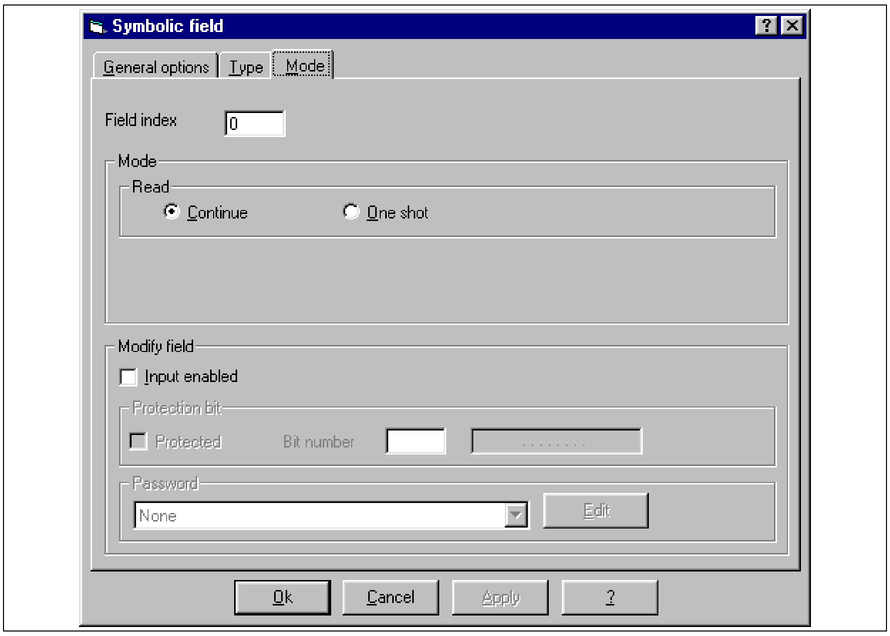


Symbolic text list
List: AUTOCLAVE E/F

Type
Type: Single bit
Select bit
First bit: 0
Last bit: 0

```
31  . . . . . . . . ! . . . . . . . .  16
15  . . . . . . . . ! . . . . . . . X   0
```

Compile as illustrated, then select the 🗀 Mode.



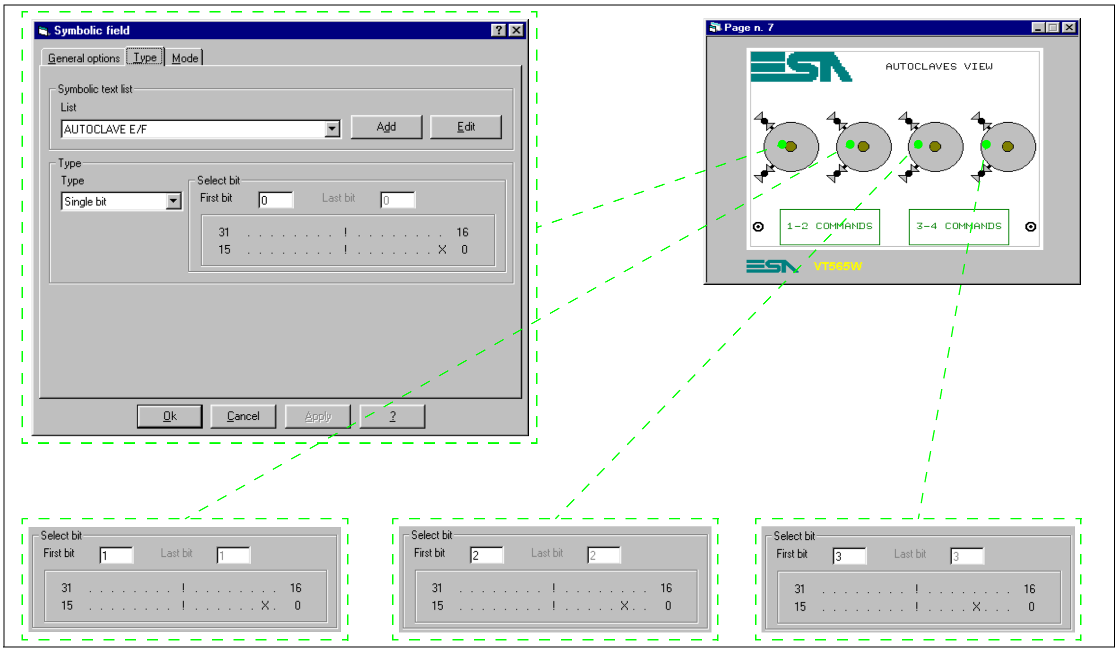
Field index: 0

Mode
Read
● Continue ○ One shot

Modify field
☐ Input enabled
Protection bit
☐ Protected Bit number
Password
None

Compile as illustrated.

Click on OK.



Symbolic field

General options | Type | Mode

Symbolic text list

List: AUTOCLAVE E/F

Type: Single bit

Select bit — First bit: 0 — Last bit: 0

```
31  . . . . . . . . ! . . . . . . . .  16
15  . . . . . . . . ! . . . . . . . X   0
```

Ok | Cancel | Apply | ?

Page n. 7 — AUTOCLAVES VIEW — 1-2 COMMANDS — 3-4 COMMANDS — VT565W

Select bit — First bit: 1 — Last bit: 1

```
31  . . . . . . . . ! . . . . . . . .  16
15  . . . . . . . . ! . . . . . . X .   0
```

Select bit — First bit: 2 — Last bit: 2

```
31  . . . . . . . . ! . . . . . . . .  16
15  . . . . . . . . ! . . . . . X . .   0
```

Select bit — First bit: 3 — Last bit: 3

```
31  . . . . . . . . ! . . . . . . . .  16
15  . . . . . . . . ! . . . . X . . .   0
```

Repeat the operation for all the lists of images and parametrize them as illustrated.

Complete the page with the elements that are lacking.

PAGE 8 -> Autoclave 1 commands

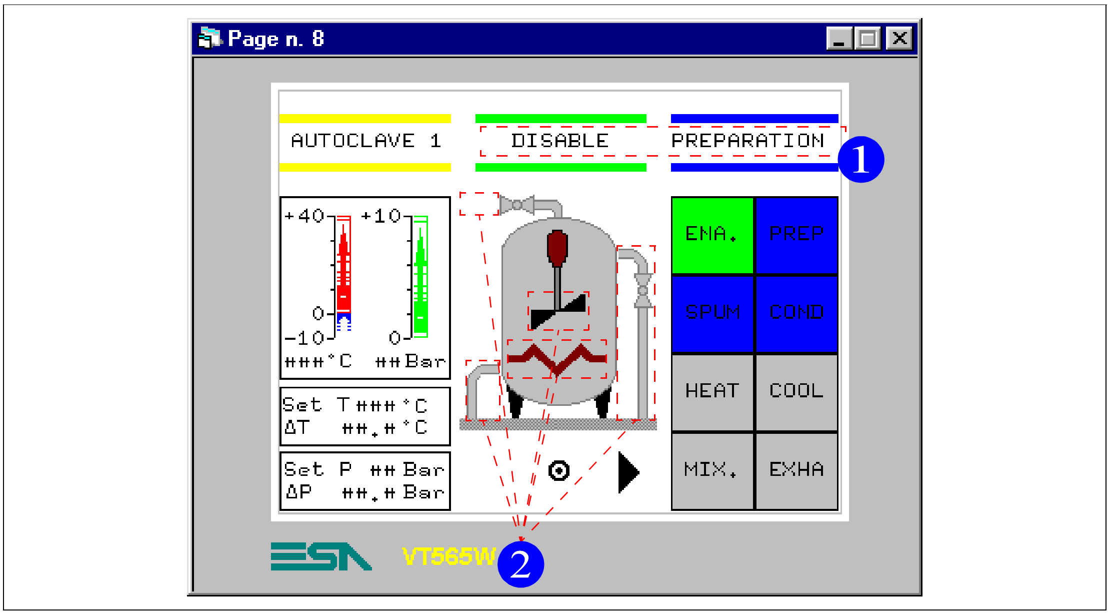


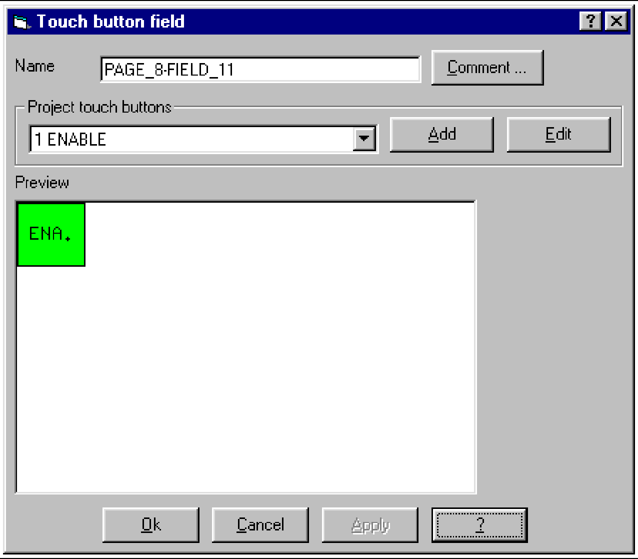
Table 8.6: Elements in page no. 8

IMAGES	MIXER
AUTOCLAVE - BACKGROUND	HEATING
TOUCH BUTTONS	**DYNAMIC FIELD (1)**
1 ENABLE	AUTOCLAVE E/D
1 PREPARATION	AUTOCLAVE STATUS
1 SPUMANTIZZA	**BAR DATA FIELD**
1 CONDITIONING	TEMP. A1+
1 HEATING	TEMP. A1-
1 COOLING	PRESS. AUTOCLAVE 1
1 MIXER	**NUMERICAL DATA FIED**
1 EXHAUST	TEMP. A1
PREVIOUS MENU -> 7	PRESS. AUTOCLAVE 1
ARROW R -> P9	SET TEMP. A1 PRE
SYMBOLIC FIELD (2)	SET dT TEMP. A1 PRE
EXHAUST	SET PRES. A1 PRE
COOL LEFT	SET dP PRES. A1 PRE
COOL RIGHT	

This page is displayed by pressing the touch button marked “1-2 COMMANDS” in page 7; using moving images, it shows the status of the autoclave.

Start with arranging the touch buttons. Once they have been arranged on the page edit the button functions, given that at the beginning they were all inserted as “Go to page”. Let us start with the touch button “INS”.

Double click on the button.

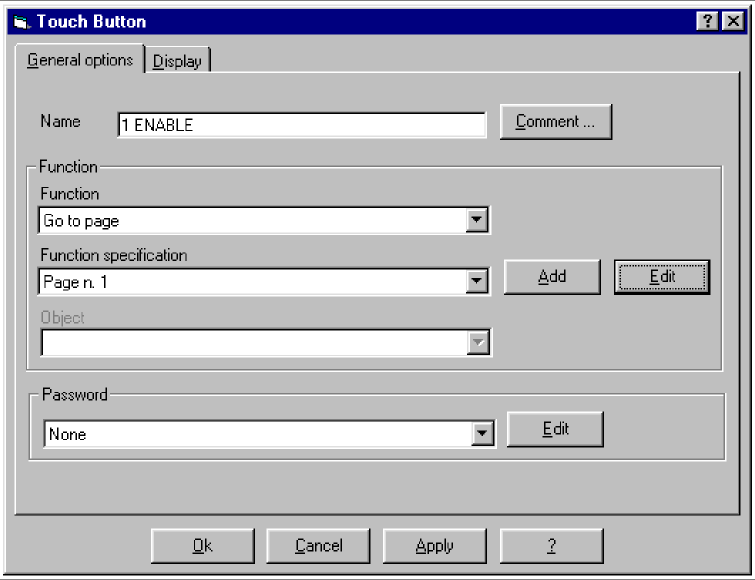

Touch button field

Name: PAGE_8-FIELD_11

Project touch buttons: 1 ENABLE

Preview: ENA.

Click on ☐ Edit.


Touch Button

General options | Display

Name: 1 ENABLE

Function

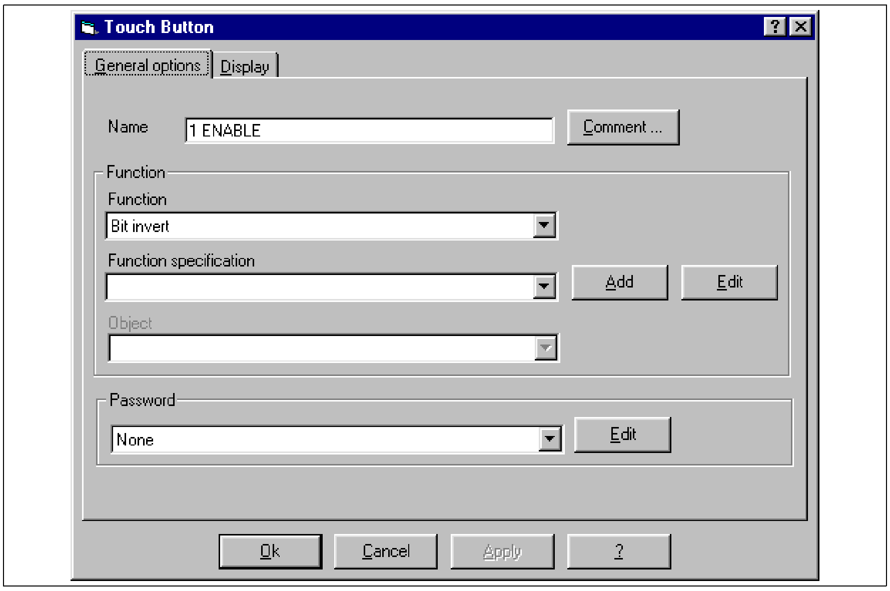
Function: Go to page

Function specification: Page n. 1

Object:

Password: None

Choose the 🕮 from the list of functions as illustrated.



Touch Button

General options | Display

Name: 1 ENABLE

Function

Function: Bit invert

Function specification:

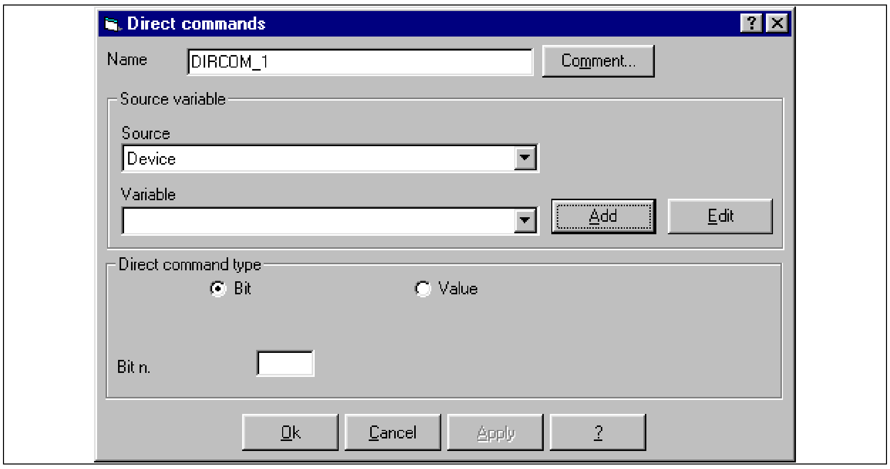
Object:

Password: None

Insert the definition of the function.

Click on Add.

See “Chapter 4 -> Direct Commands“.



Direct commands

Name: DIRCOM_1

Source variable

Source: Device

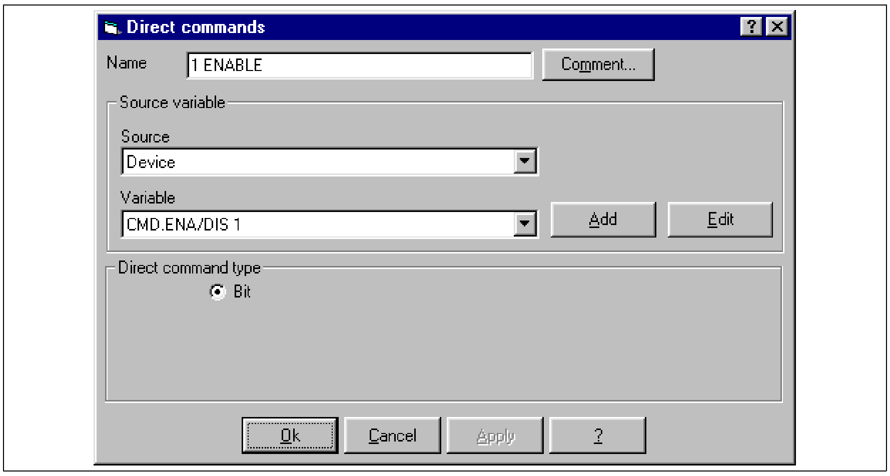
Variable:

Direct command type: [✓ Bit] Value

Bit n.:

Assign the name and the variable as illustrated.

Click on OK.


Direct commands

Name: 1 ENABLE

Source variable

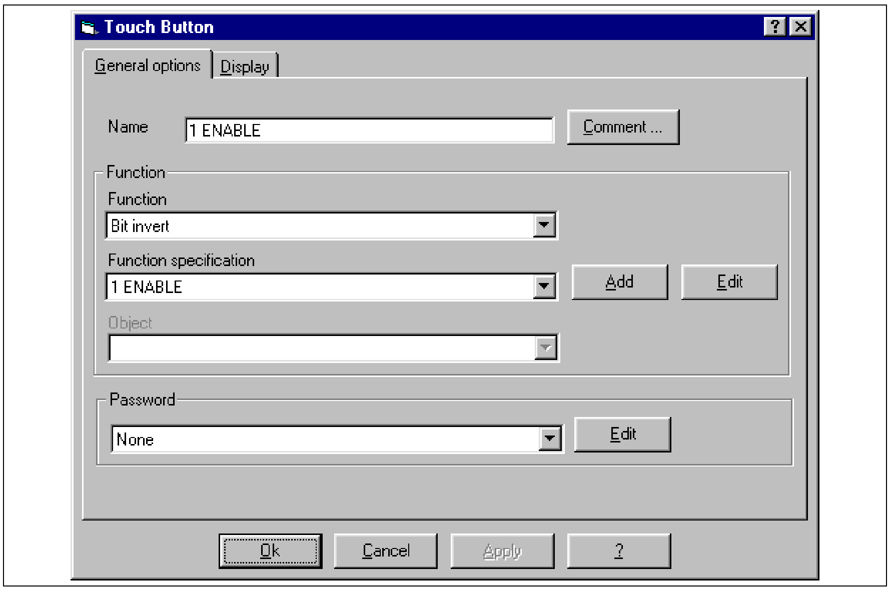
Source: Device

Variable: CMD.ENA/DIS 1

Direct command type: (●) Bit

Insert all the Direct commands (See project offered as example).

Choose the 📖 from the Function specification list as illustrated and confirm with OK.


Touch Button

General options | Display

Name: 1 ENABLE

Function

Function: Bit invert

Function specification: 1 ENABLE

Object:

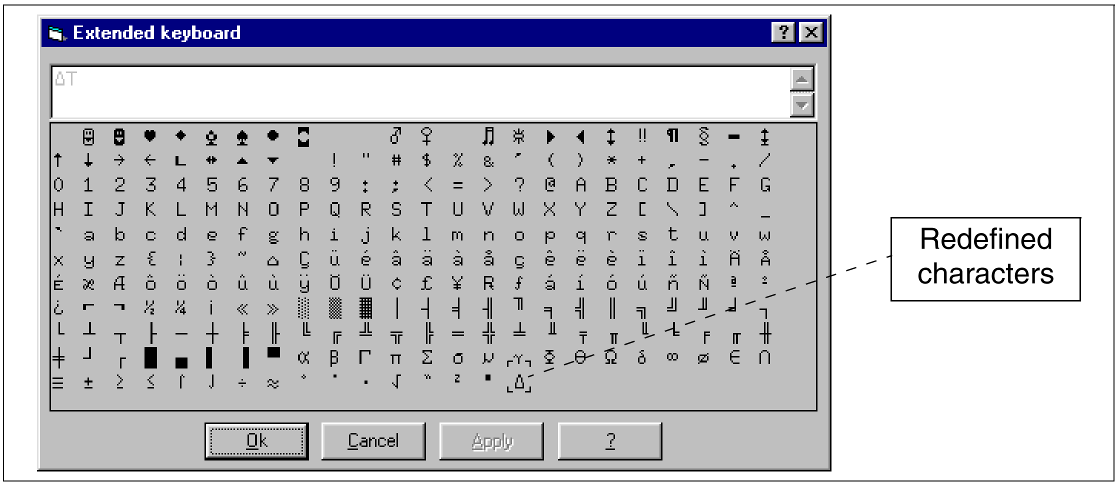
Password: None

Using the same procedure edit all the buttons in the page.

Insert the bar and numerical data as already shown. Note that the symbol 🛍 is not a part of the font chosen when building the project, it must therefore be constructed (See “Chapter 12 -> Defining the fonts“).

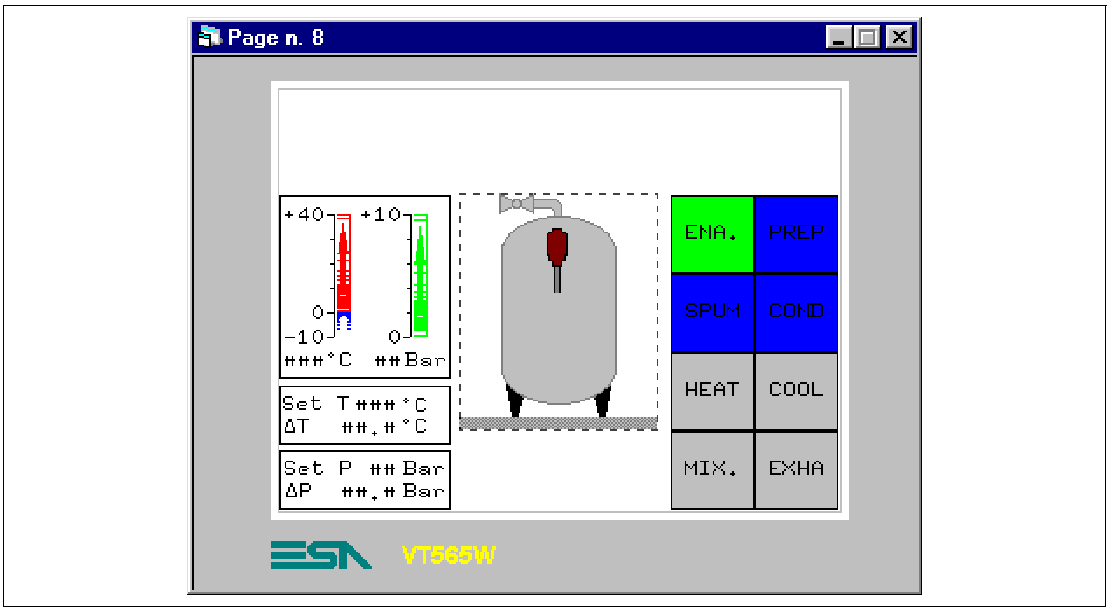
Once created, the fonts CUST8X15 and CUST8X15B must substitute those originally selected (See Page 8-7).

Insert a multilanguage label then press the ▢ F12.

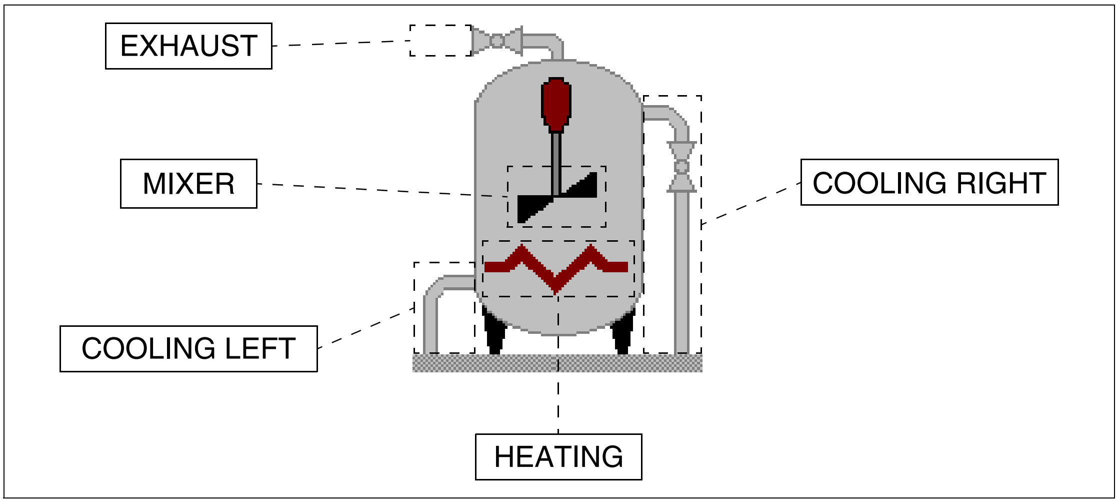


Edit the label and confirm with Ok.

Now insert the background image AUTOCLAVE BACKGROUND.

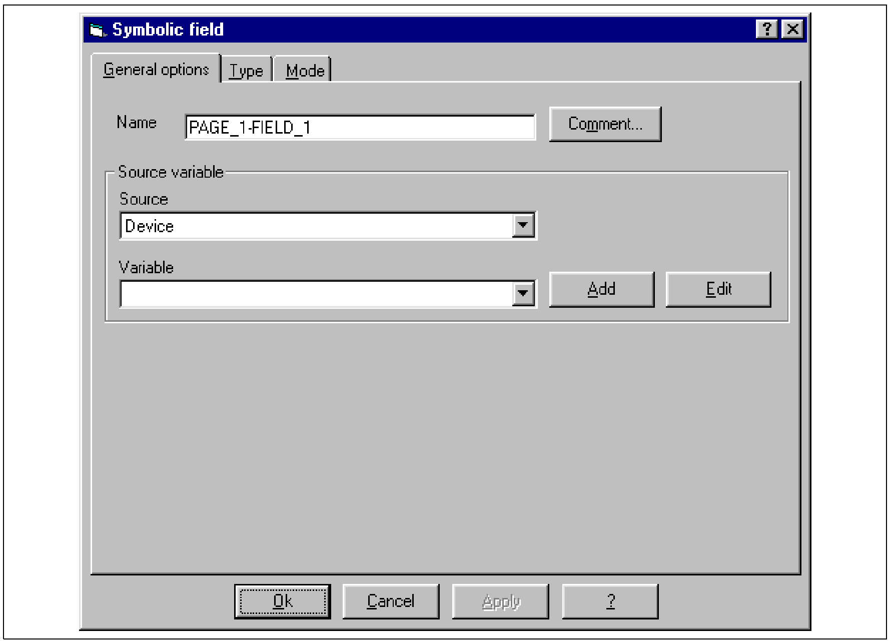


At this point we can complete image adding the parts lacking. These elements are not simple images but rather lists of dynamic images. Lists of images are used because these produce the impression of movement. To have the effect of movement a series of images in more or less rapid succession needs to be displayed. This flow of images must be managed by the connected device.



Insert the dynamic texts (See “Chapter 4 -> Dynamic Text Field“).

Click on

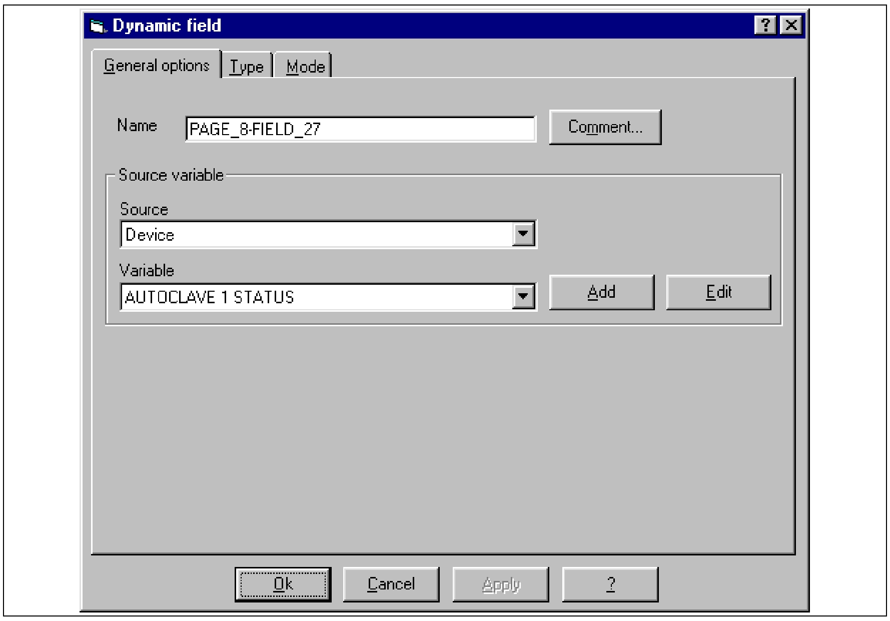


Name: PAGE_1-FIELD_1

Source variable

Source: Device

Variable:

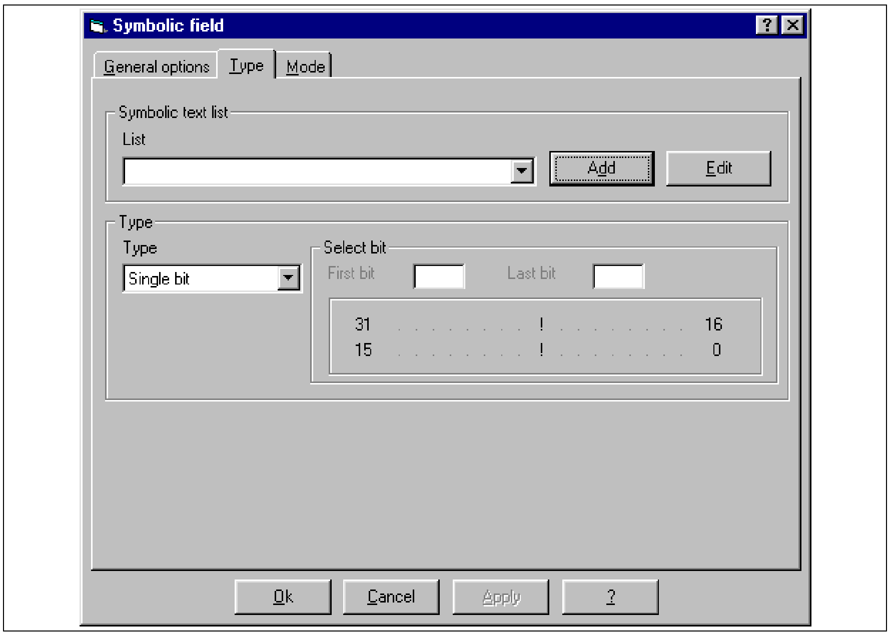

Dynamic field

General options | Type | Mode

Name: PAGE_8-FIELD_27

Source variable
Source: Device
Variable: AUTOCLAVE 1 STATUS

Assign the variable related to the dynamic text, then browse the 🗀 by clicking on Type.


Symbolic field

General options | Type | Mode

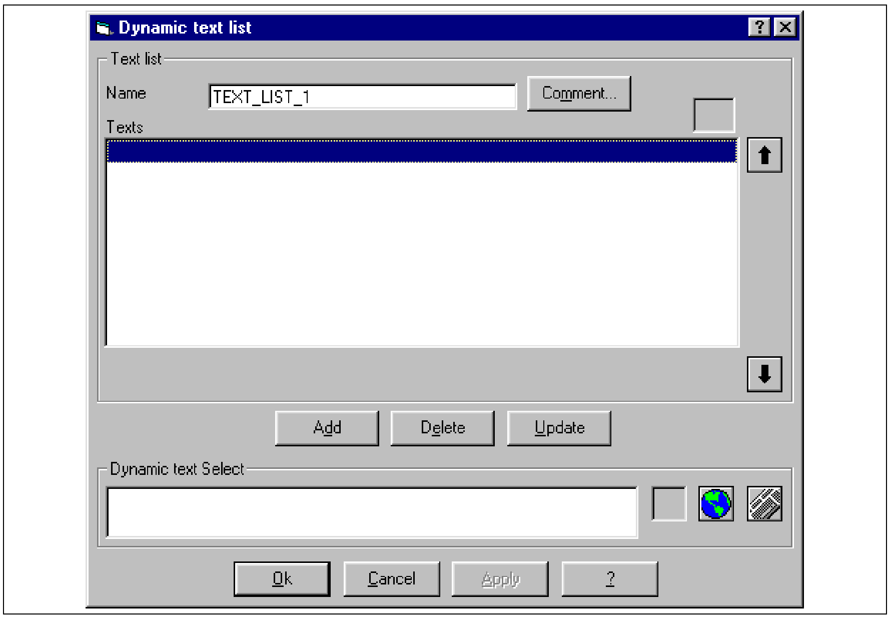
Symbolic text list
List:

Type
Type: Single bit
Select bit
First bit:
Last bit:

```
31 . . . . . . . . ! . . . . . . . . 16
15 . . . . . . . . ! . . . . . . . . 0
```

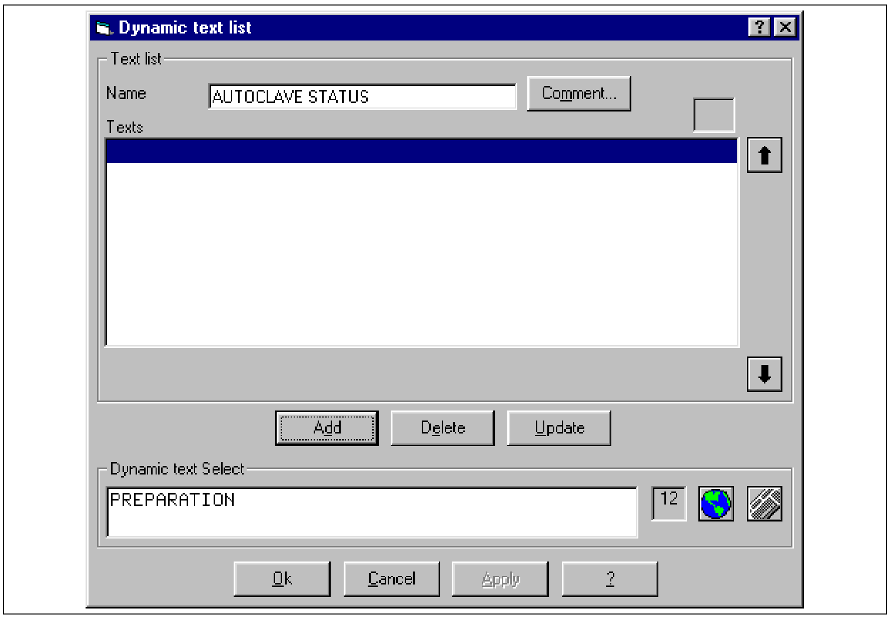
Insert dynamic text.

Click on Add



Name: TEXT_LIST_1

Assign the name and insert the first 📖.



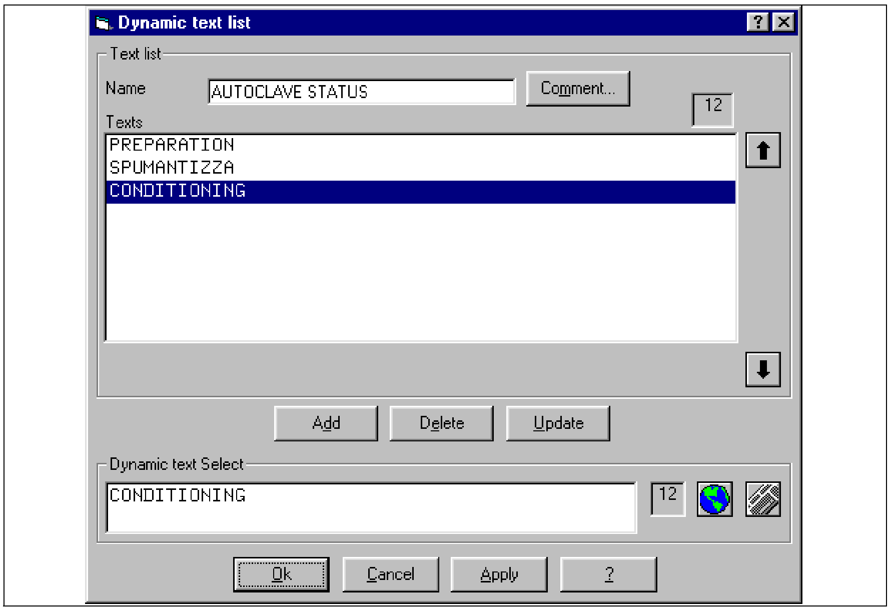
Name: AUTOCLAVE STATUS

Dynamic text Select: PREPARATION 12

Once the 📖 has been assigned, click on Add.

Insert all the texts.

Click on OK.



Name: AUTOCLAVE STATUS

Texts:
PREPARATION
SPUMANTIZZA
CONDITIONING

Dynamic text Select: CONDITIONING

Insert all the dynamic texts (See project offered as example).

Complete the page with the touch buttons and the multilanguage labels necessary.

Insert the pages listed below.

PAGE 9 -> Autoclave 2 commands

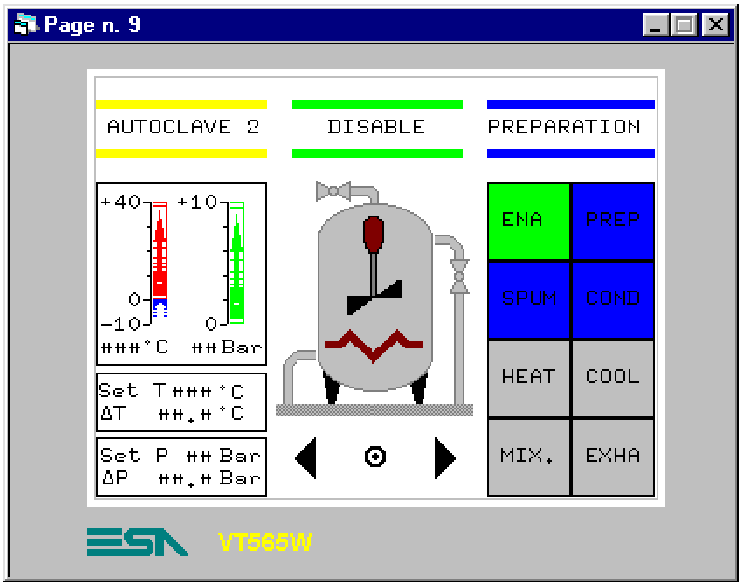


Table 8.7: Elements in page no. 9

IMAGES	COOL RIGHT
AUTOCLAVE - BACKGROUND	MIXER
TOUCH BUTTONS	HEATING
2 ENABLE	**DYNAMIC FIELD**
2 PREPARATION	AUTOCLAVE E/D
2 SPUMANTIZZA	AUTOCLAVE STATUS
2 CONDITIONING	**BAR DATA FIELD**
2 HEATING	TEMP. A2+
2 COOLING	TEMP. A2-
2 MIXER	PRESS. AUTOCLAVE 2
2 EXHAUST	**NUMERICAL DATA FIELD**
PREVIOUS MENU -> 7	TEMP. A2
ARROW R -> P10	PRESS. AUTOCLAVE 2
ARROW L -> P8	SET TEMP. A2 PRE
SYMBOLIC FIELD	SET dT TEMP. A2 PRE
EXHAUST	SET PRES. A2 PRE
COOL LEFT	SET dP PRES. A2 PRE

As for page 8.

PAGE 10 -> Autoclave 3 commands

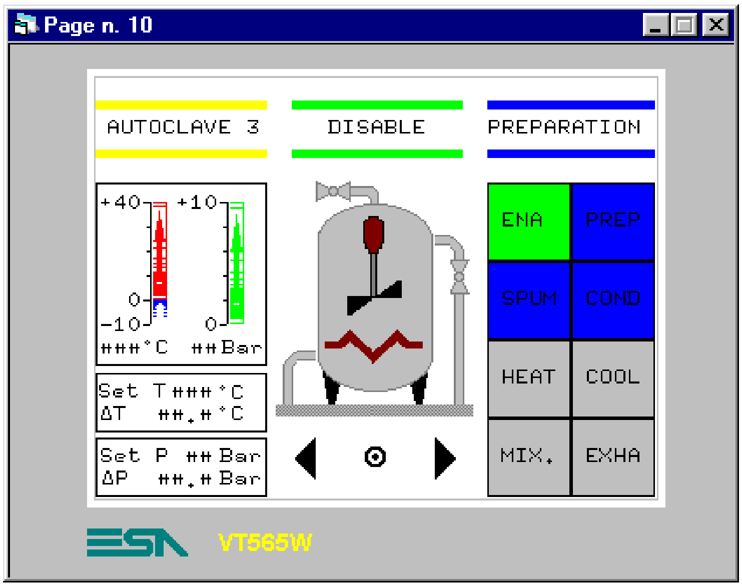


Table 8.8: Elements in page no. 10

IMAGES	COOL RIGHT
AUTOCLAVE - BACKGROUND	MIXER
TOUCH BUTTONS	HEATING
3 ENABLE	**DYNAMIC FIELDS**
3 PREPARATION	AUTOCLAVE E/D
3 SPUMANTIZZA	AUTOCLAVE STATUS
3 CONDITIONING	**BAR DATA FIELDS**
3 HEATING	TEMP. A3+
3 COOLING	TEMP. A3-
3 MIXER	PRESS. AUTOCLAVE 3
3 EXHAUST	**NUMERICAL DATA FIELDS**
PREVIOUS MENU -> 7	TEMP. A3
ARROW R -> P11	PRESS. AUTOCLAVE 3
ARROW L -> P9	SET TEMP. A3 PRE
SYMBOLIC FIELDS	SET dT TEMP. A3 PRE
EXHAUST	SET PRES. A3 PRE
COOL LEFT	SET dP PRES. A3 PRE

As for page 8.

PAGE 11 -> Autoclave 4 commands

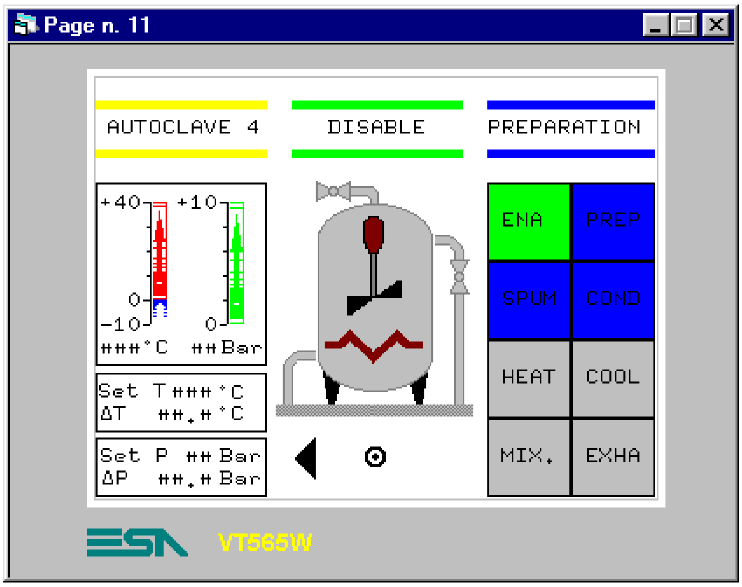


Table 8.9: Elements in page no.11

IMAGES	MIXER
AUTOCLAVE - BACKGROUND	HEATING
TOUCH BUTTONS	**DYNAMIC FIELDS**
4 ENABLE	AUTOCLAVE E/D
4 PREPARATION	AUTOCLAVE STATUS
4 SPUMANTIZZA	**BAR DATA FIELDS**
4 CONDITIONING	TEMP. A4+
4 HEATING	TEMP. A4-
4 COOLING	PRESS. AUTOCLAVE 4
4 MIXER	**NUMERICAL DATA FIELDS**
4 EXHAUST	TEMP. A4
PREVIOUS MENU -> 7	PRESS. AUTOCLAVE 4
ARROW L -> P10	SET TEMP. A4 PRE
SYMBOLIC FIELDS	SET dT TEMP. A4 PRE
EXHAUST	SET PRES. A4 PRE
COOL LEFT	SET dP PRES. A4 PRE
COOL RIGHT	

As for page 8.

PAGE 20 -> Edit Page

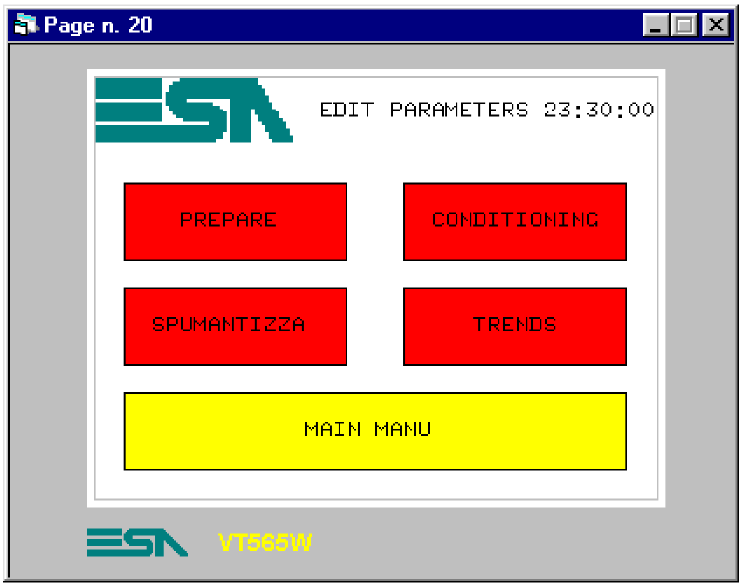


Table 8.10: Elements in page no. 20

IMAGES
ESA
TOUCH BUTTONS
EDIT- PREPARE
EDIT - CONDITIONING
EDIT - SPUMANTIZZA
EDIT - TREND
EDIT - MAIN MENU
DATE/TIME FIELD

This page is displayed by pressing the touch button “EDIT” in page 1; it gives access to the pages for setting the parameters necessary for the production process.

PAGE 21 -> Autoclave Prepare

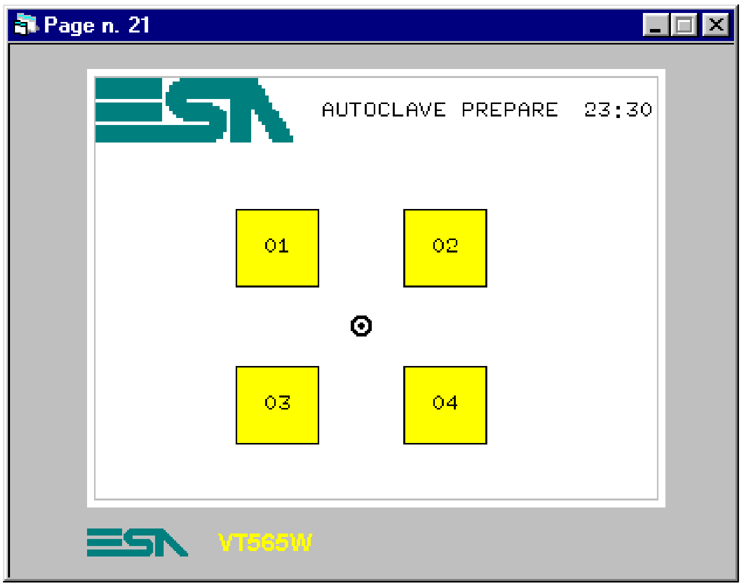


Table 8.11: Elements in page no. 21

IMAGES
ESA
TOUCH BUTTONS
BUTTON 1 PRE
BUTTON 2 PRE
BUTTON 3 PRE
BUTTON 4 PRE
PREVIOUS MENU -> 20
DATE/TIME FIELD

This page is displayed by pressing the touch button “PREPARE” in page 20; allows the user to choose which autoclave is to be parameterized.

PAGE 22 -> Autoclave 1 Prepare

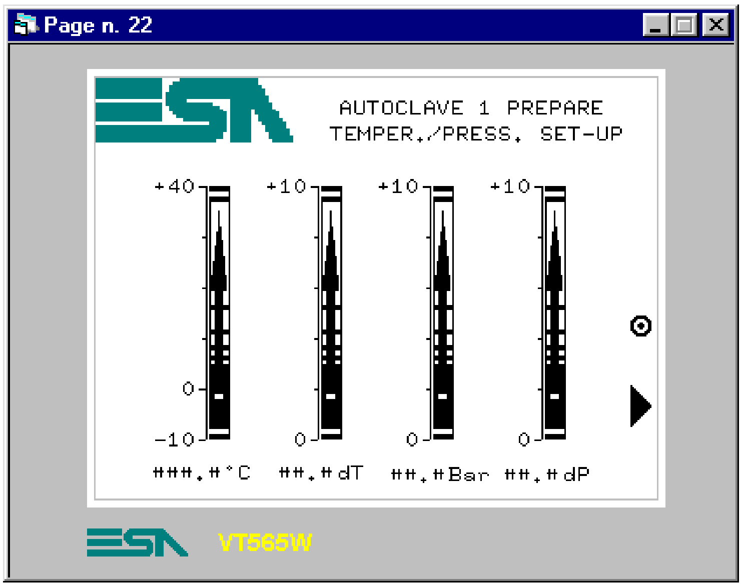


Table 8.12: Elements in page no. 22

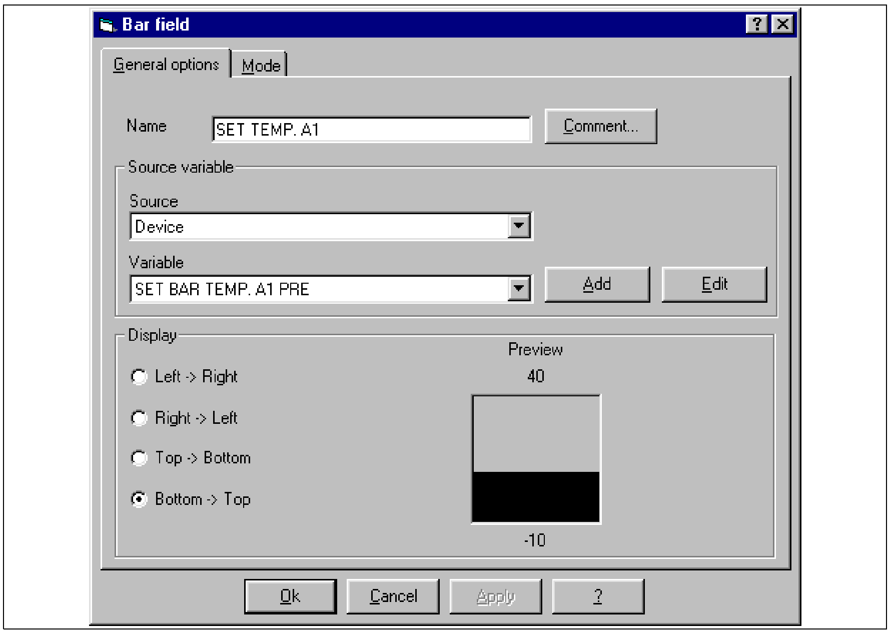
IMAGES	
	ESA
TOUCH BUTTONS	
	PREVIOUS MENU -> 21
	NEXT
BAR DATA FIELDS	
	SET BAR TEMP. A1 PRE
	SET BAR dT A1 PRE
	SET BAR PRES. A1 PRE
	SET BAR dP PRES. A1 PRE
NUMERICAL DATA FIELDS	
	SET TEMP. A1 PRE
	SET dT A1 PRE
	SET PRES. A1 PRE
	SET dP PRES. A1 PRE

This page is displayed by pressing the touch button “1” in page 21; it shows by means of bar and numerical data, both of which are settable, the temperature and pressure values necessary for the process.

The fact that the bar and numerical data are both settable means that the values can be attributed to the assigned variable using the VT.

Proceed as already described to insert the bar data.

Set as illustrated, then browse the 🗀 by clicking on Mode.



Bar field

General options | Mode

Name: SET TEMP. A1

Source variable

Source: Device

Variable: SET BAR TEMP. A1 PRE

Display

Left -> Right

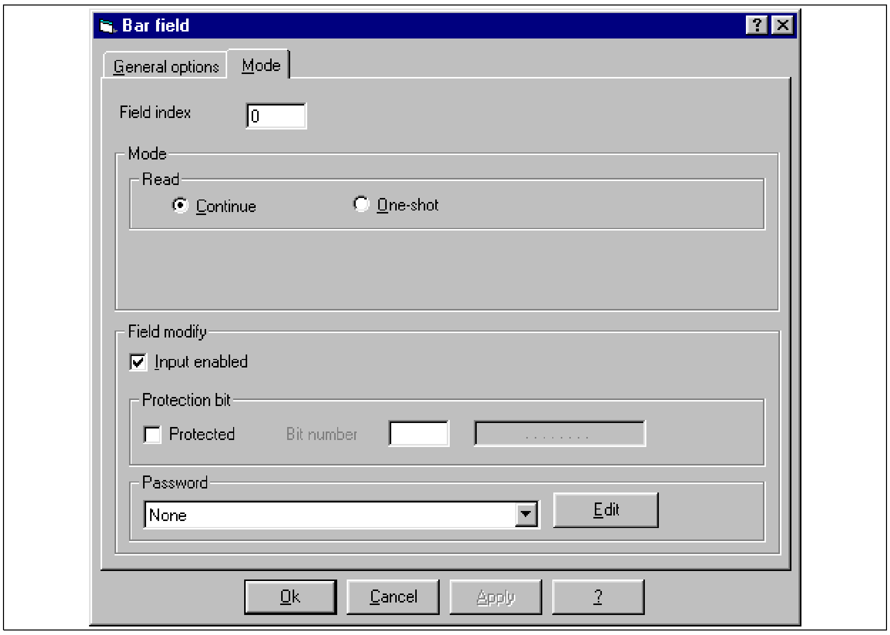
Right -> Left

Top -> Bottom

(●) Bottom -> Top

Compile as illustrated.

Click on OK.



Bar field

General options | Mode

Field index: 0

Mode

Read

(●) Continue ( ) One-shot

Field modify

[✓] Input enabled

Protection bit

[ ] Protected Bit number

Password: None

Note that Input enabled box is active.

This applies to all bar and numerical data.

PAGE 23 -> Autoclave 2 Prepare

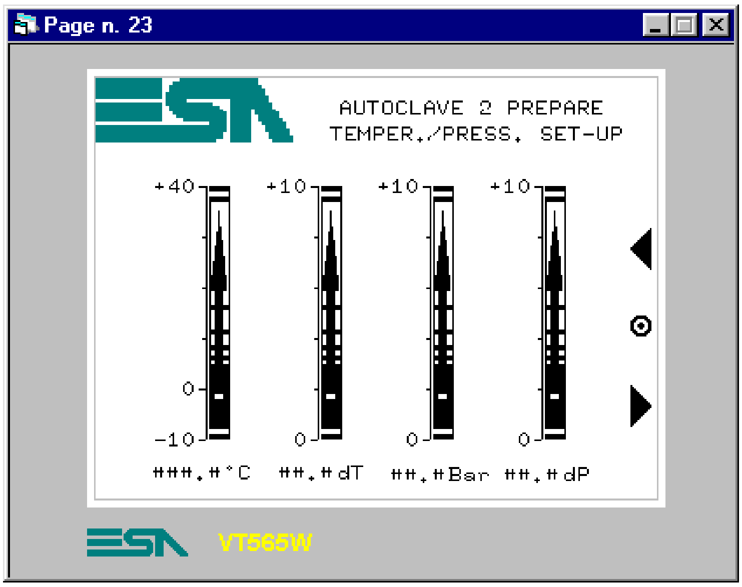


Table 8.13: Elements in page no. 23

IMAGES
ESA
TOUCH BUTTONS
PREVIOUS MENU -> 21
NEXT
PREVIOUS
BAR DATA FIELD
SET BAR TEMP. A2 PRE
SET BAR dT A2 PRE
SET BAR PRES. A2 PRE
SET BAR dP PRES. A2 PRE
NUMERICAL DATA FIELD
SET TEMP. A2 PRE
SET dT A2 PRE
SET PRES. A2 PRE
SET dP PRES. A2 PRE

This page is displayed by pressing the touch button marked “2” in page 21; it shows by means of bar and numerical data, both of which are settable, the temperature and pressure values necessary for the process.

PAGE 24 -> Autoclave 3 Prepare

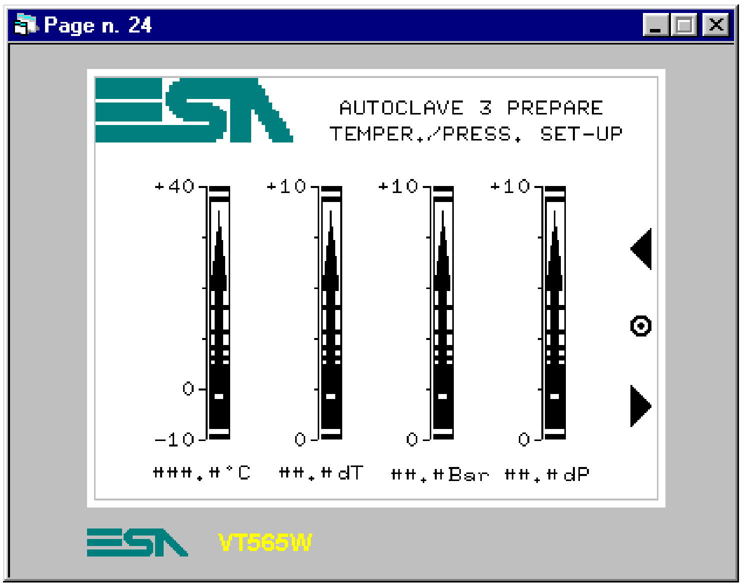


Table 8.14: Elements in page no. 24

IMAGES
ESA
TOUCH BUTTONS
PREVIOUS MENU -> 21
NEXT
PREVIOUS
BAR DATA FIELDS
SET BAR TEMP. A3 PRE
SET BAR dT A3 PRE
SET BAR PRES. A3 PRE
SET BAR dP PRES. A3 PRE
NUMERICAL DATA FIELDS
SET TEMP. A3 PRE
SET dT A3 PRE
SET PRES. A3 PRE
SET dP PRES. A3 PRE

This page is displayed by pressing the touch button marked “3” in page 21; it shows by means of bar and numerical data, both of which are settable, the temperature and pressure values necessary for the process.

PAGE 25 -> Autoclave 4 Prepare

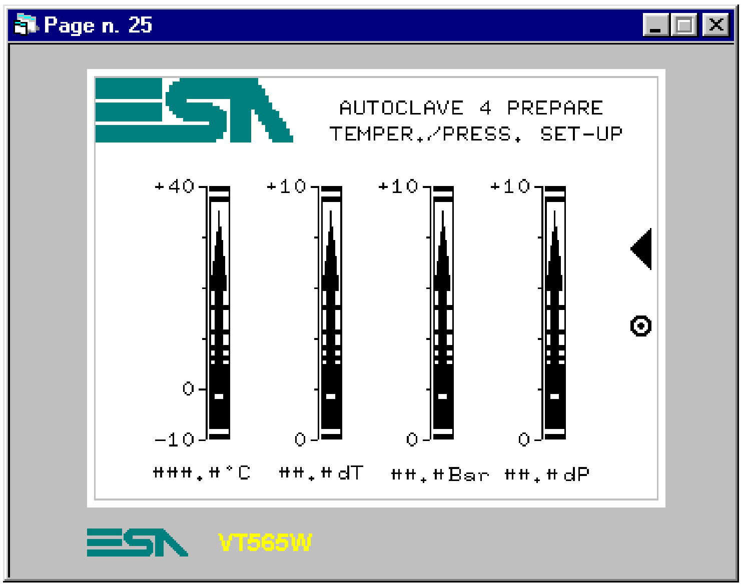


Table 8.15: Elements in page no. 24

IMAGES
ESA
TOUCH BUTTONS
PREVIOUS MENU -> 21
PREVIOUS
BAR DATA FIELDS
SET BAR TEMP. A4 PRE
SET BAR dT A4 PRE
SET BAR PRES. A4 PRE
SET BAR dP PRES. A4 PRE
NUMERICAL DATA FIELDS
SET TEMP. A4 PRE
SET dT A4 PRE
SET PRES. A4 PRE
SET dP PRES. A4 PRE

This page is displayed by pressing the touch button marked “4” in page 21; it shows by means of bar and numerical data, both of which are settable, the temperature and pressure values necessary for the process.

PAGE 26 -> Autoclave conditioning

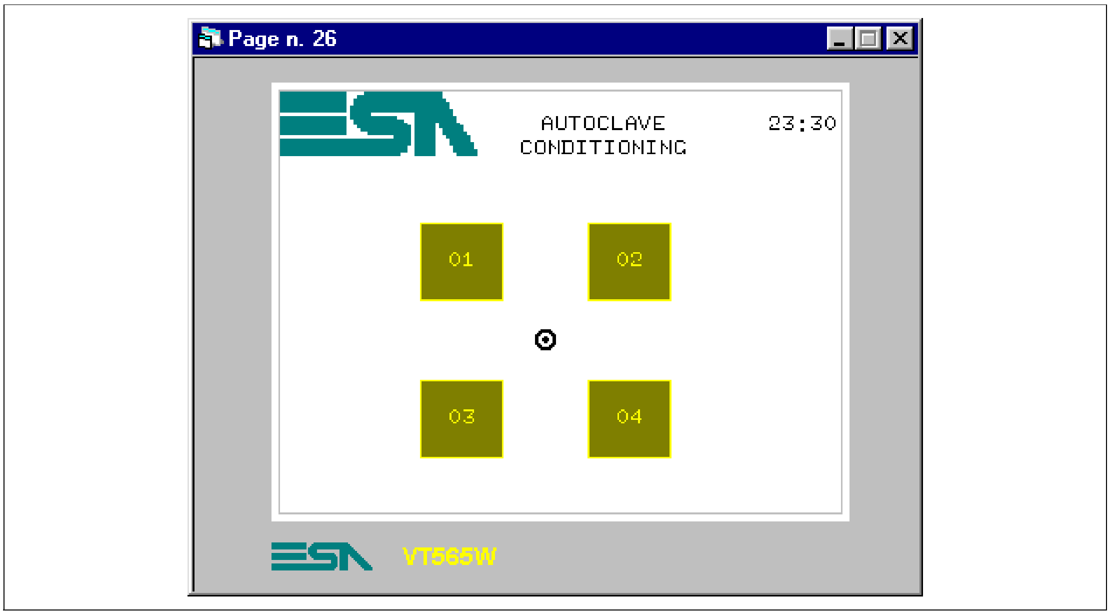


Table 8.16: Elements in page no. 26

IMAGES
ESA
TOUCH BUTTONS
BUTTON 1 CON
BUTTON 2 CON
BUTTON 3 CON
BUTTON 4 CON
PREVIOUS MENU -> 20
DATE/TIME FIELDS

This page is displayed by pressing the touch button marked “CONDITIONING” in page 20; it allows you to select which autoclave is to be parameterized.

PAGE 27 -> Autoclave 1 conditioning

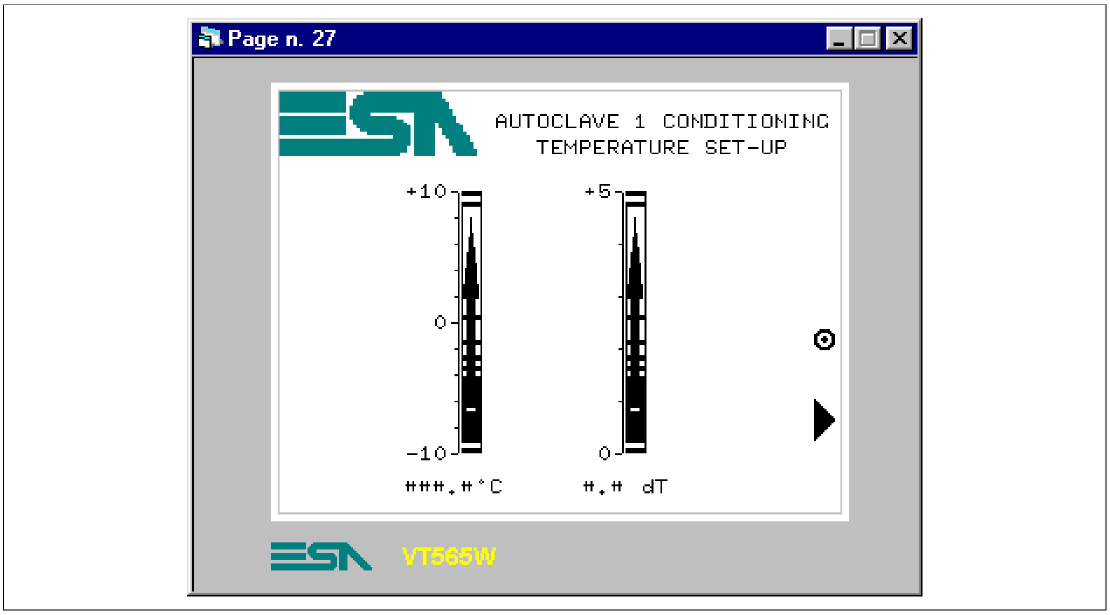


Table 8.17: Elements in page no. 27

IMAGES	
	ESA
TOUCH BUTTONS	
	PREVIOUS MENU -> 26
	NEXT
BAR DATA FIELDS	
	SET BAR TEMP. A1 CON
	SET BAR dT A1 CON
NUMERICAL DATA FIELDS	
	SET TEMP. A1 CON
	SET dT A1 CON

This page is displayed by pressing touch button “1” in page 26; it shows by means of bar and numerical data, both of which are settable, the temperature values necessary for the process.

PAGE 28 -> Autoclave 2 conditioning

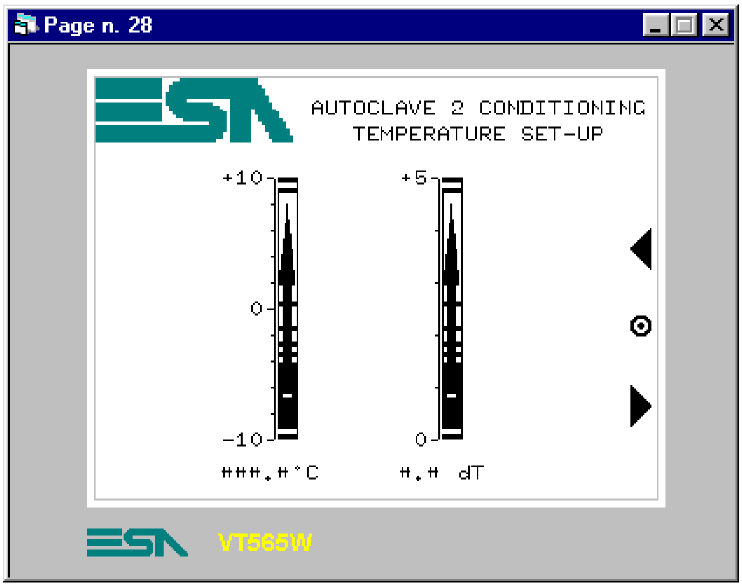


Table 8.18: Elements in page no. 28

IMAGES
ESA
TOUCH BUTTONS
PREVIOUS MENU -> 26
NEXT
PREVIOUS
BAR DATA FIELDS
SET BAR TEMP. A2 CON
SET BAR dT A2 CON
NUMERICAL DATA FIELDS
SET TEMP. A2 CON
SET dT A2 CON

This page is displayed by pressing the touch button marked “2” in page 26; it shows by means of bar and numerical data, both of which are settable, the temperature values necessary for the process.

PAGE 29 -> Autoclave 3 conditioning

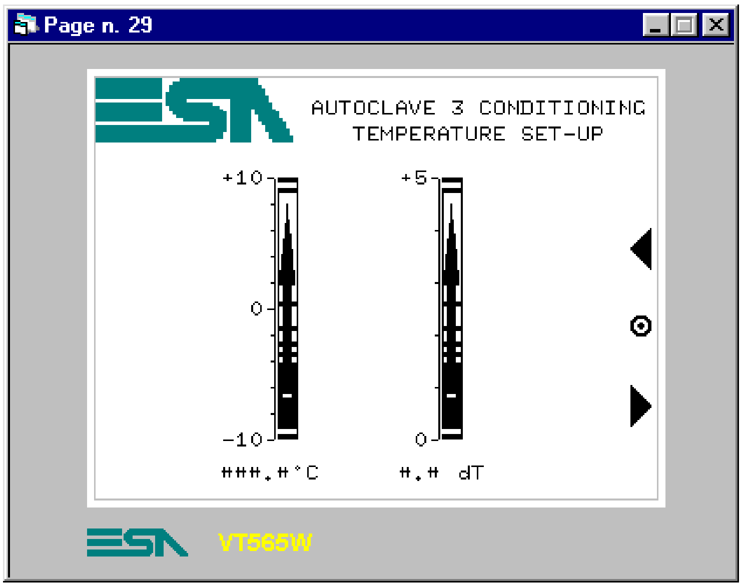


Table 8.19: Elements in page no. 29

IMAGES
ESA
TOUCH BUTTONS
PREVIOUS MENU -> 26
NEXT
PREVIOUS
BAR DATA FIELDS
SET BAR TEMP. A3 CON
SET BAR dT A3 CON
NUMERICAL DATA FIELDS
SET TEMP. A3 CON
SET dT A3 CON

This page is displayed by pressing the touch button marked “3” in page 26; it shows by means of bar and numerical data, both of which are settable, the temperature values necessary for the process.

PAGE 30 -> Autoclave 4 conditioning

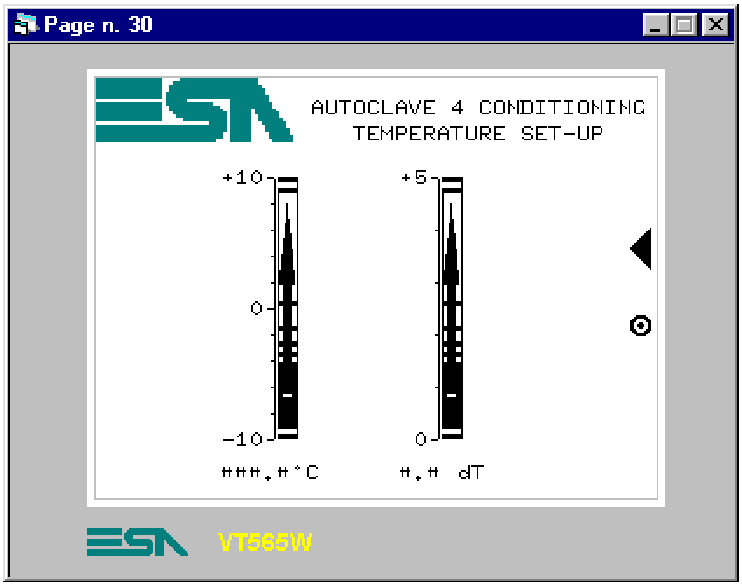


Table 8.20: Elements in page no. 30

IMAGES
ESA
TOUCH BUTTONS
PREVIOUS MENU -> 26
PREVIOUS
BAR DATA FIELDS
SET BAR TEMP. A4 CON
SET BAR dT A4 CON
NUMERICAL DATA FIELDS
SET TEMP. A4 CON
SET dT A4 CON

This page is displayed by pressing the touch button marked “4” in page 26; it shows by means of bar and numerical data, both of which are settable, the temperature values necessary for the process.

PAGE 40 -> Set-up

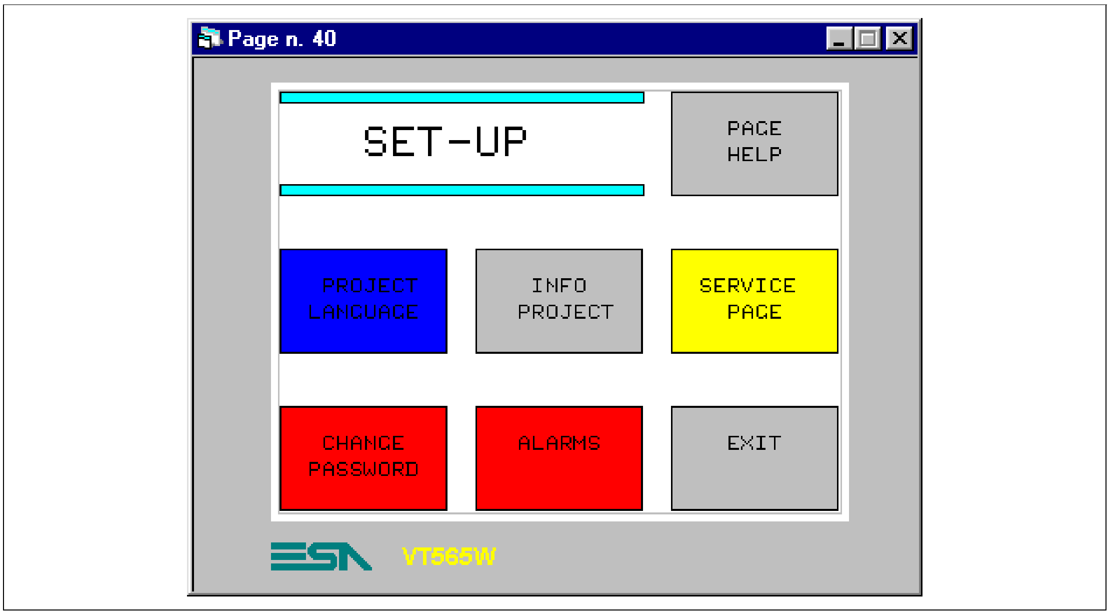


Table 8.21: Elements in page no. 40

TOUCH BUTTONS
SET PROJECT LANGUAGE
SET INFO PROJECT
SET CHANGE PASSWORD
SET PAGE HELP
SET SERVICE PAGE
SET MAIN MENU
ALARM - GO TO PAGE

This page is displayed by pressing the touch button marked “SPUMANTIZZA” in page 20; it allows the user to select which autoclave needs to be parameterized.

PAGE 41 -> Project Language

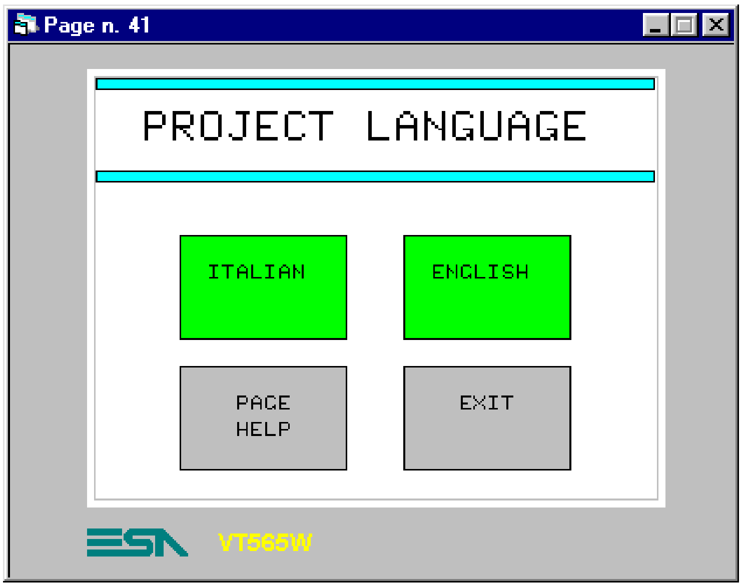
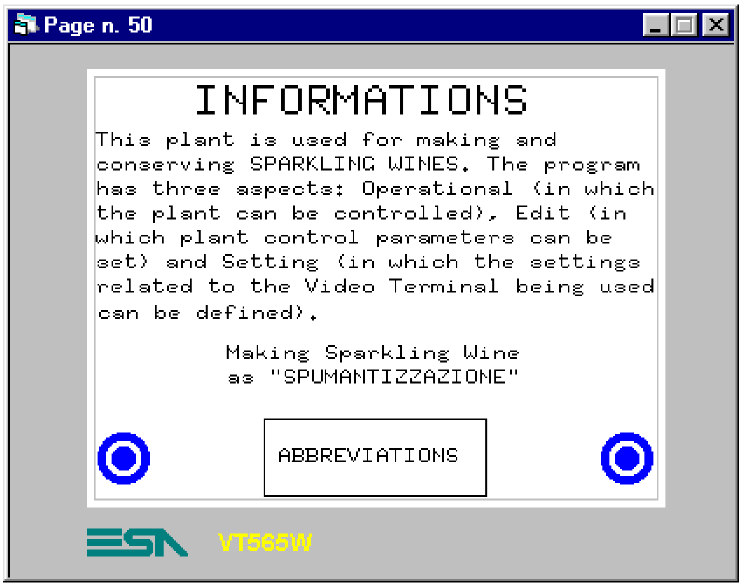


Table 8.22: Elements in page no. 41

TOUCH BUTTONS
ITALIAN
ENGLISH
SET PAGE HELP
SET EXIT TO PREVIOUS

This page is displayed by pressing the touch button marked "PROJECT LANGUAGE" in page 40; This shows the languages in which the project can be displayed on the VT.

PAGE 50 -> Information



INFORMATIONS

This plant is used for making and conserving SPARKLING WINES. The program has three aspects: Operational (in which the plant can be controlled), Edit (in which plant control parameters can be set) and Setting (in which the settings related to the Video Terminal being used can be defined).

Making Sparkling Wine as "SPUMANTIZZAZIONE"

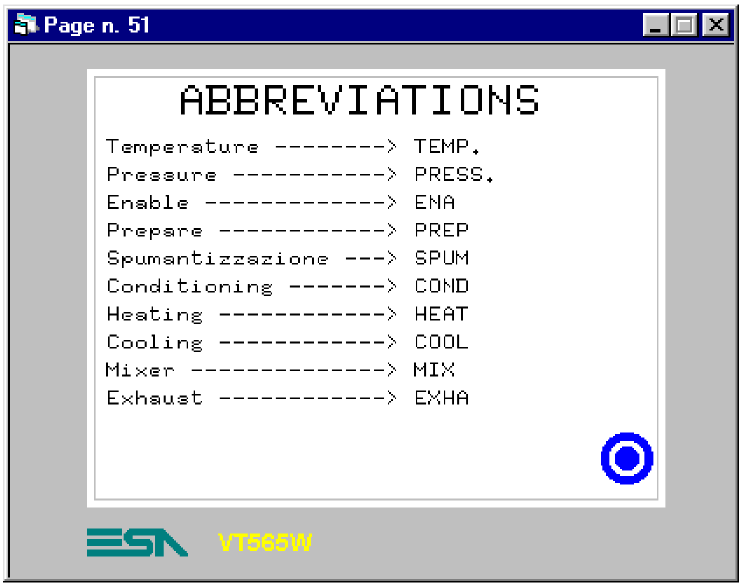
ABBREVIATIONS

Table 8.23: Elements in page no. 50

TOUCH BUTTONS
ABBREVIATIONS
PREVIOUS MENU -> 1

This page is displayed by pressing the touch button marked “Info” in page 1; it shows a page for information purposes composed exclusively of multilanguage labels.

PAGE 51 -> Abbreviations



ABBREVIATIONS

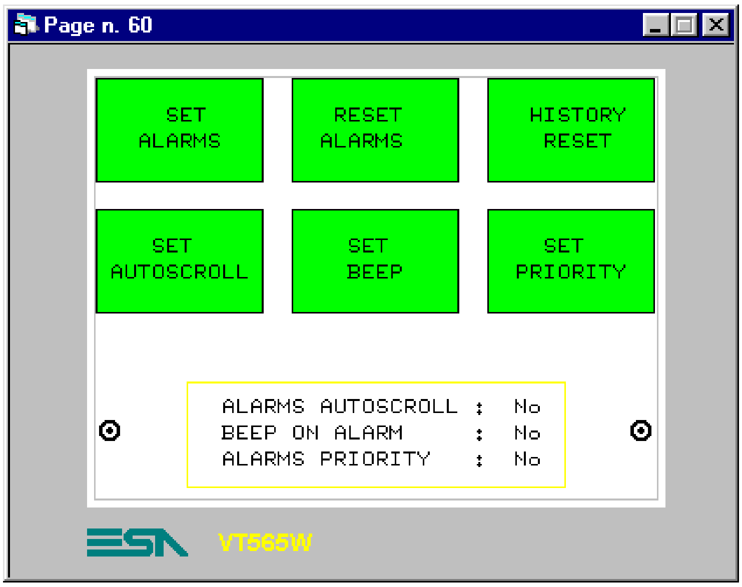
Temperature --------->	TEMP.
Pressure ------------>	PRESS.
Enable -------------->	ENA
Prepare ------------->	PREP
Spumantizzazione ---->	SPUM
Conditioning -------->	COND
Heating ------------->	HEAT
Cooling ------------->	COOL
Mixer --------------->	MIX
Exhaust ------------->	EXHA

Table 8.24: Elements in page no. 51

TOUCH BUTTONS
PREVIOUS MENU -> 1

This page is displayed by pressing the touch button marked “ABBREVIATIONS” in page 50; It shows a page for information purposes composed exclusively of multilanguage labels.

PAGE 60 -> Alarms



```
ALARMS AUTOSCROLL :  No
BEEP ON ALARM     :  No
ALARMS PRIORITY   :  No
```

Table 8.25: Elements in page no. 60

TOUCH BUTTONS
PREVIOUS MENU -> 40
DEMO - SET ALARM
DEMO - RESET ALARM
DEMO - HISTORY RESET
DEMO - SET AUTOSCRO.
DEMO - SET BEEP
DEMO - SET PRIORITY
DYNAMIC FIELD
NO/YES

This page is displayed by pressing the touch button marked “ALARM” in page 40; it shows a series of buttons to simulate alarms, given that in reality there is no functioning plant.

PAGE 65 -> Help Overpress.

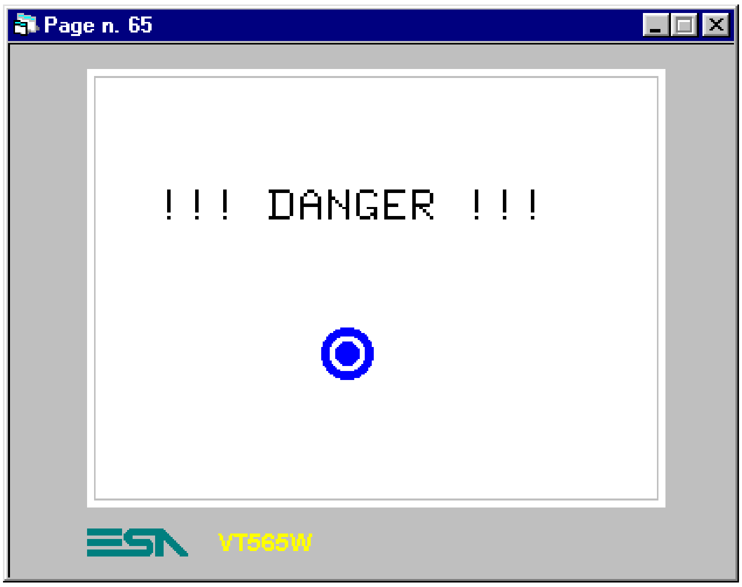


Table 8.26: Elements in page no. 65

TOUCH BUTTONS
PREVIOUS MENU -> 60

This page is displayed by pressing the touch button marked “????” in alarm display mode; it shows an example of supplementary help to explain the alarm.

PAGE 70 -> Recipes

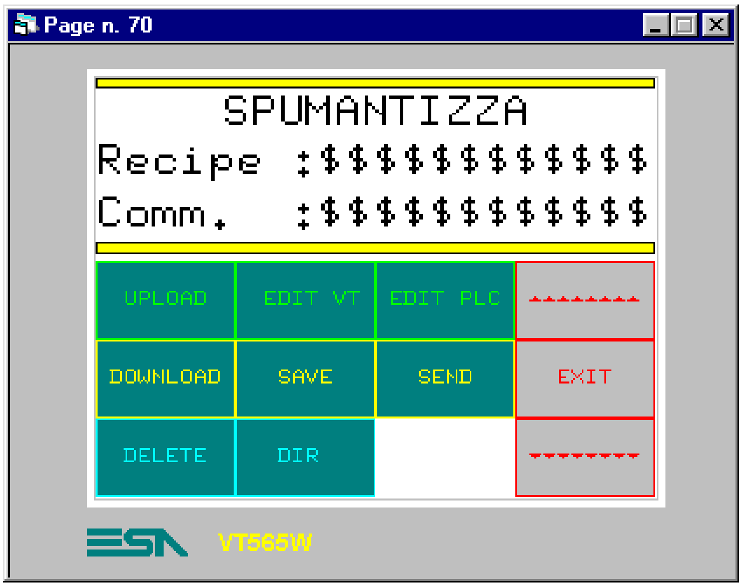


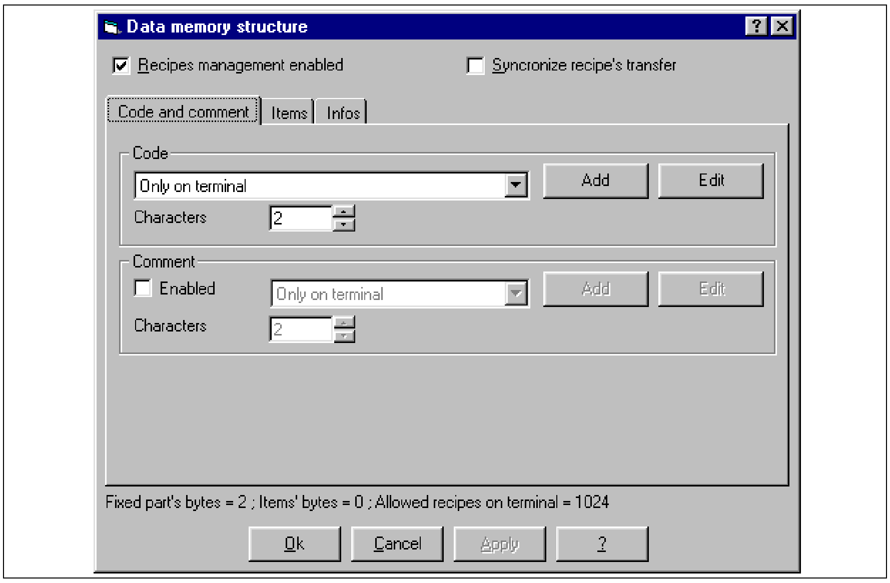
Table 8.27: Elements in page no. 70

TOUCH BUTTONS
RECIPE - DELETE
RECIPE - UPLOAD
RECIPE - EDIT VT
RECIPE - EDIT PLC
RECIPE - DOWNLOAD
RECIPE - SEND
RECIPE - SAVE
RECIPE - DIR
RECIPE - EXIT -> 20
RECIPE - PAGE DOWN
RECIPE - PAGE UP
ASCII FIELD
RECIPE CODE
RECIPE COMMENT

This page is displayed by pressing the touch button marked “SPUMANTIZZA” in page 20; it shows a series of commands for managing the recipes needed to control the process.

The first thing is to declare which variables will be needed to compose the recipe (See “Chapter 6 -> Data memory structure“).

Click on ***Configure > Data memory structure*** and the following mask appears on screen:

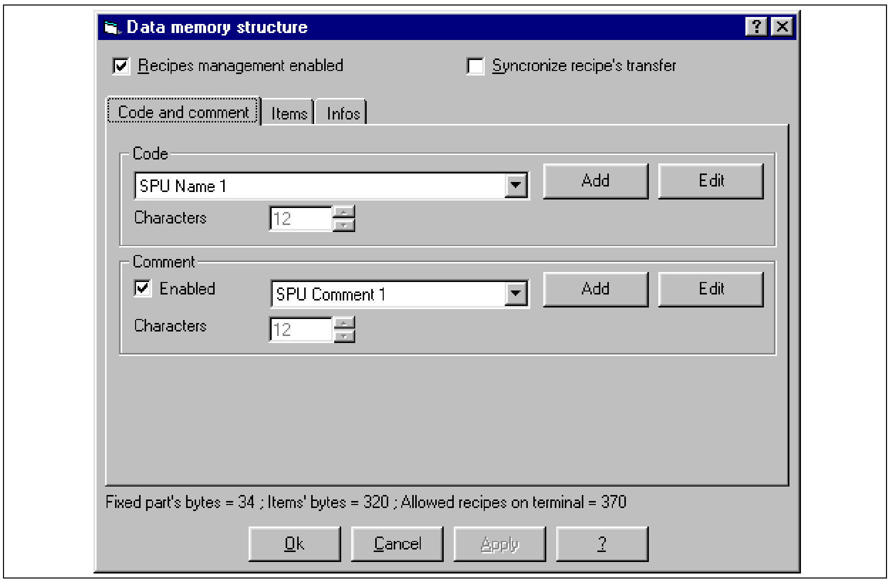

Data memory structure

☑ Recipes management enabled
☐ Syncronize recipe's transfer

Code
Only on terminal
Characters 2

Comment
☐ Enabled
Only on terminal
Characters 2

Fixed part's bytes = 2 ; Items' bytes = 0 ; Allowed recipes on terminal = 1024


Data memory structure

☑ Recipes management enabled
☐ Syncronize recipe's transfer

Code
SPU Name 1
Characters 12

Comment
☑ Enabled
SPU Comment 1
Characters 12

Fixed part's bytes = 34 ; Items' bytes = 320 ; Allowed recipes on terminal = 370

Set the parameters as illustrated.

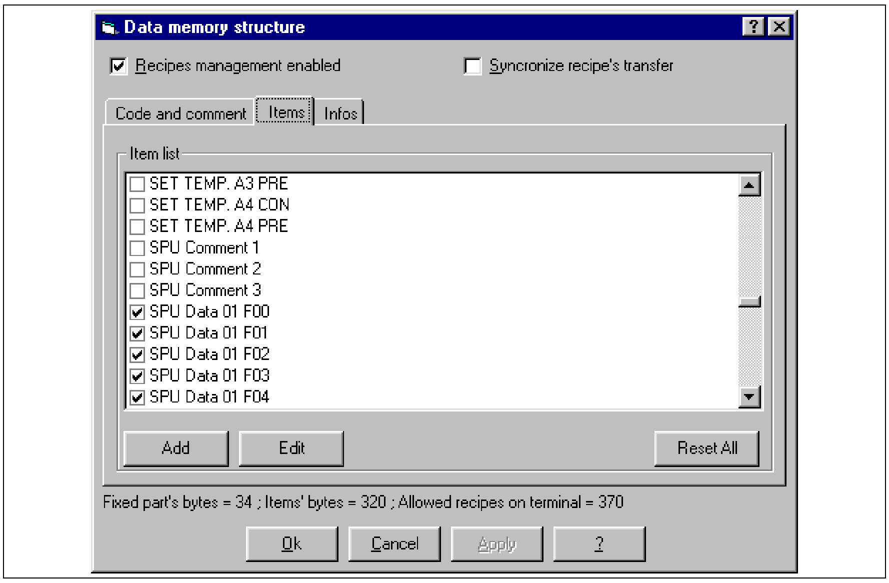
Select the 🕮 Options.

⚠ **The length of the code and the comment depend on the number of characters declared in the variable.**

Identify the variables that must be inserted in the recipe, then select them.

Set the parameters as illustrated.

Select the 📖 Information.


Data memory structure

☑ Recipes management enabled ☐ Syncronize recipe's transfer

Code and comment | Items | Infos

Item list

☐	SET TEMP. A3 PRE
☐	SET TEMP. A4 CON
☐	SET TEMP. A4 PRE
☐	SPU Comment 1
☐	SPU Comment 2
☐	SPU Comment 3
☑	SPU Data 01 F00
☑	SPU Data 01 F01
☑	SPU Data 01 F02
☑	SPU Data 01 F03
☑	SPU Data 01 F04

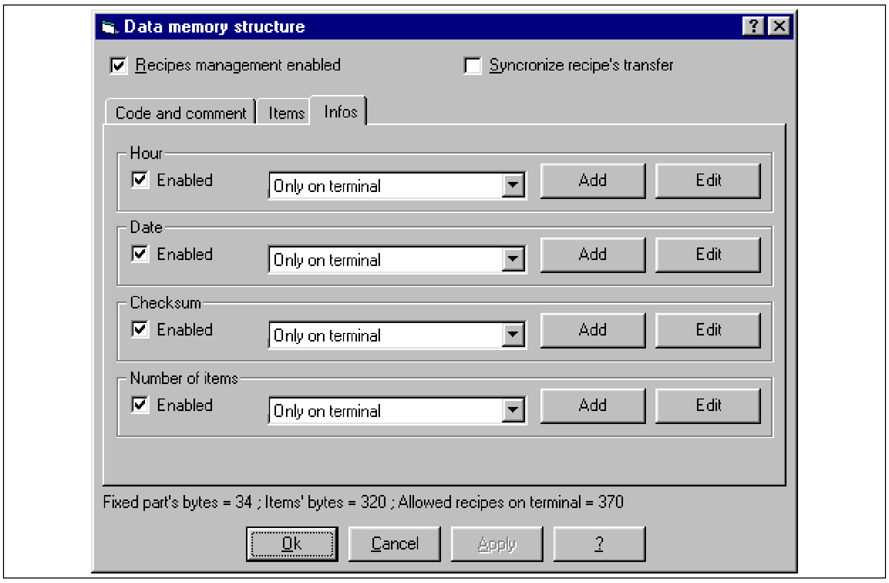
Fixed part's bytes = 34 ; Items' bytes = 320 ; Allowed recipes on terminal = 370

The variables "recipe code" and "recipe comment" do not need to be inserted because they are not part of the options of the recipe (See "Chapter 6 -> Data memory structure").

The information shown in the figure is contained only in the VT.

Set the parameters as illustrated.

Click on Ok.


Data memory structure

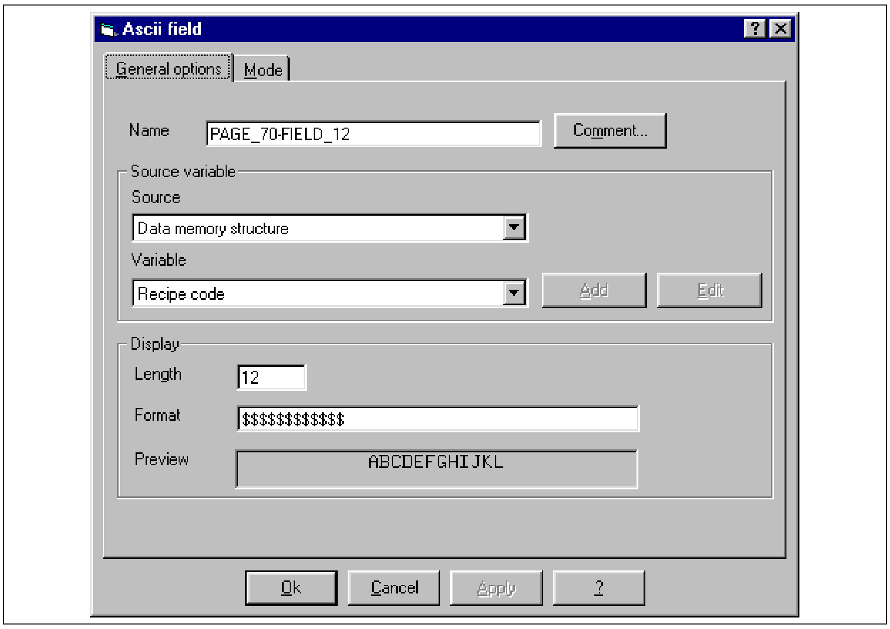
☑ Recipes management enabled ☐ Syncronize recipe's transfer

Code and comment | Items | Infos

Hour	☑ Enabled	Only on terminal
Date	☑ Enabled	Only on terminal
Checksum	☑ Enabled	Only on terminal
Number of items	☑ Enabled	Only on terminal

Fixed part's bytes = 34 ; Items' bytes = 320 ; Allowed recipes on terminal = 370

Once all the variables that comprise the recipe have been defined insert the code and the comment in the page. Click on ABC, put the cursor in the page and click.



Ascii field

General options | Mode

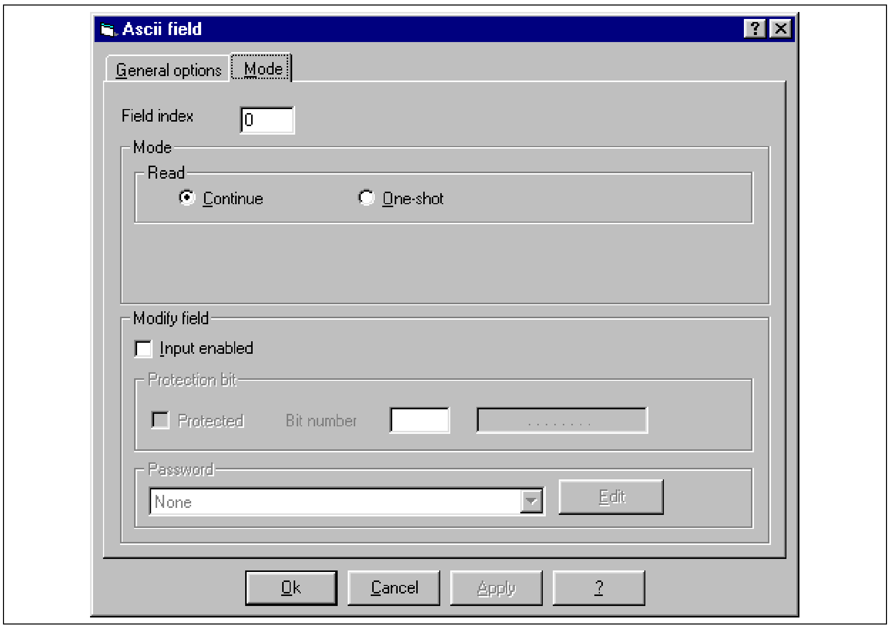
Name: PAGE_70-FIELD_12

Source variable
Source: Data memory structure
Variable: Recipe code

Display
Length: 12
Format: $$$$$$$$$$$$
Preview: ABCDEFGHIJKL

Set the parameters as illustrated.

Select the 🕮 Mode



Ascii field

General options | Mode

Field index: 0

Mode
Read: (● Continue) ○ One-shot

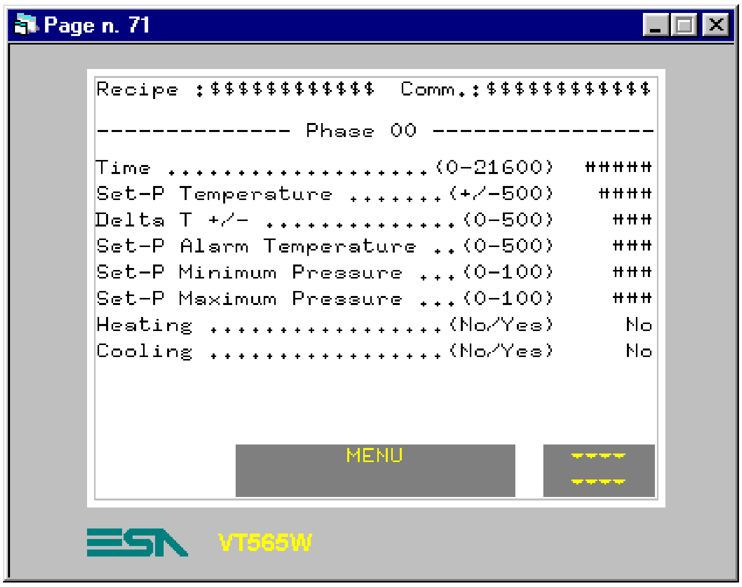
Modify field
☐ Input enabled
Protection bit: ☐ Protected Bit number
Password: None

Set the parameters as illustrated.

Click on Ok.

Insert the variable recipe comment using the same procedure. (See project offered as example).

PAGE 71 -> Recipe - Page 1



Page n. 71

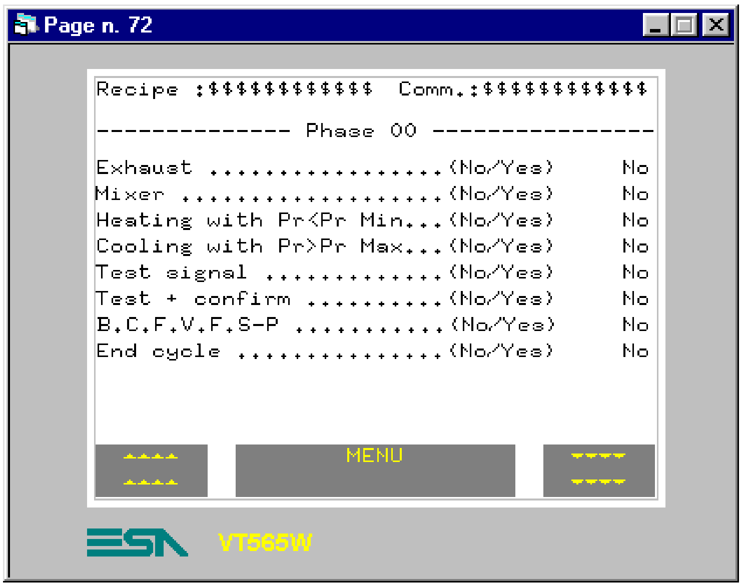
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 00 ----------------

Time ...................(0-21600)  #####
Set-P Temperature .......(+/-500)   ####
Delta T +/- ..............(0-500)    ###
Set-P Alarm Temperature ..(0-500)    ###
Set-P Minimum Pressure ...(0-100)    ###
Set-P Maximum Pressure ...(0-100)    ###
Heating ................(No/Yes)     No
Cooling ................(No/Yes)     No
```

Table 8.28: Elements in page no. 71

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
NUMERICAL FIELDS
SPU DATA 01 F00
SPU DATA 02 F00
SPU DATA 03 F00
SPU DATA 04 F00
SPU DATA 05 F00
SPU DATA 06 F00
DYNAMIC TEXT FIELDS
SPU DATA 07 F00
SPU DATA 08 F00
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button “ --------- “ in page 70; it shows the first part of the recipe needed to control the process.

PAGE 72 -> Recipe - Page 2



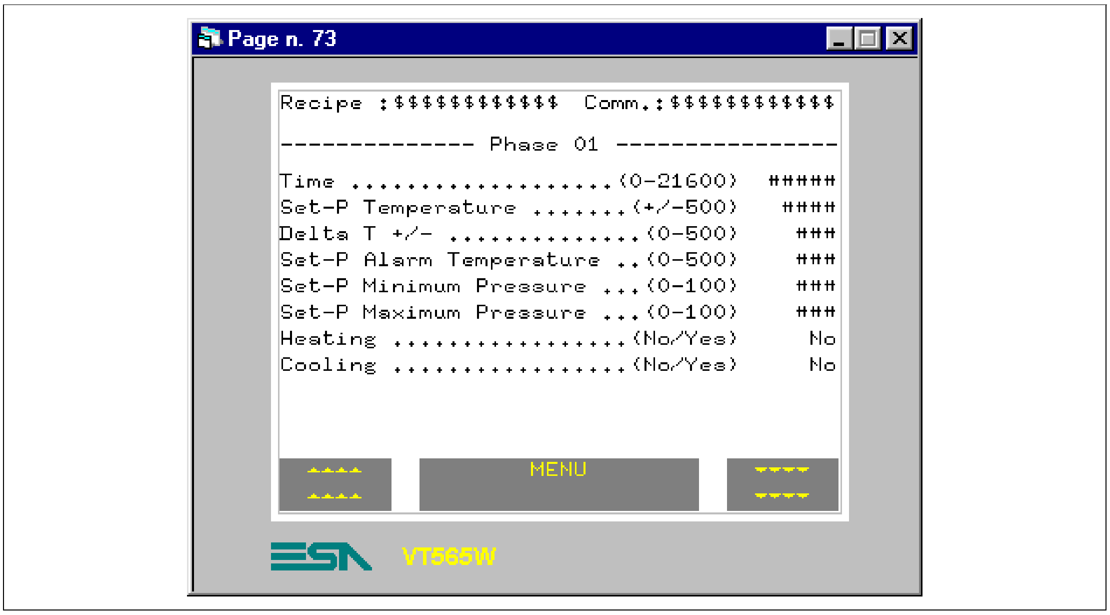
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 00 ----------------

Exhaust ................(No/Yes)     No
Mixer ..................(No/Yes)     No
Heating with Pr<Pr Min...(No/Yes)    No
Cooling with Pr>Pr Max...(No/Yes)    No
Test signal ............(No/Yes)     No
Test + confirm ..........(No/Yes)    No
B.C.F.V.F.S-P ...........(No/Yes)    No
End cycle ..............(No/Yes)     No
```

Table 8.29: Elements in page no. 72

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
DYNAMIC TEXT FIELDS
SPU DATA 09 F00
SPU DATA 10 F00
SPU DATA 11 F00
SPU DATA 12 F00
SPU DATA 13 F00
SPU DATA 14 F00
SPU DATA 15 F00
SPU DATA 16 F00
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button “ ” in page 71; it shows the next part of the recipe needed to control the process.

PAGE 73 -> Recipe - Page 3



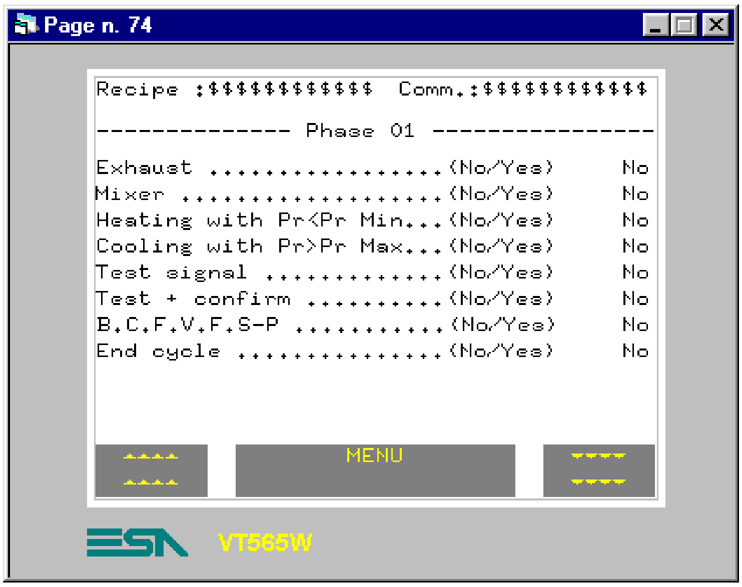
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 01 ----------------

Time ...................(0-21600)  #####
Set-P Temperature .......(+/-500)   ####
Delta T +/- ..............(0-500)    ###
Set-P Alarm Temperature ..(0-500)    ###
Set-P Minimum Pressure ...(0-100)    ###
Set-P Maximum Pressure ...(0-100)    ###
Heating .................(No/Yes)     No
Cooling .................(No/Yes)     No
```

Table 8.30: Elements in page no. 73

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
NUMERICAL FIELDS
SPU DATA 01 F01
SPU DATA 02 F01
SPU DATA 03 F01
SPU DATA 04 F01
SPU DATA 05 F01
SPU DATA 06 F01
DYNAMIC TEXT FIELDS
SPU DATA 05 F01
SPU DATA 06 F01
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button “ “ in page 72; it shows the next part of the recipe needed to control the process.

PAGE 74 -> Recipe - Page 4



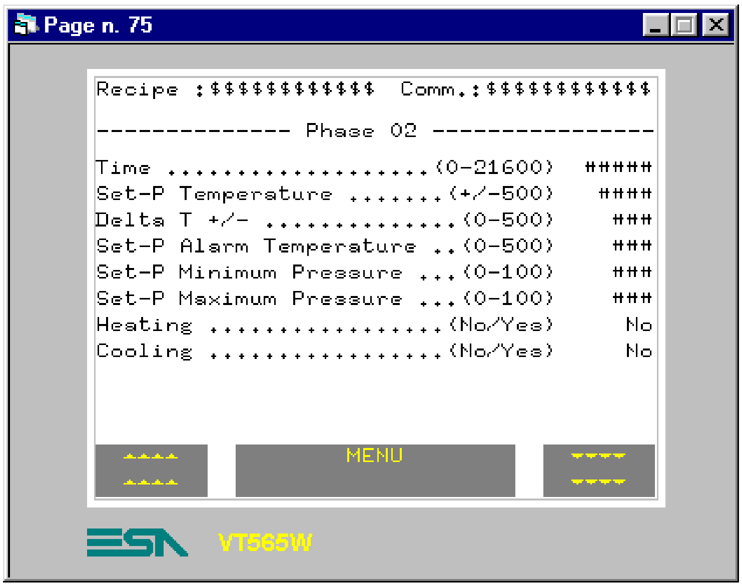
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 01 ----------------

Exhaust ................(No/Yes)     No
Mixer ..................(No/Yes)     No
Heating with Pr<Pr Min...(No/Yes)     No
Cooling with Pr>Pr Max...(No/Yes)     No
Test signal ............(No/Yes)     No
Test + confirm .........(No/Yes)     No
B.C.F.V.F.S-P ..........(No/Yes)     No
End cycle ..............(No/Yes)     No
```

Table 8.31: Elements in page no. 74

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
DYNAMIC TEXT FIELDS
SPU DATA 09 F01
SPU DATA 10 F01
SPU DATA 11 F01
SPU DATA 12 F01
SPU DATA 13 F01
SPU DATA 14 F01
SPU DATA 15 F01
SPU DATA 16 F01
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button marked “ “ in page 73; it shows the next part of the recipe needed to control the process.

PAGE 75 -> Recipe - Page 5



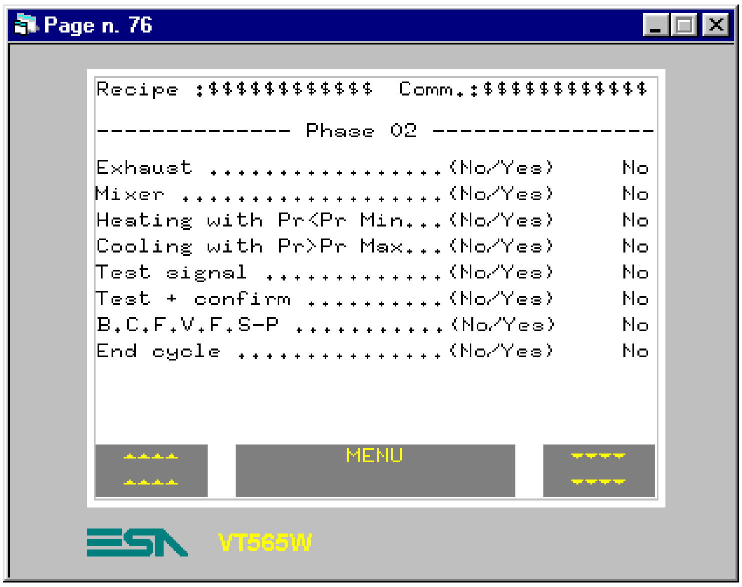
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 02 ----------------

Time ...................(0-21600)  #####
Set-P Temperature .......(+/-500)   ####
Delta T +/- ..............(0-500)    ###
Set-P Alarm Temperature ..(0-500)    ###
Set-P Minimum Pressure ...(0-100)    ###
Set-P Maximum Pressure ...(0-100)    ###
Heating ................(No/Yes)     No
Cooling ................(No/Yes)     No
```

Table 8.32: Elements in page no. 75

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
NUMERICAL FIELDS
SPU DATA 01 F02
SPU DATA 02 F02
SPU DATA 03 F02
SPU DATA 04 F02
SPU DATA 05 F02
SPU DATA 06 F02
DYNAMIC TEXT FIELDS
SPU DATA 05 F02
SPU DATA 06 F02
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button marked “ ” in page 74; it shows the next part of the recipe needed to control the process.

PAGE 76 -> Recipe - Page 6



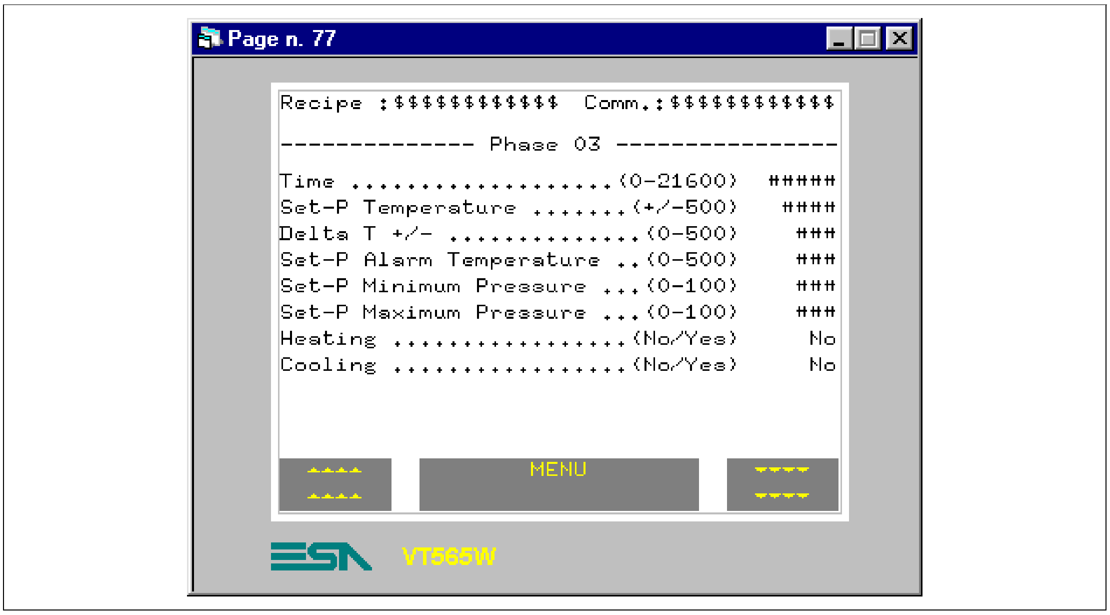
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 02 ----------------

Exhaust .................(No/Yes)     No
Mixer ...................(No/Yes)     No
Heating with Pr<Pr Min...(No/Yes)     No
Cooling with Pr>Pr Max...(No/Yes)     No
Test signal .............(No/Yes)     No
Test + confirm ..........(No/Yes)     No
B.C.F.V.F.S-P ...........(No/Yes)     No
End cycle ...............(No/Yes)     No
```

Table 8.33: Elements in page no. 76

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
DYNAMIC TEXT FIELDS
SPU DATA 09 F02
SPU DATA 10 F02
SPU DATA 11 F02
SPU DATA 12 F02
SPU DATA 13 F02
SPU DATA 14 F02
SPU DATA 15 F02
SPU DATA 16 F02
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button marked “ “ in page 75; it shows the next part of the recipe needed to control the process.

PAGE 77 -> Recipe - Page 7



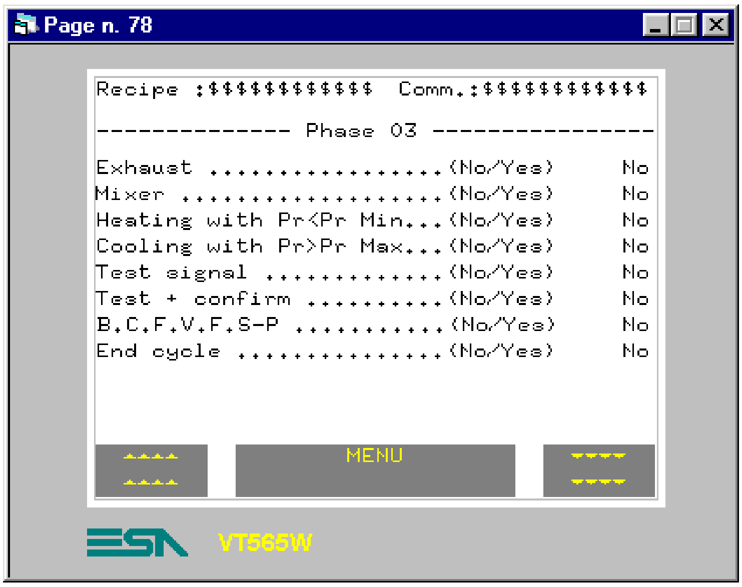
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 03 ----------------

Time ...................(0-21600)  #####
Set-P Temperature .......(+/-500)  ####
Delta T +/- ..............(0-500)   ###
Set-P Alarm Temperature ..(0-500)   ###
Set-P Minimum Pressure ...(0-100)   ###
Set-P Maximum Pressure ...(0-100)   ###
Heating .................(No/Yes)    No
Cooling .................(No/Yes)    No
```

Table 8.34: Elements in page no. 77

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
NUMERICAL FIELDS
SPU DATA 01 F03
SPU DATA 02 F03
SPU DATA 03 F03
SPU DATA 04 F03
SPU DATA 05 F03
SPU DATA 06 F03
DYNAMIC TEXT FIELDS
SPU DATA 05 F03
SPU DATA 06 F03
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button marked “ “ in page 76; it shows the next part of the recipe needed to control the process.

PAGE 78 -> Recipe - Page 8



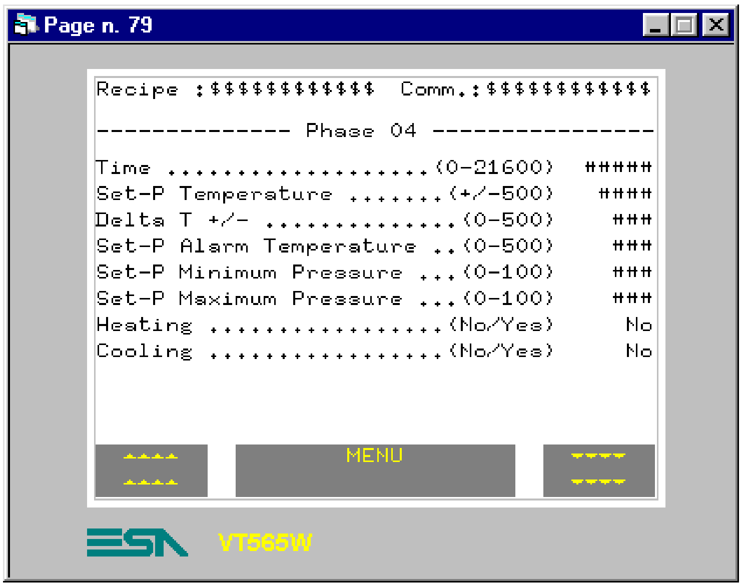
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 03 ----------------

Exhaust .................(No/Yes)     No
Mixer ...................(No/Yes)     No
Heating with Pr<Pr Min...(No/Yes)     No
Cooling with Pr>Pr Max...(No/Yes)     No
Test signal .............(No/Yes)     No
Test + confirm ..........(No/Yes)     No
B.C.F.V.F.S-P ...........(No/Yes)     No
End cycle ...............(No/Yes)     No
```

Table 8.35: Elements in page no. 78

TOUCH BUTTONS	
	RECIPE - MENU
	PAGE DOWN 1
	PAGE UP 1
DYNAMIC TEXT FIELDS	
	SPU DATA 09 F03
	SPU DATA 10 F03
	SPU DATA 11 F03
	SPU DATA 12 F03
	SPU DATA 13 F03
	SPU DATA 14 F03
	SPU DATA 15 F03
	SPU DATA 16 F03
ASCII FIELDS	
	RECIPE CODE
	RECIPE COMMENT
MULTILANGUAGE LABEL	

This page is displayed by pressing the touch button marked “ “ in page 77; it shows the next part of the recipe needed to control the process.

PAGE 79 -> Recipe - Page 9



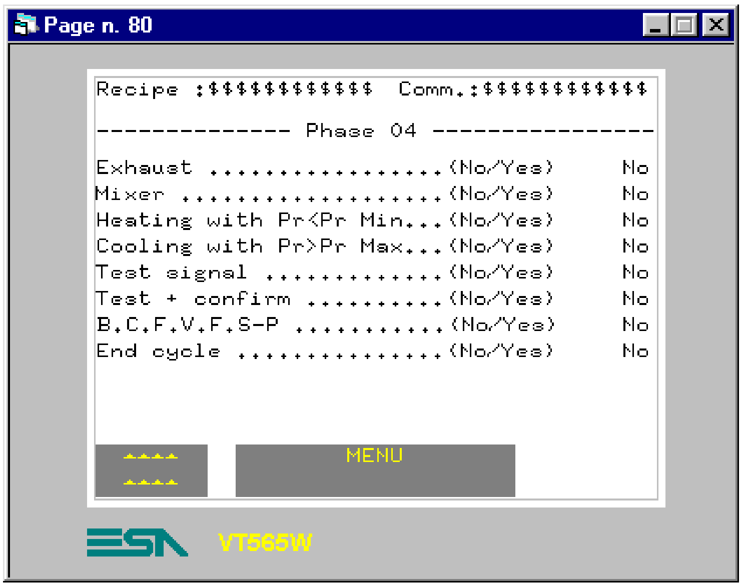
```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 04 ----------------

Time ..................(0-21600)  #####
Set-P Temperature .......(+/-500)  ####
Delta T +/- ..............(0-500)   ###
Set-P Alarm Temperature ..(0-500)   ###
Set-P Minimum Pressure ...(0-100)   ###
Set-P Maximum Pressure ...(0-100)   ###
Heating ................(No/Yes)    No
Cooling ................(No/Yes)    No
```

Table 8.36: Elements in page no. 79

TOUCH BUTTONS
RECIPE - MENU
PAGE DOWN 1
PAGE UP 1
NUMERICAL FIELDS
SPU DATA 01 F04
SPU DATA 02 F04
SPU DATA 03 F04
SPU DATA 04 F04
SPU DATA 05 F04
SPU DATA 06 F04
DYNAMIC TEXT FIELDS
SPU DATA 05 F04
SPU DATA 06 F04
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

This page is displayed by pressing the touch button marked “ ” in page 78; it shows the next part of the recipe needed to control the process.

PAGE 80 -> Recipe - Page 10



```
Recipe :$$$$$$$$$$$$  Comm.:$$$$$$$$$$$$

-------------- Phase 04 ----------------

Exhaust .................(No/Yes)     No
Mixer ...................(No/Yes)     No
Heating with Pr<Pr Min...(No/Yes)     No
Cooling with Pr>Pr Max...(No/Yes)     No
Test signal .............(No/Yes)     No
Test + confirm ..........(No/Yes)     No
B.C.F.V.F.S-P ...........(No/Yes)     No
End cycle ...............(No/Yes)     No
```

Table 8.37: Elements in page no. 80

TOUCH BUTTONS
RECIPE - MENU
PAGE UP 1
DYNAMIC TEXT FIELDS
SPU DATA 09 F04
SPU DATA 10 F04
SPU DATA 11 F04
SPU DATA 12 F04
SPU DATA 13 F04
SPU DATA 14 F04
SPU DATA 15 F04
SPU DATA 16 F04
ASCII FIELDS
RECIPE CODE
RECIPE COMMENT
MULTILANGUAGE LABEL

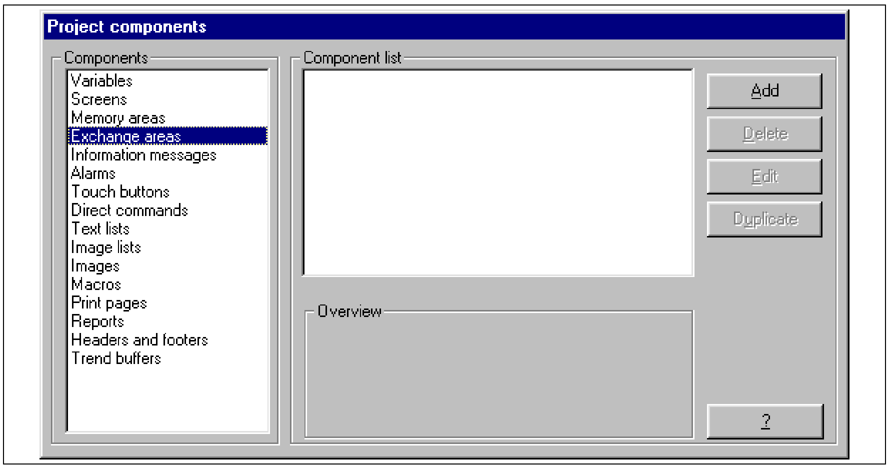
This page is displayed by pressing the touch button marked “---- ----“ in page 79; it shows the next part of the recipe needed to control the process.

Once all the pages have inserted, the user must assign to the touch buttons with the “Go to page” function the appropriate page associations, given that in the phase of creating the project almost all have been associated to page 1.

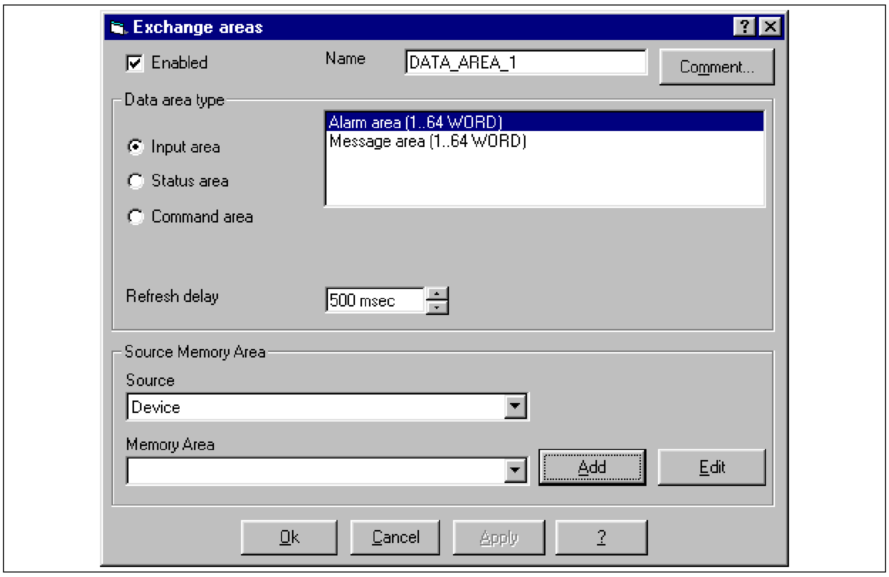
Data exchange area

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

Select the 🕮 Data exchange area (See “Chapter 4 -> Exchange areas“).



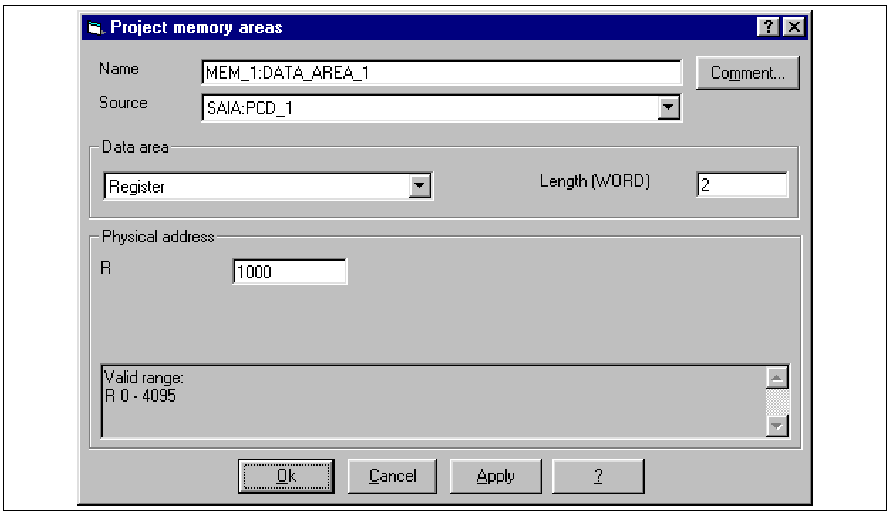
Click on Add.



[✓] Enabled
Name: DATA_AREA_1
Data area type: [● Input area] Status area, Command area
Alarm area (1..64 WORD) (selected)
Message area (1..64 WORD)
Refresh delay: 500 msec
Source: Device
Memory Area:

Set parameters as illustrated

Click on Add.



Name: MEM_1:DATA_AREA_1
Source: SAIA:PCD_1
Data area: Register
Length (WORD): 2
Physical address
R: 1000
Valid range:
R 0 - 4095

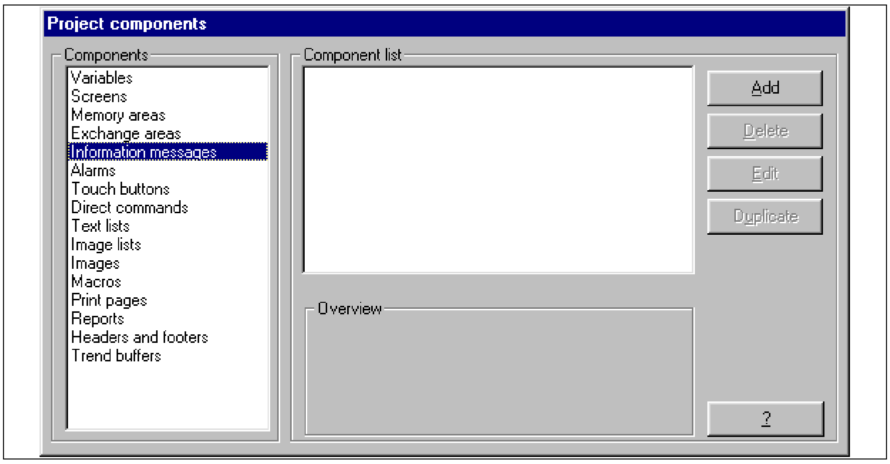
Set parameters as illustrated.

Click on OK.

Using the same method, insert all the memory areas necessary (See project offered as example).

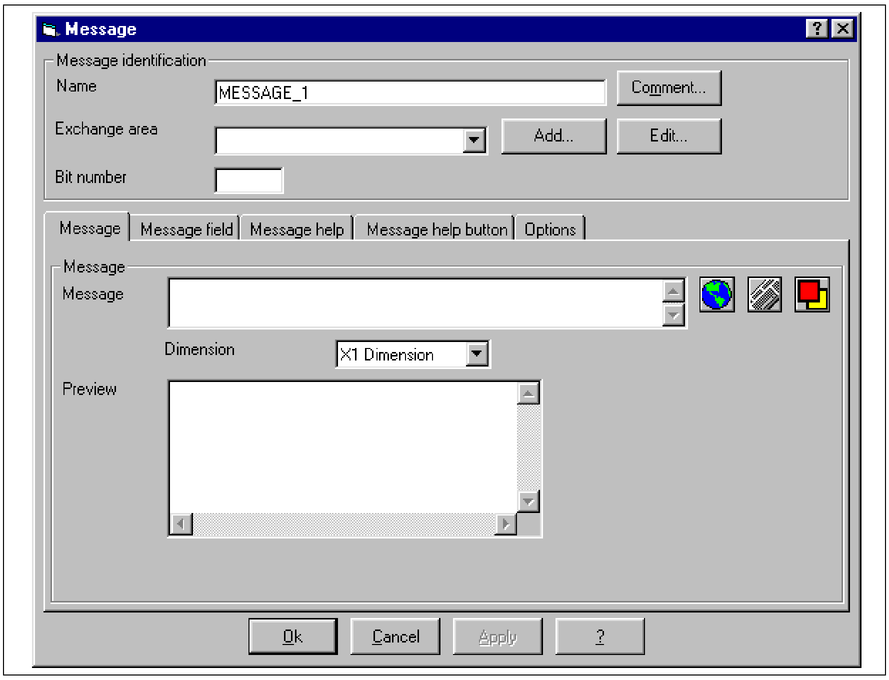
Information messages

In the example the need arises to inform the operator who is to use the VT that certain operations are not possible under certain conditions (See “Chapter 4 -> Information Messages“).



Select the 🕮 required.

Click on the ☐ Add.

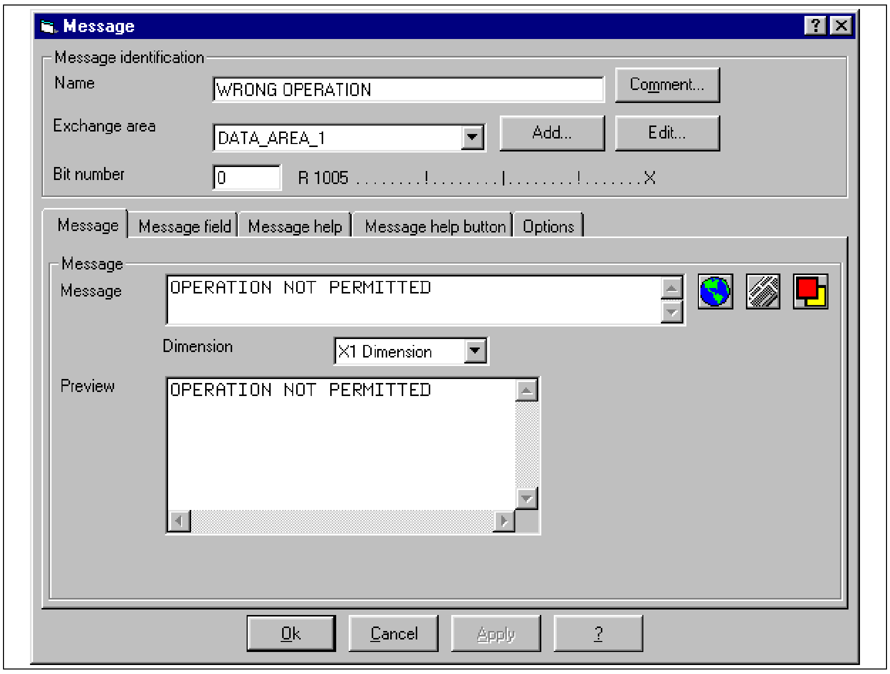


Name: MESSAGE_1
Dimension: X1 Dimension

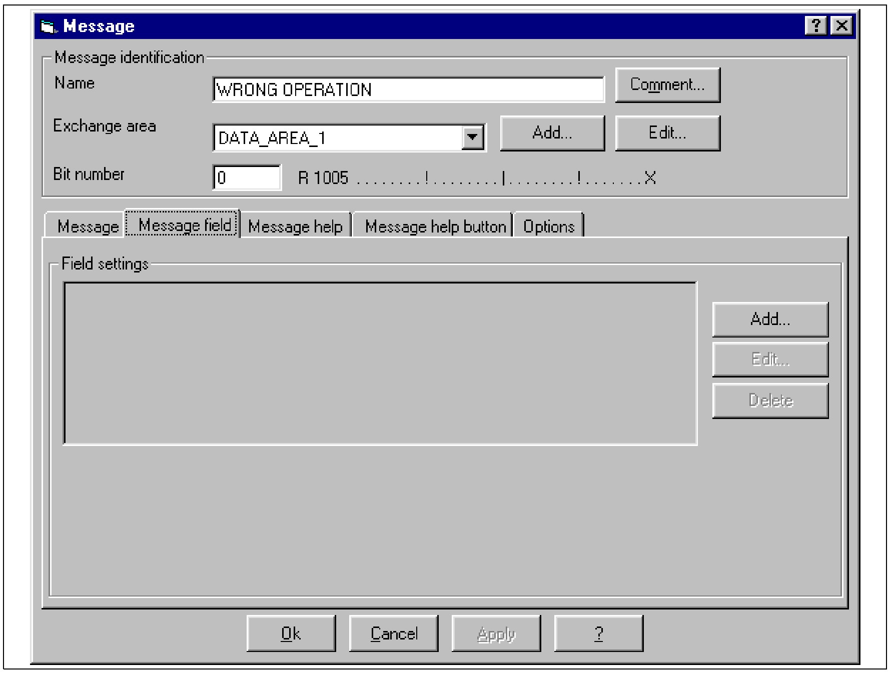
Insert the bit number to which the message must be assigned; assign the name to the message and edit the text.

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 field.



Name: WRONG OPERATION
Exchange area: DATA_AREA_1
Bit number: 0 R 1005 . . . . ! . . . . I . . . . ! X
Message: OPERATION NOT PERMITTED
Dimension: X1 Dimension
Preview: OPERATION NOT PERMITTED



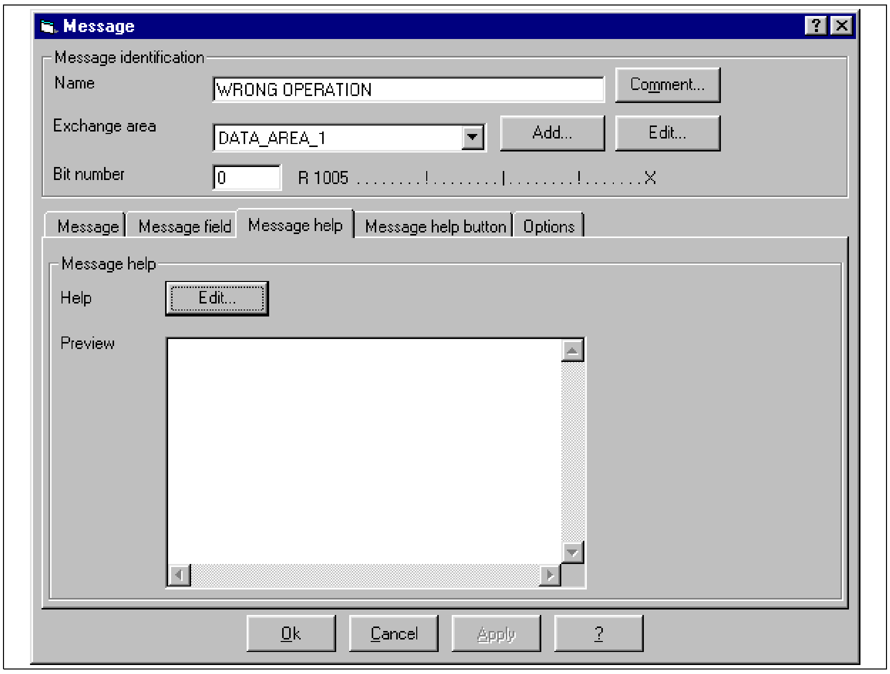
WRONG OPERATION

DATA_AREA_1

0 R 1005 . . . . ! . . . . | . . . . ! X

Given that there are no particular values and/or information on the variable related to the al message, so do not use this function.

Click on ☐ Help message.

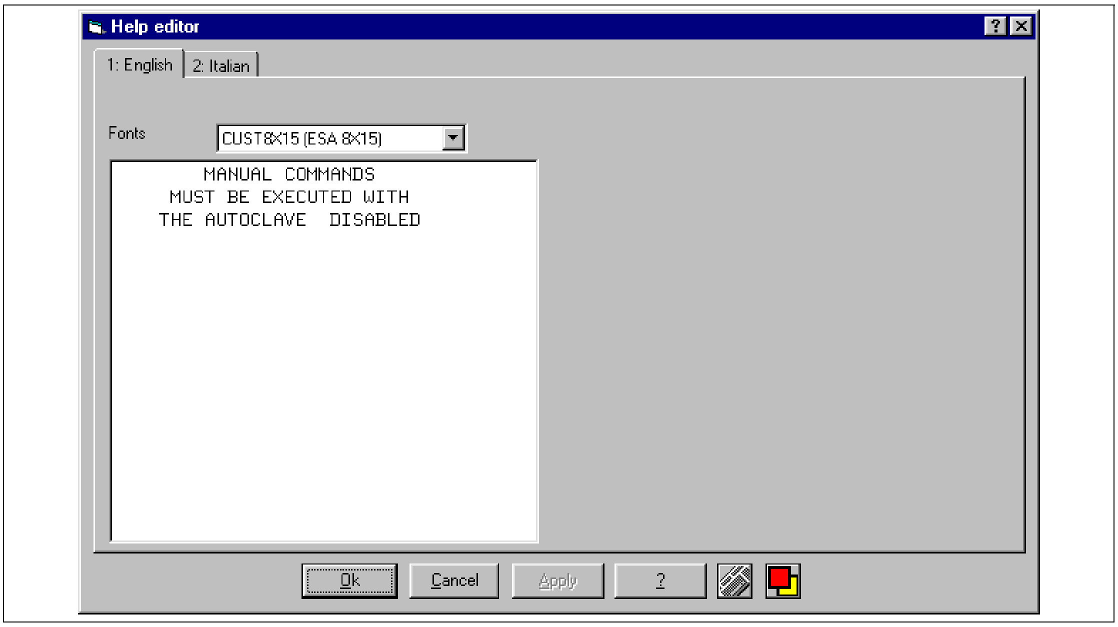


WRONG OPERATION

DATA_AREA_1

0 R 1005 . . . . ! . . . . | . . . . ! X

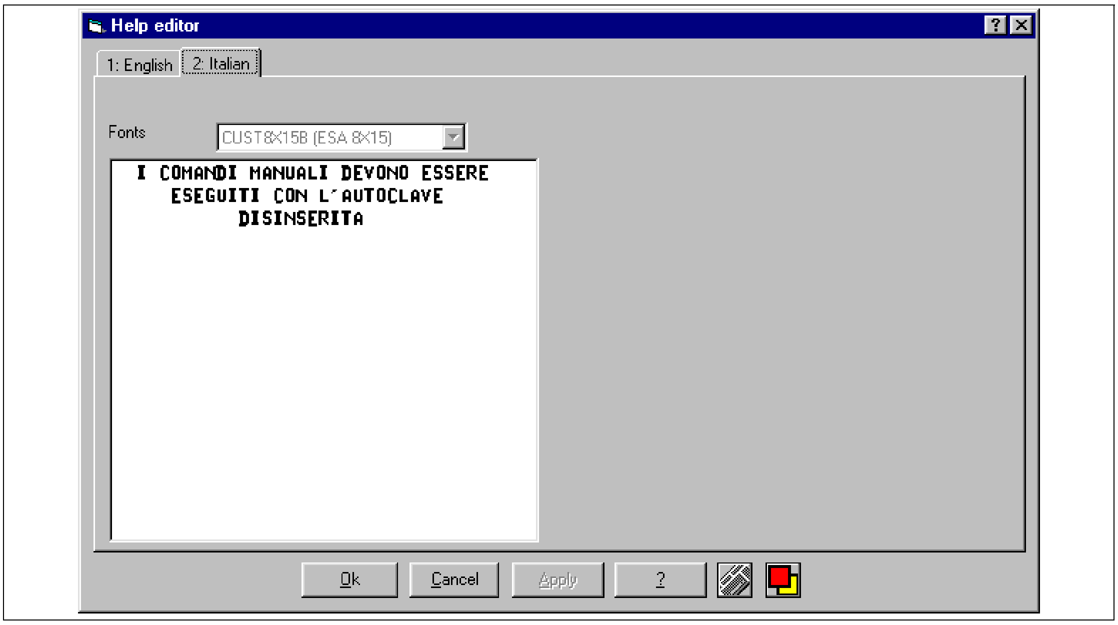
Click on ☐ Edit to insert the help message.



Fonts CUST8X15 (ESA 8X15)

MANUAL COMMANDS
MUST BE EXECUTED WITH
THE AUTOCLAVE DISABLED

Edit the text then click on Italian to insert the translation.



Fonts CUST8X15B (ESA 8X15)

I COMANDI MANUALI DEVONO ESSERE
ESEGUITI CON L'AUTOCLAVE
DISINSERITA

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

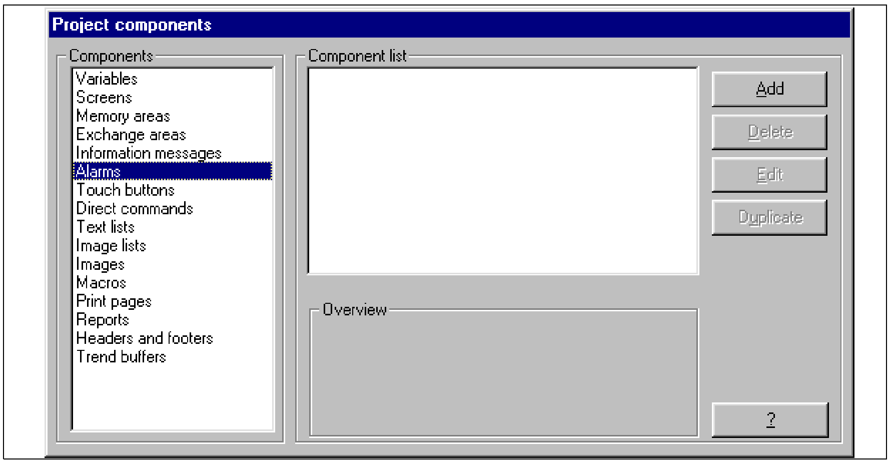
Using the same procedure insert all the messages (See project offered as example).

Alarms

Proceed to insert the alarms (See “Chapter 4 -> Alarms“).

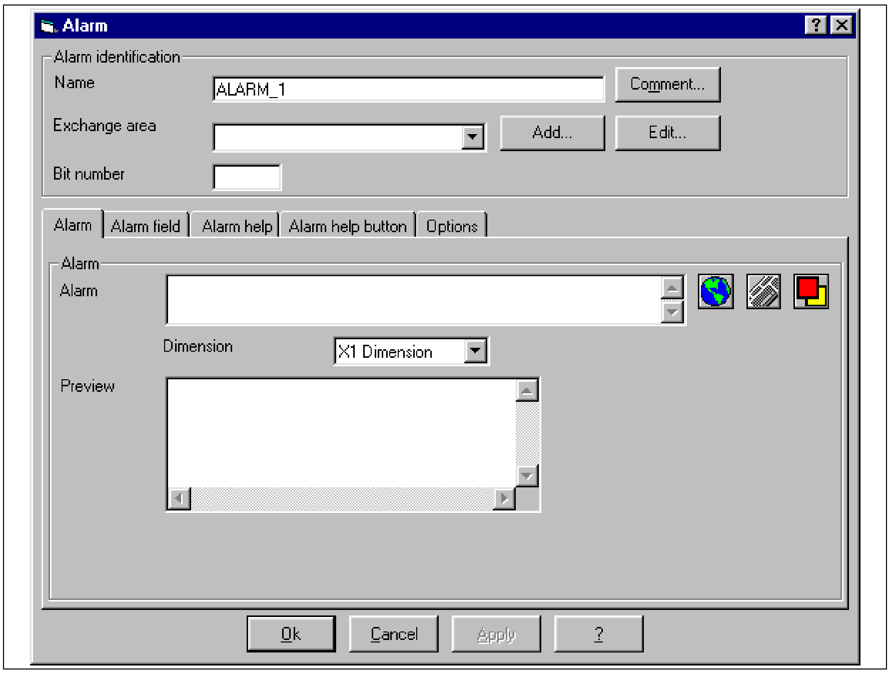
Select the 🕮 Alarms.

Click on ☐ Add.



Project components

Components: Variables, Screens, Memory areas, Exchange areas, Information messages, Alarms, Touch buttons, Direct commands, Text lists, Image lists, Images, Macros, Print pages, Reports, Headers and footers, Trend buffers



Alarm

Alarm identification

Name: ALARM_1

Exchange area:

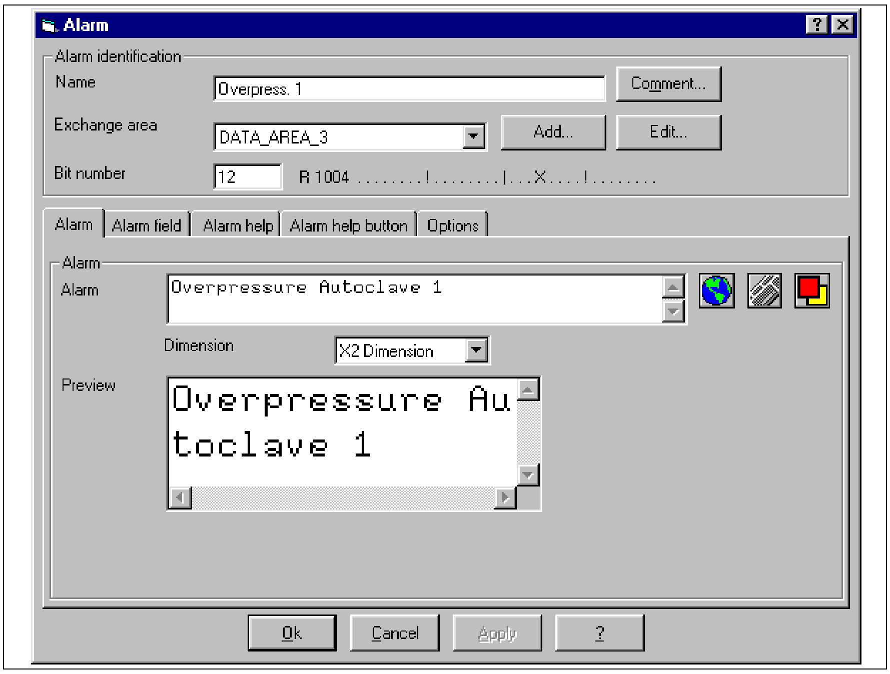
Bit number:

Dimension: X1 Dimension

Insert the bit number to which to assign the alarm; assign a name to the alarm and edit the text.

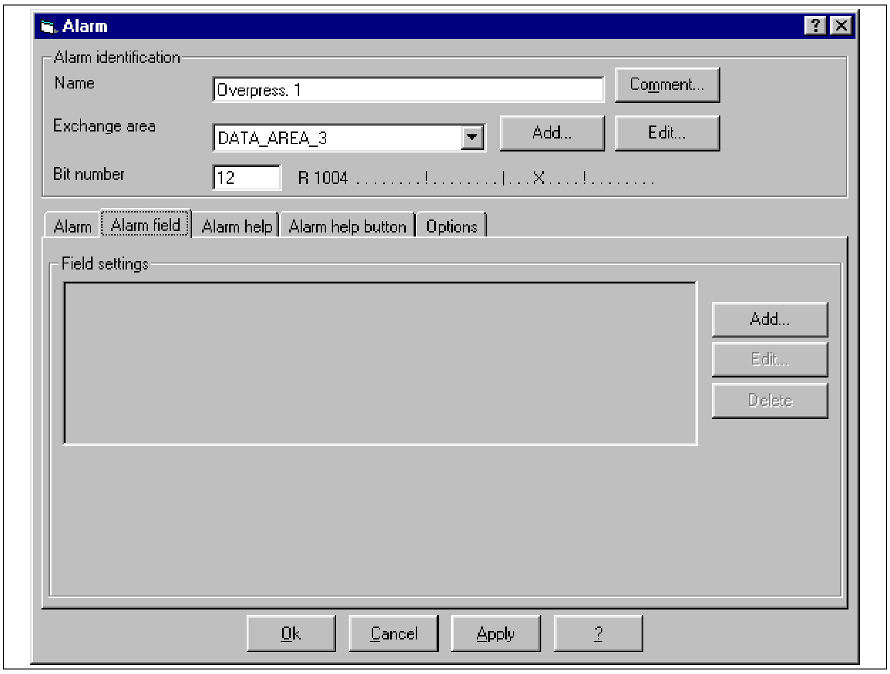
The comment is not assigned because no further information is needed to explain the function of the alarm.

Then choose the dimensions of the text for displaying the alarm. Once this has been done, click on ▢ Alarm field.

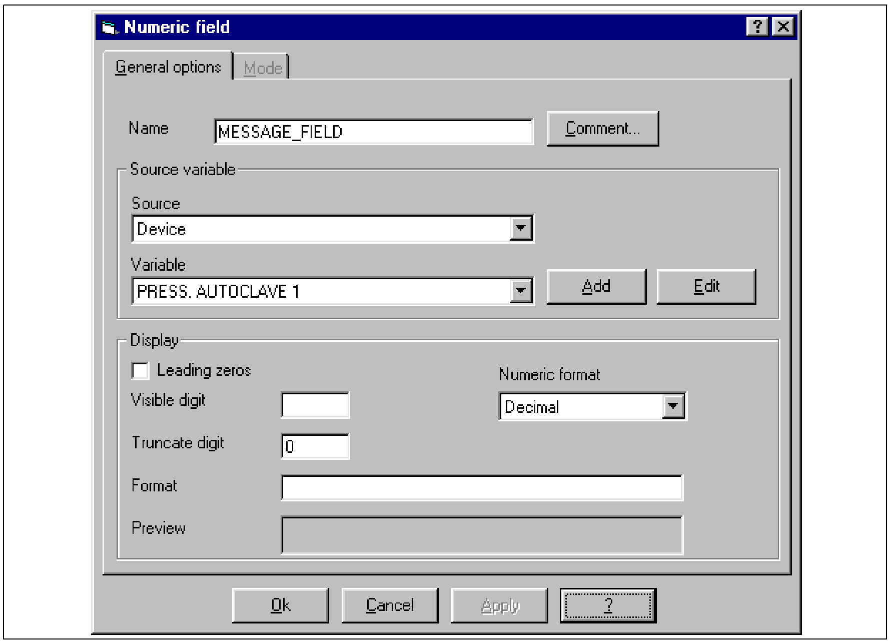


Name: Overpress. 1
Exchange area: DATA_AREA_3
Bit number: 12 R 1004 . . . . ! . . . . | . . . X . . . . ! . . . .
Alarm: Overpressure Autoclave 1
Dimension: X2 Dimension
Preview: Overpressure Autoclave 1

Click on Add.



Name: Overpress. 1
Exchange area: DATA_AREA_3
Bit number: 12 R 1004 . . . . ! . . . . | . . . X . . . . ! . . . .



Name: MESSAGE_FIELD

Source variable

Source: Device

Variable: PRESS. AUTOCLAVE 1

Display

Leading zeros [ ]

Numeric format: Decimal

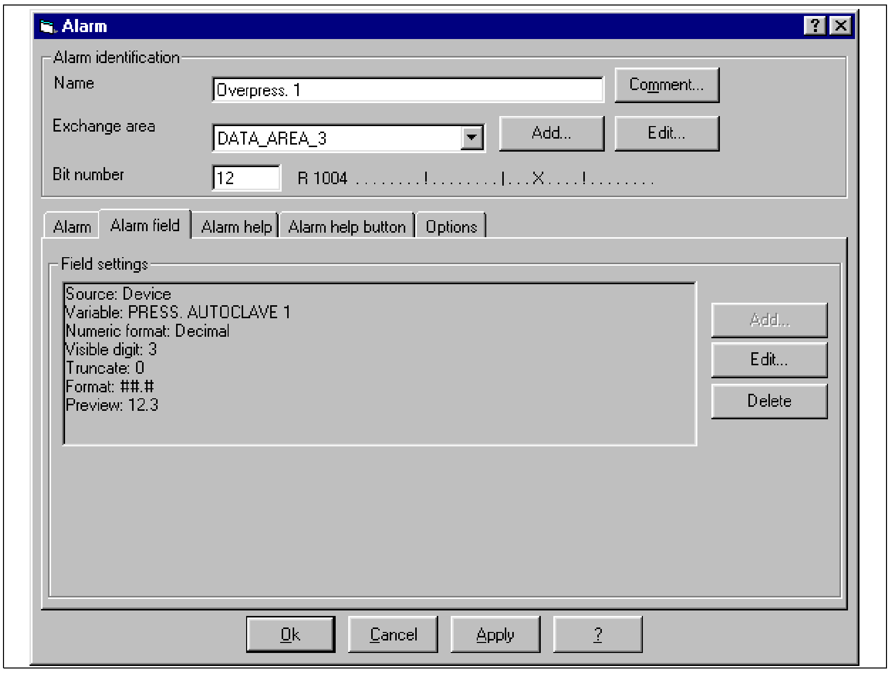
Visible digit:

Truncate digit: 0

Format:

Preview:

Compile as illustrated and click on OK.



Alarm identification

Name: Overpress. 1

Exchange area: DATA_AREA_3

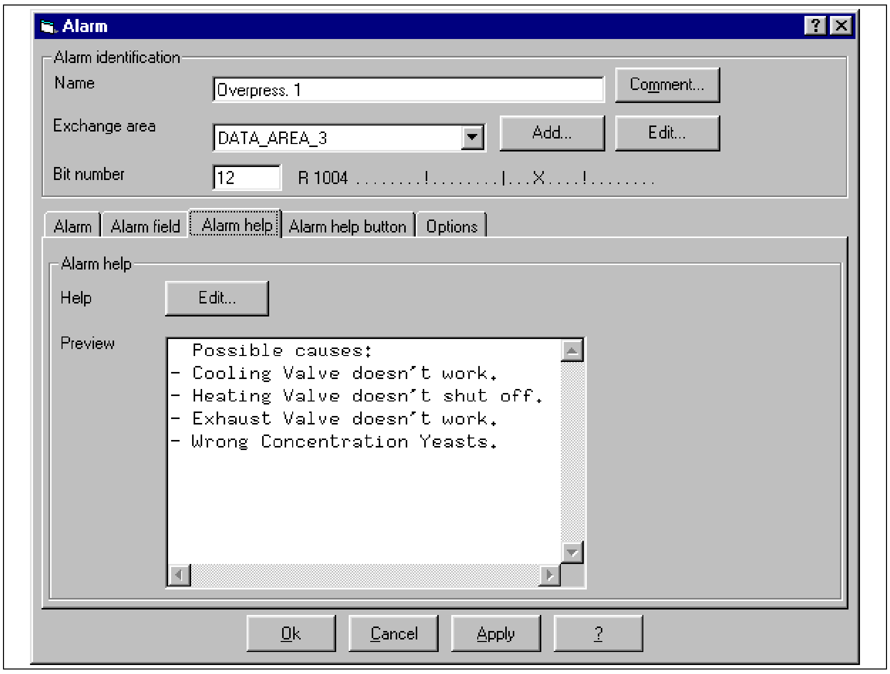
Bit number: 12 R 1004 . . . . ! . . . . ! . . . X . . . . ! . . . .

Field settings

Source: Device
Variable: PRESS. AUTOCLAVE 1
Numeric format: Decimal
Visible digit: 3
Truncate: 0
Format: ##.#
Preview: 12.3

The mask thus compiled is displayed.

Select the 🕮 Alarm help.

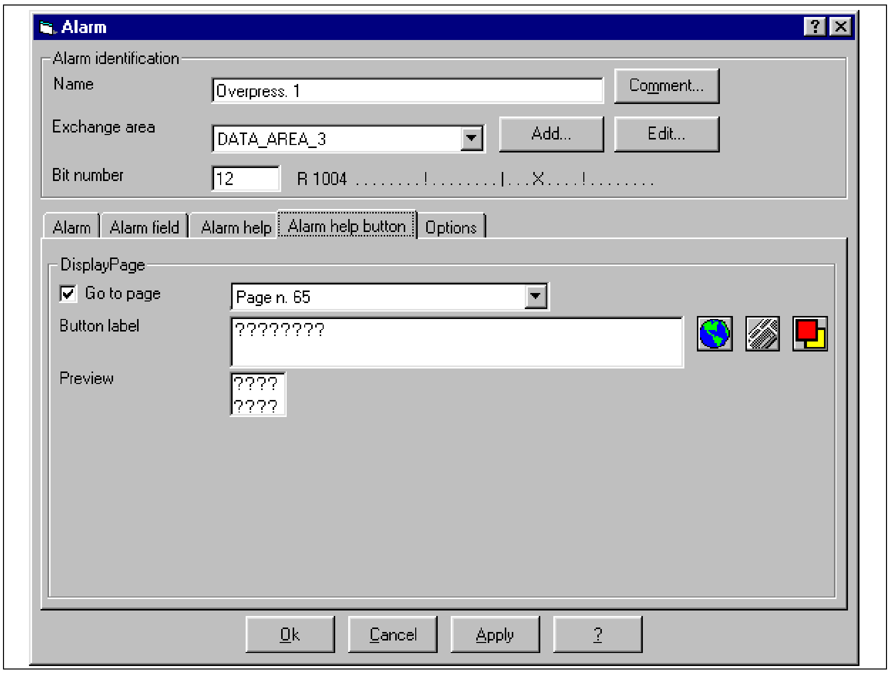


Alarm identification
Name: Overpress. 1
Exchange area: DATA_AREA_3
Bit number: 12 R 1004 ....!........|...X....!........

Alarm help
Help
Preview

```
   Possible causes:
-  Cooling Valve doesn't work.
-  Heating Valve doesn't shut off.
-  Exhaust Valve doesn't work.
-  Wrong Concentration Yeasts.
```

Insert text as illustrated and click on Alarm help button.

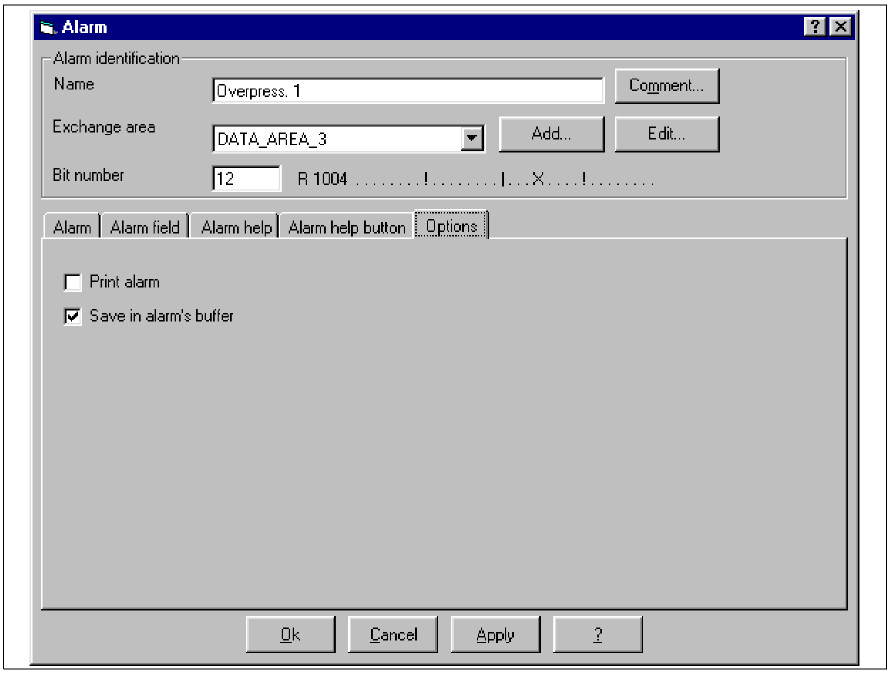


Alarm identification
Name: Overpress. 1
Exchange area: DATA_AREA_3
Bit number: 12 R 1004 ....!........|...X....!........

DisplayPage
☑ Go to page: Page n. 65
Button label: ????????
Preview: ????
????

Compile as illustrated, assigning the translation too.

Click on OK to accept the settings.



Alarm

Alarm identification

Name: Overpress. 1

Exchange area: DATA_AREA_3

Bit number: 12 R 1004 . . . . ! . . . . | . . . X . . . . ! . . . .

☐ Print alarm

☑ Save in alarm's buffer

Compile as illustrated.

Click on OK for the settings to be accepted.

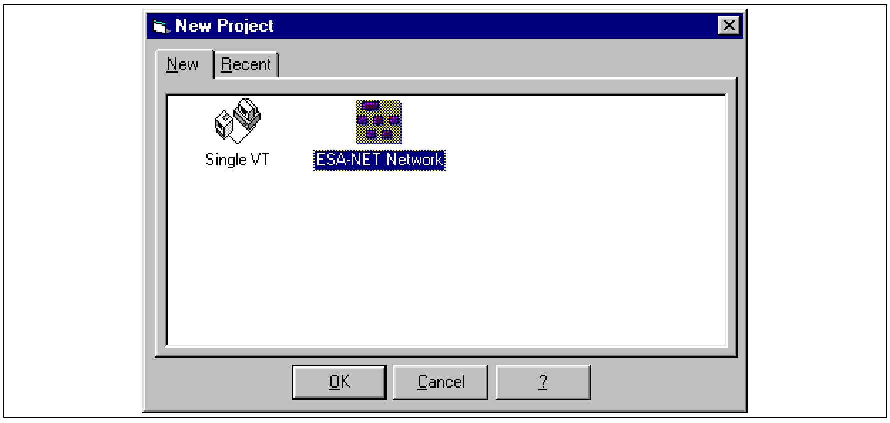
Now insert all the alarms the project needs. (See project offered as example).

The single VT project is now complete and must be compiled (See “Chapter 9 -> Compiling and transferring a project“); once it has been compiled without errors, the project must be saved.

Given that we need to create a project for an ESA-NET network, we must duplicate the single VT project and name it “SPUM565 for saia pcd (remote control).vts”

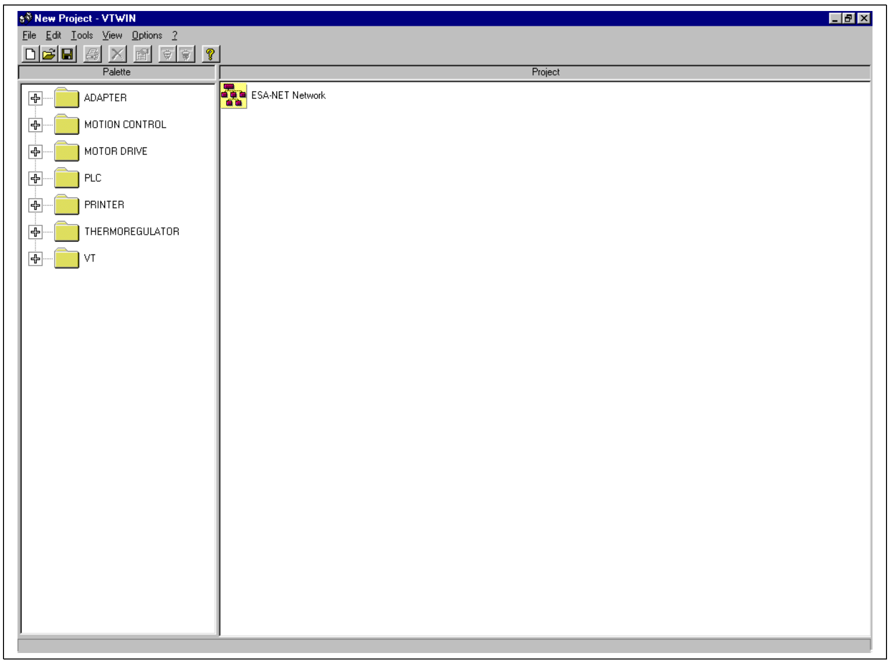
ESA-NET network

At this point, in order to create a network project, click on the 🕮 ***File > New*** (See “Chapter 5 -> New…“)



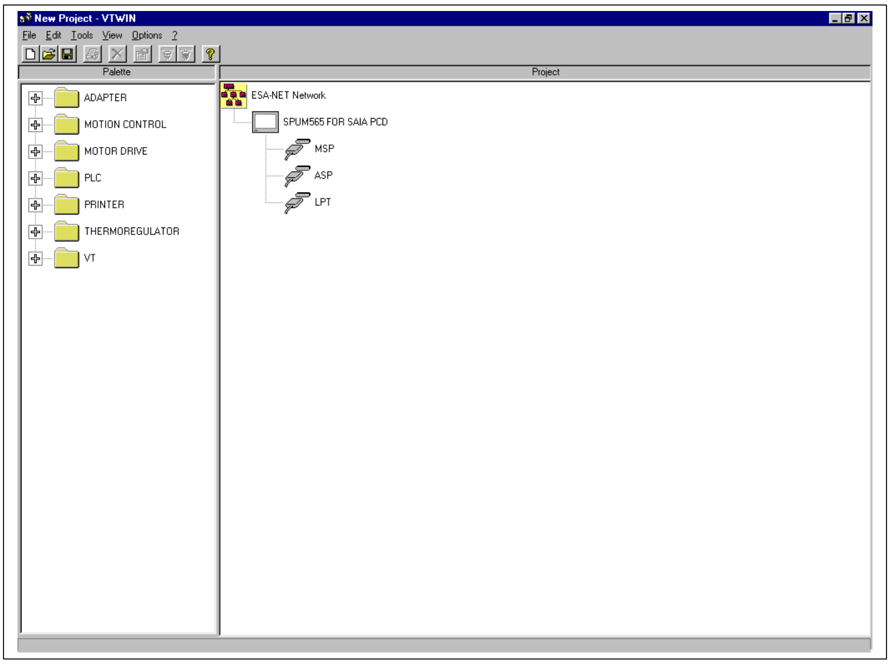
Select the icon ESA_NET network.

Click on the ☐ OK.



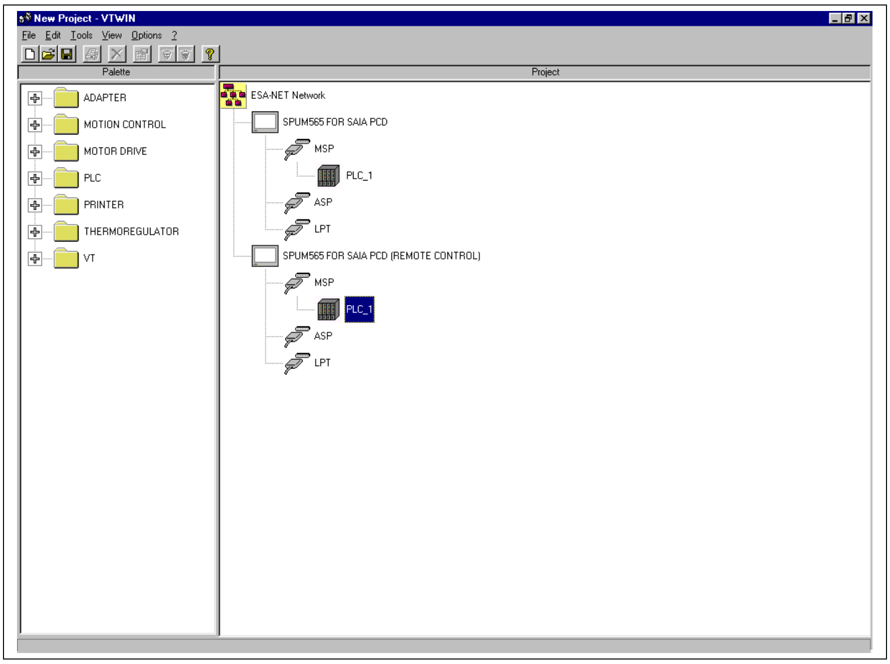
Select the icon on the Project side, then click on the 📖 Tools>Import... and choose the project “spum565 for saia pcd.vts” for the list.

Confirm with OK.



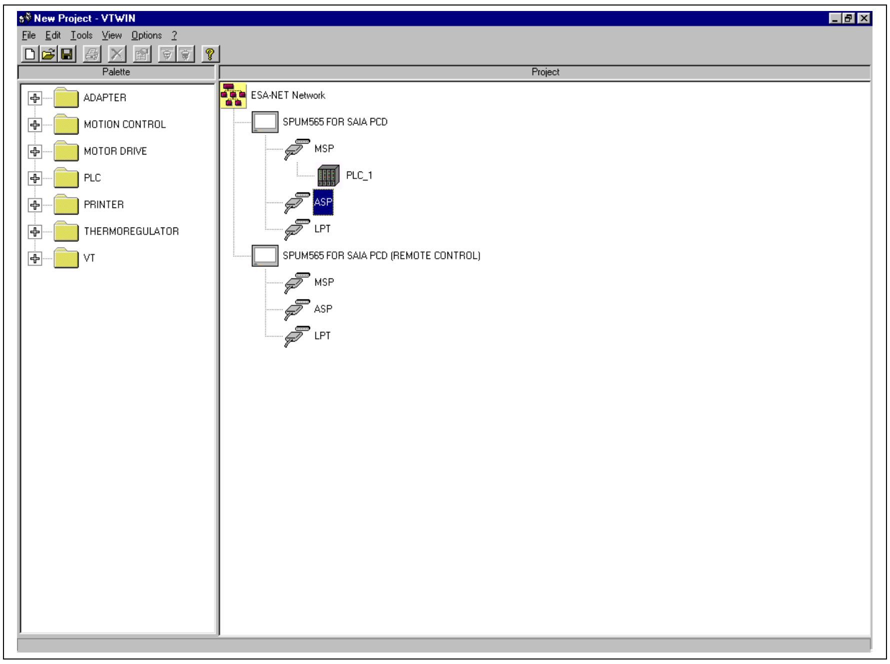
Repeat the operation and choose the project “spum565 for saia pcd (remote control).vts” from the list.

Confirm with OK.



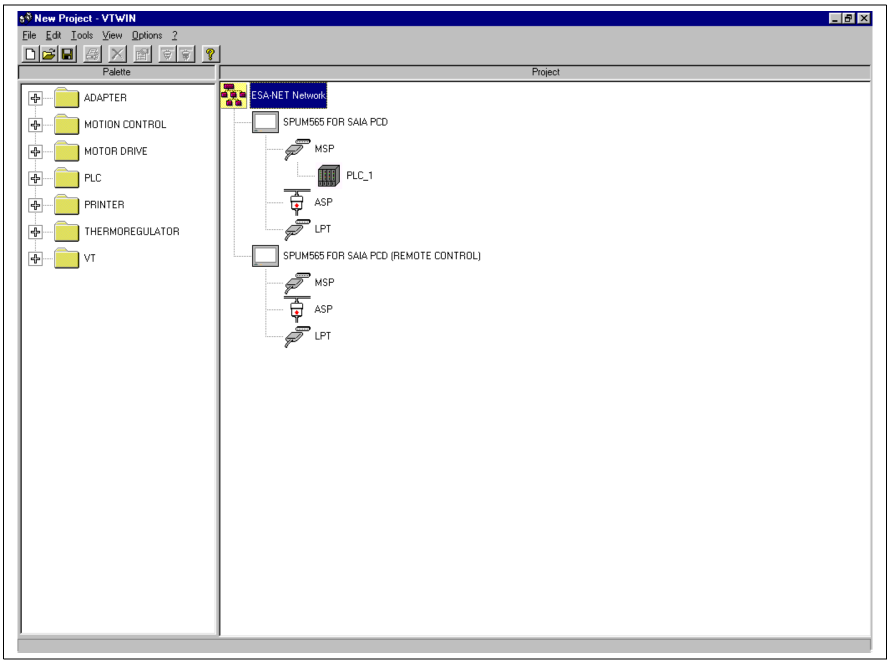
As illustrated, select PLC_1, then click on the Edit>Delete to remove the device.

Confirm with OK.

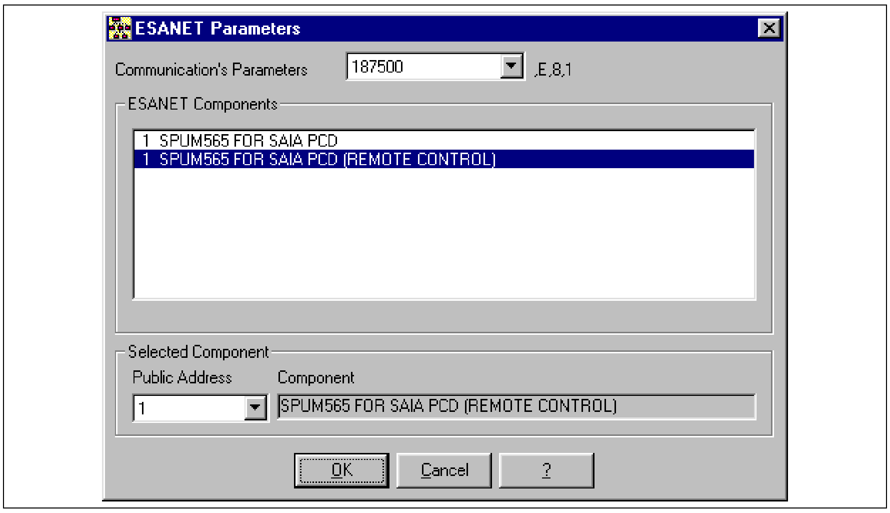


As illustrated, select the ASP port and click on . In this way the connection with the ESA-NET network is established. Repeat the operation for the ASP port of the other project too.

Confirm with OK.



Select as illustrated, then click on the 🕮 Edit>Property... The following window is displayed.



ESANET Parameters

Communication's Parameters 187500 ,E,8,1

ESANET Components

1 SPUM565 FOR SAIA PCD
1 SPUM565 FOR SAIA PCD (REMOTE CONTROL)

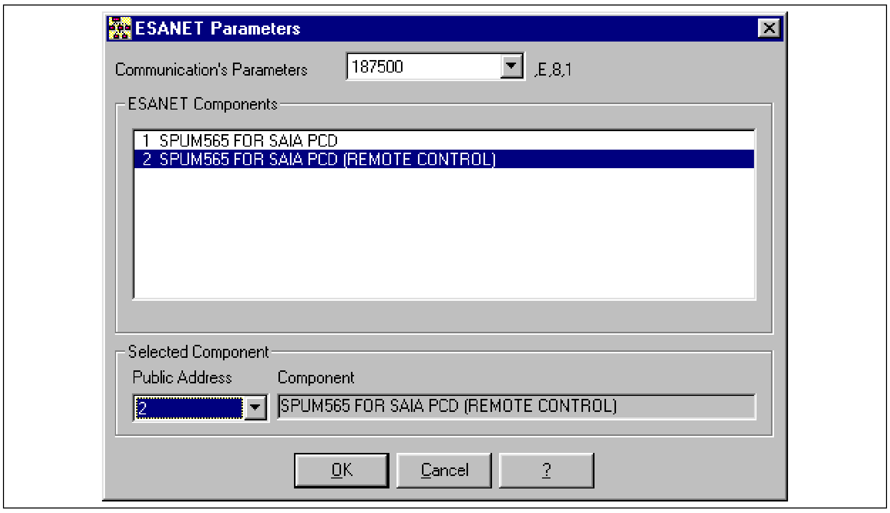
Selected Component

Public Address 1

Component SPUM565 FOR SAIA PCD (REMOTE CONTROL)

Select as illustrated, then change the Public address to 2.

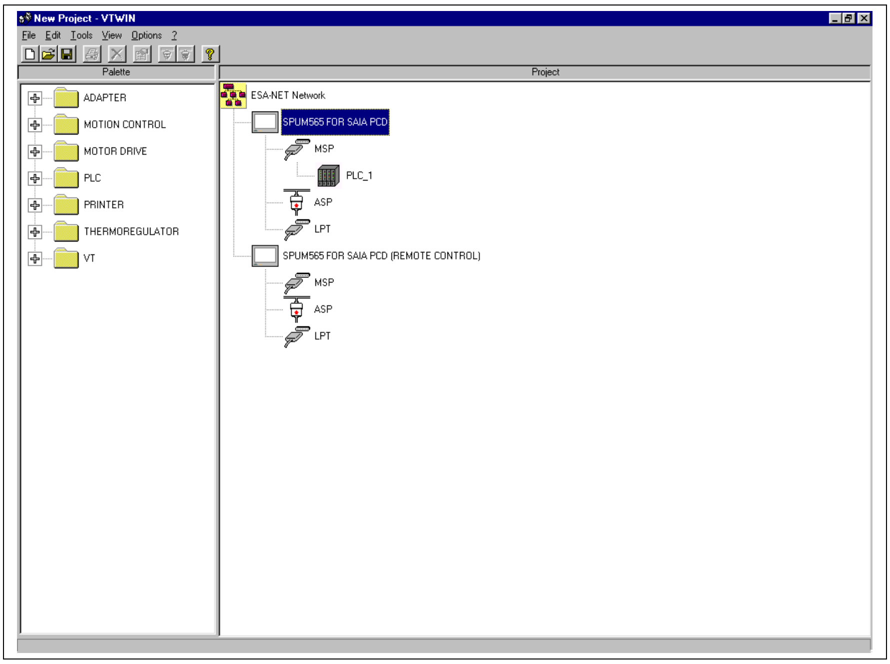
Confirm with OK.


ESANET Parameters

Communication's Parameters: 187500 ,E,8,1

ESANET Components
1 SPUM565 FOR SAIA PCD
2 SPUM565 FOR SAIA PCD (REMOTE CONTROL)

Selected Component
Public Address: 2
Component: SPUM565 FOR SAIA PCD (REMOTE CONTROL)

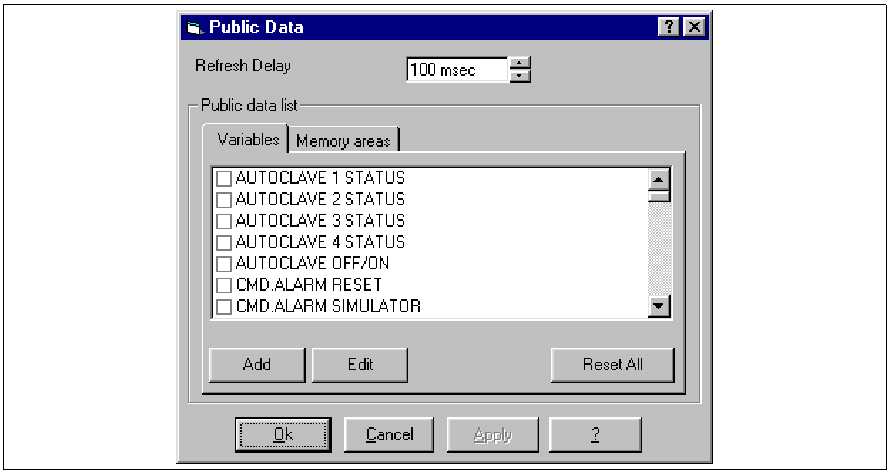

New Project - VTWIN

Palette: ADAPTER, MOTION CONTROL, MOTOR DRIVE, PLC, PRINTER, THERMOREGULATOR, VT

```
ESA-NET Network
  SPUM565 FOR SAIA PCD
    MSP
      PLC_1
    ASP
    LPT
  SPUM565 FOR SAIA PCD (REMOTE CONTROL)
    MSP
    ASP
    LPT
```

The network project has now been created; click on ***File > Save*** to confirm the settings.

Open the project highlighted in the figure above to carry out changes needed to work in an ESA-NET network context; once the project has been opened, click on ***Configure > Public data*** to have the following mask appear.

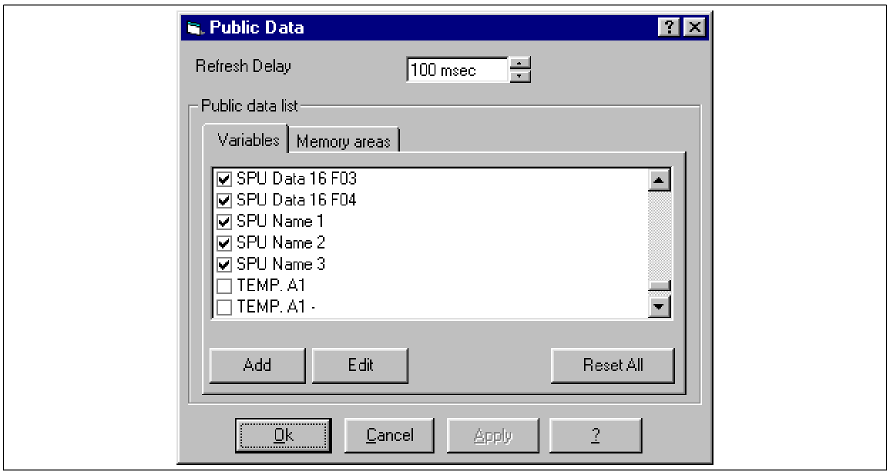


Refresh Delay: 100 msec

- [ ] AUTOCLAVE 1 STATUS
- [ ] AUTOCLAVE 2 STATUS
- [ ] AUTOCLAVE 3 STATUS
- [ ] AUTOCLAVE 4 STATUS
- [ ] AUTOCLAVE OFF/ON
- [ ] CMD.ALARM RESET
- [ ] CMD.ALARM SIMULATOR

Identify the variables to be exported and select all of them.

To know which variables must be declared as public, see the project offered as an example.



Refresh Delay: 100 msec

- [x] SPU Data 16 F03
- [x] SPU Data 16 F04
- [x] SPU Name 1
- [x] SPU Name 2
- [x] SPU Name 3
- [ ] TEMP. A1
- [ ] TEMP. A1 -

Click on Ok to confirm.

The project is now complete and ready to function in an ESA-NET network context. It remains only to compile and transfer it (See “Chapter 9 -> Compiling and transferring a project“).

Make any necessary changes to the other project too (See project offered as example) then compile and transfer.

The project is now finished.

Chapter 9 Compiling and transferring a project

This chapter consists of a total of 6 pages.

Compiling a project

Compilation is a method of creating a file automatically in a format the VT 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.

Configuring the language to be used (providing one is present):

> Allows the user to define which set of languages to transfer to the terminal (see “Chapter 6 -> Configuring languages“).

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.

Transfering the project

Once the project has been compiled correctly, it must be transferred to the operator terminal. The transfer may use the serial port of the PC directly connected to the VT or a modem.

⚠ **Transfer by Modem is possible only with VTs and not with ADTs.**

To activate serial transfer click on ***Tools > Download project***, while to activate modem transfer click on ***Tools > Download with Modem***. In both cases, if the project has not already been compiled, the system automatically displays the compilation window with the above-listed parameters, otherwise the download window is displayed in which the PC-VT communication parameters can be selected.

⚠ **It is wise to open the VTWIN transfer window BEFORE preparing the terminal for reception (see Hardware Manual).**

⚠ **If there are ADT connected to the terminal (see Hardware Manual), the project must be transferred first to the VT and then to the adapter.**

⚠ **What has been explained relates exclusively to VT projects and not for ADT projects directly connected to a PC. For ADT projects directly connected to a PC, it is the user application package that is used for the transfer (see** "Chapter 18 -> User application package"**).**

⚠ **If the transfer is carried out via Modem check that the modem directly connected to the VT is correctly set (see Hardware Manual** "Chapter 31 -> Settings for the Modem"**).**

⚠ **For an MPI transfer via modem to be possible there has to be a special unit in the network to be accessed that allows a connection between the modem and the MPI network.**

⚠ **Once a project has been transferred, the VT will automatically switch to reception mode for successive transfers when using an Ethernet or MPI port.**

⚠ **Projects can be transferred in MPI mode using a special board (E.g. SIEMENS CP5611 or CP5511) or by using the PC/MPI CABLE SIEMENS 6ES7901-2BF00-0AA0 adapter. Power musy be supplied to the latter by its own supply unit or by a S7-300 PLC. If transfers are not in the context of a network but directly from PC to VT, the "PC/PC as sole bus master" parameter must be set in the STEP 7 (for further details see STEP 7 manual).**

To prepare the VT for transferring the project, see the relevant Hardware Manual.

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

Serial connection:

> Use this to choose the communication port used by the PC. (The Modem is in any case seen as a communication port).

Baud rate:

> Allows you to select the transfer speed of data passing between the PC and the VT in the one case and between the PC and the Modem in the other.
>
> ⚠ **Using the modem, the transfer speed is fixed at 38400bit/sec.**

Ethernet (only with Ethernet VTs):

> Allows this port to be selected for transfer.

IP address of terminal (only with Ethernet VT):

> Allows IP address to be defined of terminal to which to transfer the project.

MPI (only with Step 7 installed in PC):

> Allows this port to be selected for transfer.

MPI address of terminal (only with Step 7 installed in PC):

> Allows the MPI address to be defined of the terminal to which to transfer the project.

Modem (only with Step 7 and Modem installed in PC):

> Makes it possible to define/activate a transfer of the project to a terminal connected in an MPI network.

Update terminal:

> Allows you to enable the transfer of the project to the terminal. This 🕮 is activated automatically.

Update terminal Fw:

Use this to specify whether, during the transfer, the VT firmware is also to be transferred.

This allows you to force the loading of the firmware. When VTWIN becomes aware that the firmware available is more recent than that contained in the VT, it automatically updates it. Normally this 📖 is not activated because, for one thing, the transfer times become much longer. It can be activated in the event that there are doubts as to whether the VT is functioning correctly.

Update adapters:

Allows you to enable the transfer of the project to the adapter. This 📖 is activated automatically.

Update adapters Fw:

Use this to specify whether, during the transfer, the adapter firmware is also to be transferred.
This allows you to force the loading of the firmware. When VTWIN becomes aware that the firmware available is more recent than that contained in the adapter, it automatically updates it. Normally this 📖 is not activated because, for one thing, the transfer times become much longer. It can be activated in the event that there are doubts as to whether the adapter is functioning correctly

Telephone number (only for transfers using the modem):

Allows you to enter the telephone number to establish a connection with a remote modem.

⚠ **The dialing format for the telephone number and the special characters depends on which modem is being used.**

Dialing (only for transfers using a modem):

Allows you to define whether the telephone number dialing should be By Tone or By Impulse. The choice should be based on the nature of the telephone line being used.

Attempts (only in the case of transfers using a modem):

Allows you to define how many attempts at reconnection should be made if the call fails.

You are advised not to set the number of attempts too high (recommended 1 - 5). If no connection is established, check the telephone number, connections and modem parameters.

User Name (only for transfers with modem for MPI):

Makes it possible introduce the user name to access the remote connection (for further details see user documentation related to the adaptor unit used).

Password (only for transfers with modem for MPI):

Makes it possible introduce the password to access the remote connection (for further details see user documentation related to the adaptor unit used).

Chapter 10 Creating and printing documentation

This chapter consists of a total of 4 pages.

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.

The print program is activated; this contains a series of allowing the definition of all the necessary print parameters, which are then compiled and/or selected.

⚠ **VTWIN simply uses the drivers of the printers that are selected, without adding any particular control operation; print on file, print front/back etc. depend, therefore, exclusively on the printer selected.**

Name of printer:

> Used to specify which printer to choose from those available (installed on the personal computer) to print the project. Click on Settings....

Printer Port:

> Display the port or path of the printer selected.

Name of format:

> Used to select which print format to use from those available.

Description of format:

> Shows what is needed for the format displayed and/or chosen to be printed.

Preview:

By clicking on ▢ Preview a special function is accessed allowing you to see what is to be printed and how which will vary according to the model selected. This 🗀 contains in its turn two menus:

Control

Allows you to browse the print preview.

View

Allows you to select the Layout (1 or 2 pages) of the print preview).

Edit:

By clicking on ▢ Edit you call up a mask containing all the parameters needed for the print pattern. This mask consists of several 🗀 described below.

Name:

Name of pattern in edit phase; aso the name can be edited.

🗀 Optional sections:

This allows the type of information to be changed that you intend to print (e.g.. only pages or only variables or variables and pages etc.). The order in which these items of information are to be printed can also be modified. (Print order is the order in which the information is inserted in the list.)

Sections available:

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

Sections selected:

This is the list of items of information to be printed. By selecting the 🕮🕮🕮 one at a time and clicking on ▢ Configure you specify the elements to be printed from the 🕮 in question.

⚠ **If when the 🕮 is selected, the ▢ Configure is not activated, this means that no customizing is possible. (E.g. when Variables is selected the ▢ is not activated, meaning**

that the variables will be printed in the mode as planned and that in VTWIN they not modifiable by the user).

Global settings:

Used to make global settings for the various types of documents to be printed (cover, index etc.).

Include cover:

Allows the user to define whether a cover is required or not. If so, it can be chosen from the list, assuming that at least one cover has been created. To do this click on New...

Index:

Define whether an index is required.

Project information:

Define whether to have project information.

Comments in all sections:

Defines whether to have comments in all sections.

Page settings:

Used to define the Layout of the page. (Applicable to all pages).

Margins:

Used to define the page margins.

Header:

Allows the user to write a header line and decide whether to print it or not.

Footer:

Allows the user to write a footer line and decide whether to print it or not.

Page numbers:

Determines where to put the page number.

Chapter 11 Creating a back-up of the project

This chapter consists of a total of 2 pages.

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 every so often to make for reasons of safety a back-up copy of what you have created and/or edited.

It is important to save the .VTS file (Single VT project) or .VTN file (ESA-NET network project); all the files necessary for the project itself can be obtained from this file.

If you lose the source project (.VTS or .VTN), the information contained in it will be lost definitively; a possible recover from the VT 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 12 Defining the fonts

This chapter consists of a total of 6 pages.

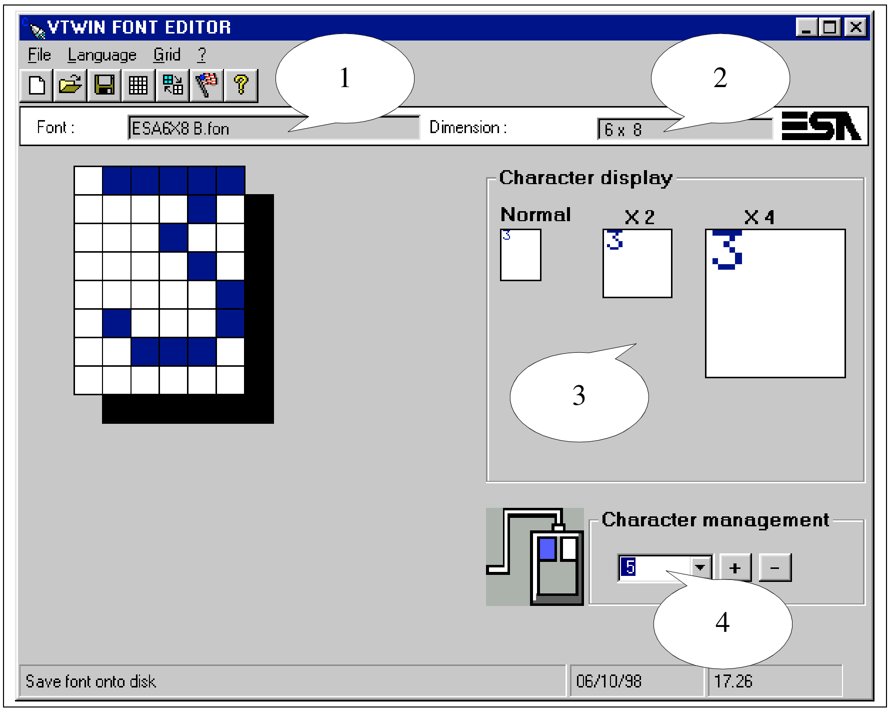
VTWIN contains a program allowing the character fonts to be edited and/or created.

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.

In the case of graphic panels 255 characters (from 1 to 255) can be modified.

When a font is used it is saved along with the project; this makes it possible to transfer the project to other PCs without encountering any problems related to representation. This is true both for text panels and for graphic panels.

To call up the program click on ***Tools* > *Font Editor*** or click on ***Start* > *Programs* > *VTWIN* > *Font Editor***; the main mask appears.



Font : ESA6X8 B.fon
Dimension : 6 x 8
5
Save font onto disk
06/10/98
17.26

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

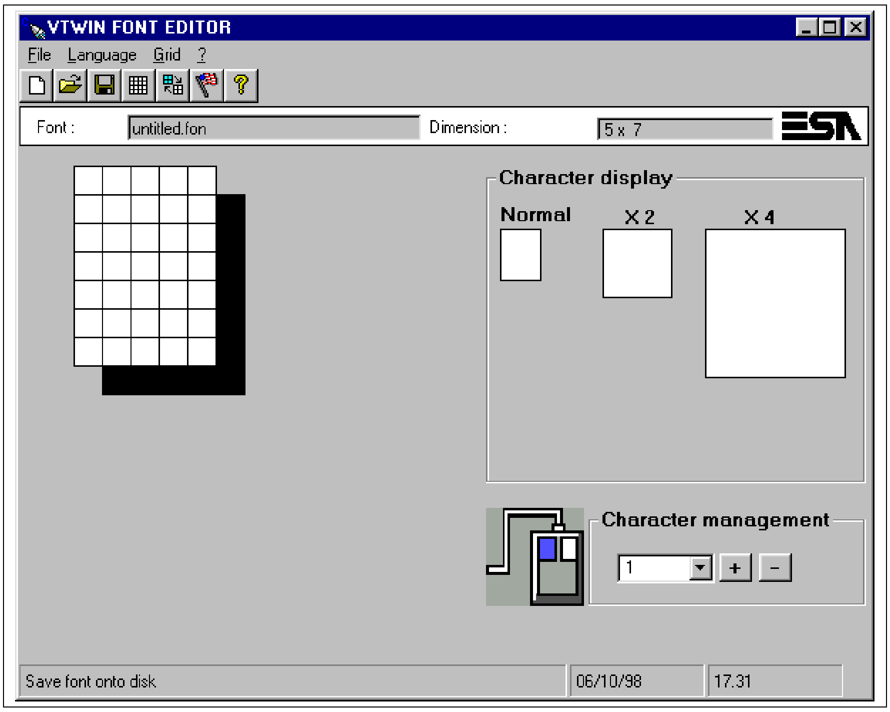
The table shows all the icons of the menus together with their meanings.

Table 12.1: List of icons used in VTWIN Font Editor, menu attribution and meaning.

Tool Bar	Pulldown menu	Action
🗋	***File > New***	Creates a new font.
📂	***File > Open font***	Opens an already existing font.
💾	***File > Save font***	Save a font on disk.
▦	***Grid > Clear***	Deletes the content of the grid.
▨	***Grid > Invert***	Inverts the content of the grid. (White becomes black and vice versa).
⚑	***Language***	Allows the language of the program to be selected.
?	***?***	Calls up the Font Help.

Personalizing a Font

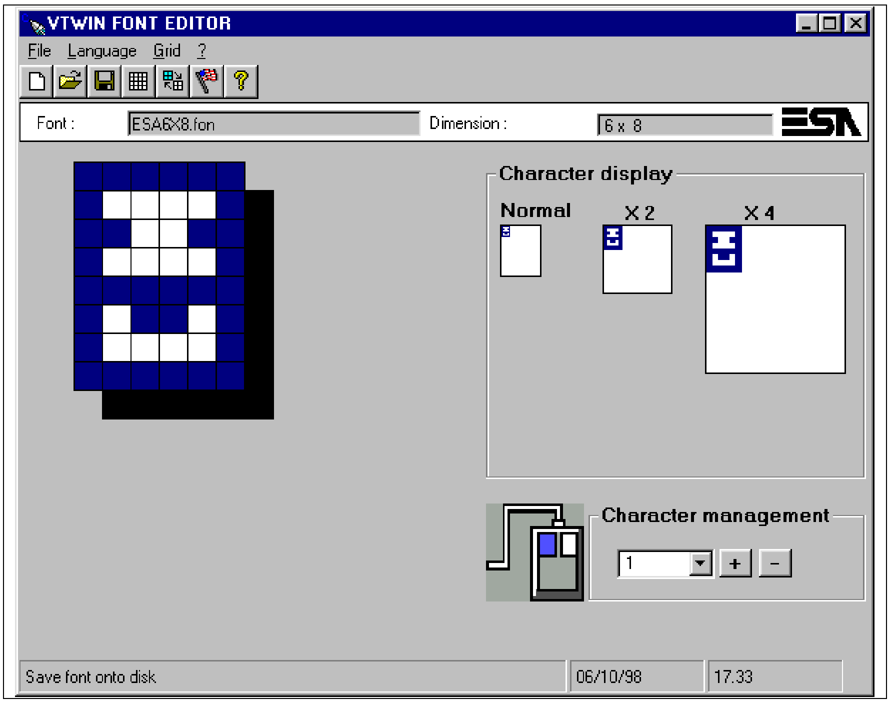
Below is an example of personalizing a project font by generating a new one, the font CUST6X8.fon.



Font : untitled.fon
Dimension : 5 x 7

Save font onto disk 06/10/98 17.31

Click on ***File > Open font***



Font : ESA6X8.fon
Dimension : 6 x 8

Save font onto disk 06/10/98 17.33

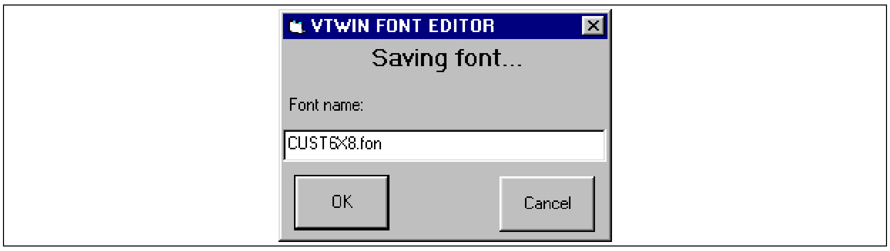
Select the font as illustrated.

Click on OK.

Click on ***File > Save font***

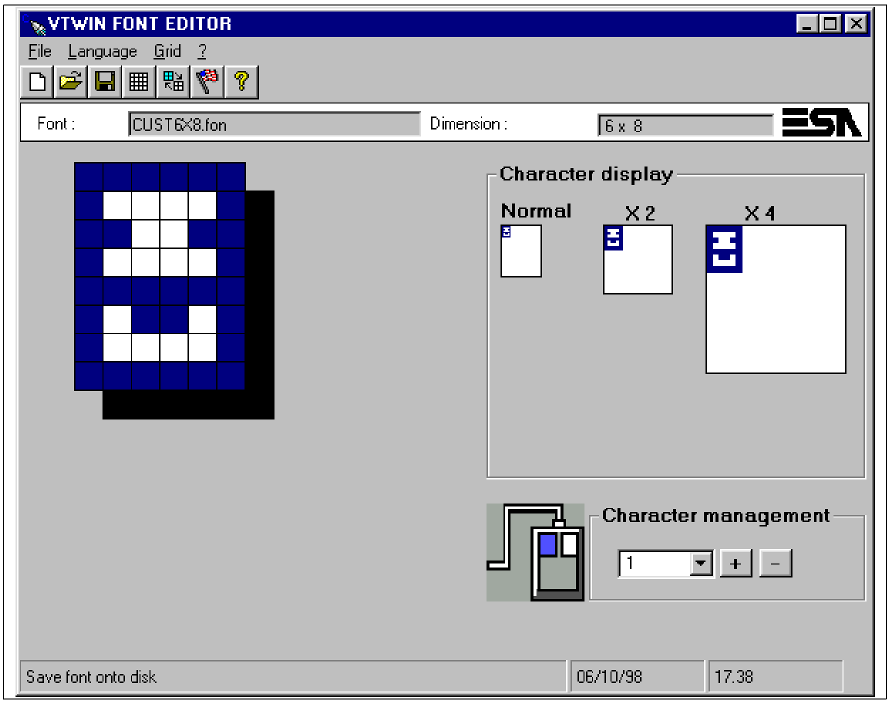
Assign a name as illustrated.

Click on OK.



CUST6X8.fon

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.

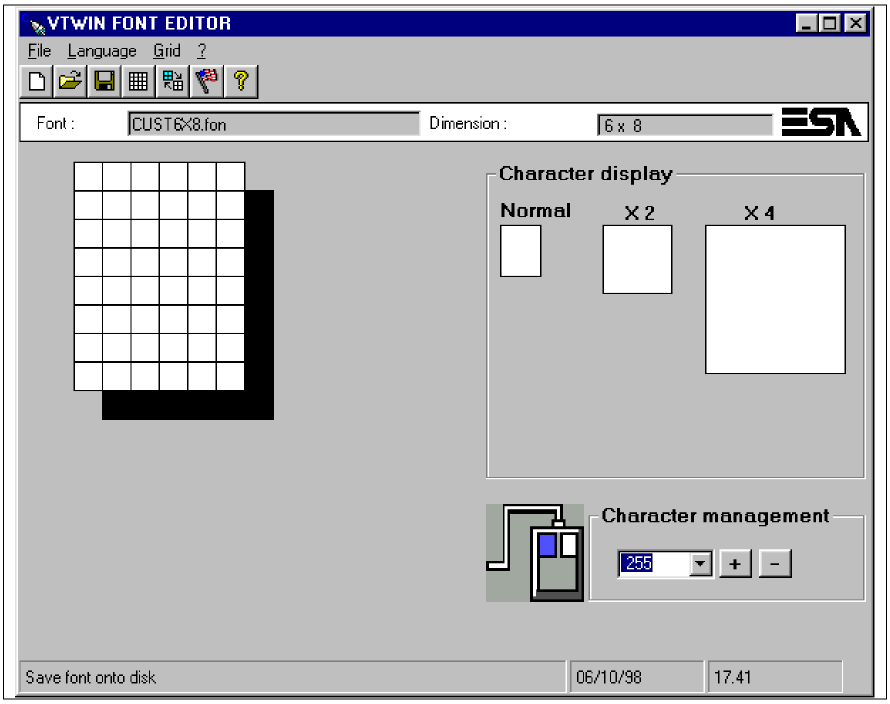


Font : CUST6X8.fon

Dimension : 6 x 8

Character management: 1

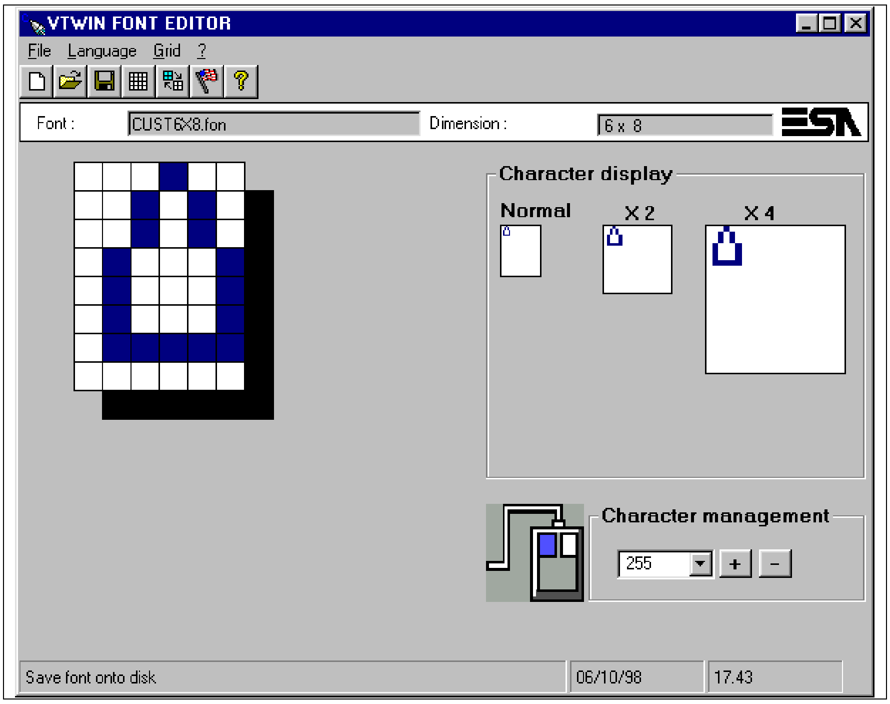
Save font onto disk | 06/10/98 | 17.38



Font : CUST6X8.fon
Dimension : 6 x 8
255
Save font onto disk
06/10/98
17.41

Set "Character management" arbitrarily on 255.

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



Font : CUST6X8.fon
Dimension : 6 x 8
255
Save font onto disk
06/10/98
17.43

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

Chapter 13 Creating an update disk

This chapter consists of a total of 6 pages.

VTWIN possesses a function that makes it possible to effect an update of the project loaded into the terminal without using VTWIN for the transfer.

The following can be updated and/or restored:

- Project + Firmware
- Recipes

To use this function it is essential first to create a Backup of what is to be updated (see "Chapter 6 -> Backup/Restore"**).**

We indicate below how to create an update. The parameters given there are purely for the purposes of giving an example and will need to be adapted to the requirements of any real case.

Creating the update

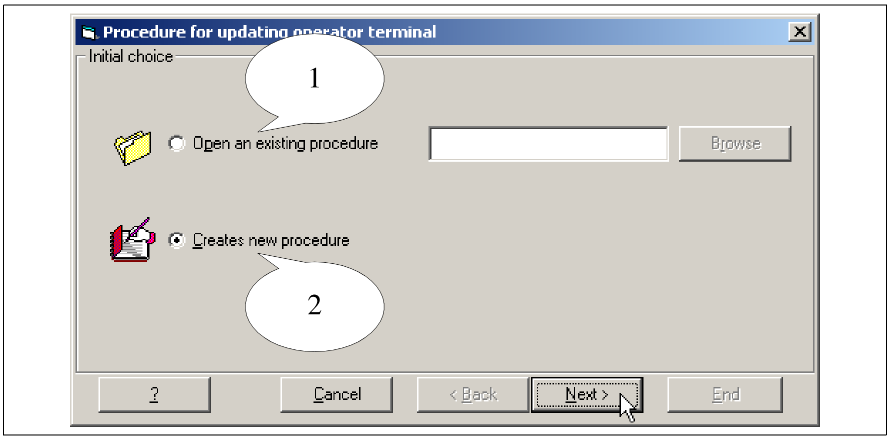
To call up the program click on ***Tools > Create update disk for operator terminal*** from the configurator menu; the principal window is now activated.

1) With this a previously saved procedure can be fetched. It can be edited to create a new one.

2) With this a new procedure can be created.

Select as in fig.

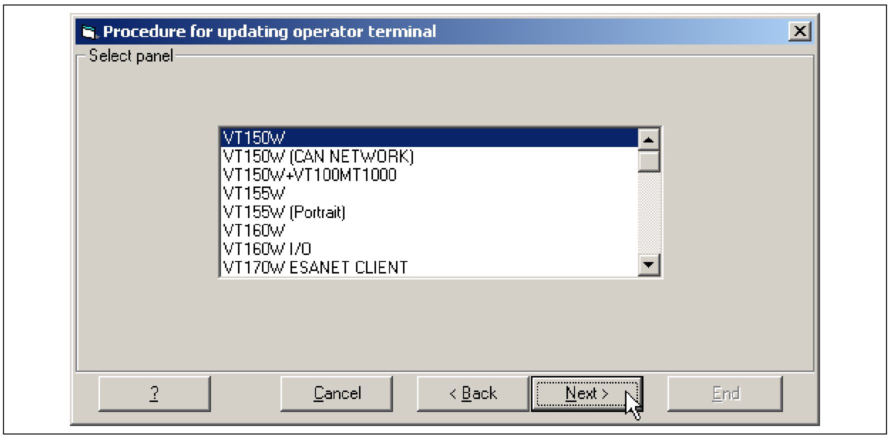
Click on Forward >


Procedure for updating operator terminal
Initial choice
1
Open an existing procedure
Browse
Creates new procedure
2
?
Cancel
< Back
Next >
End

Select the type of terminal to update.

Select as in fig.

Click on Forward >



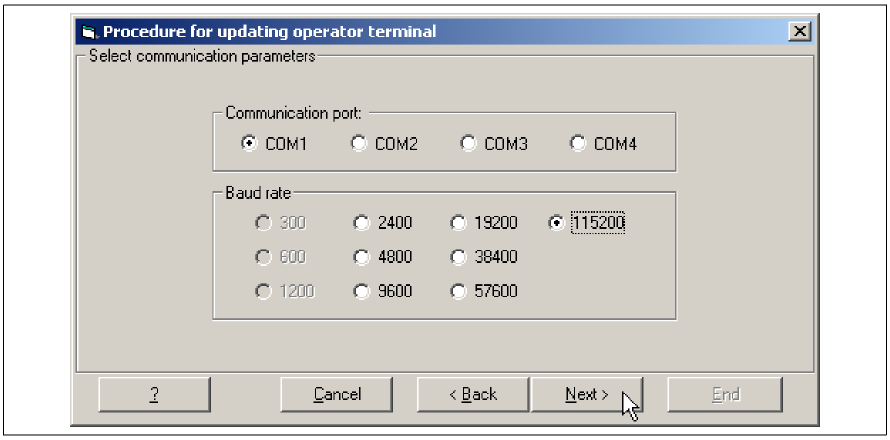
VT150W
VT150W (CAN NETWORK)
VT150W+VT100MT1000
VT155W
VT155W (Portrait)
VT160W
VT160W I/O
VT170W ESANET CLIENT

Select the serial port to be used for the transfer.

Select the speed, too.

Set as in fig.

Click on Forward >


Procedure for updating operator terminal

Select communication parameters

Communication port: [✓ COM1] COM2 COM3 COM4

Baud rate: 300 600 1200 2400 4800 9600 19200 38400 57600 [✓ 115200]

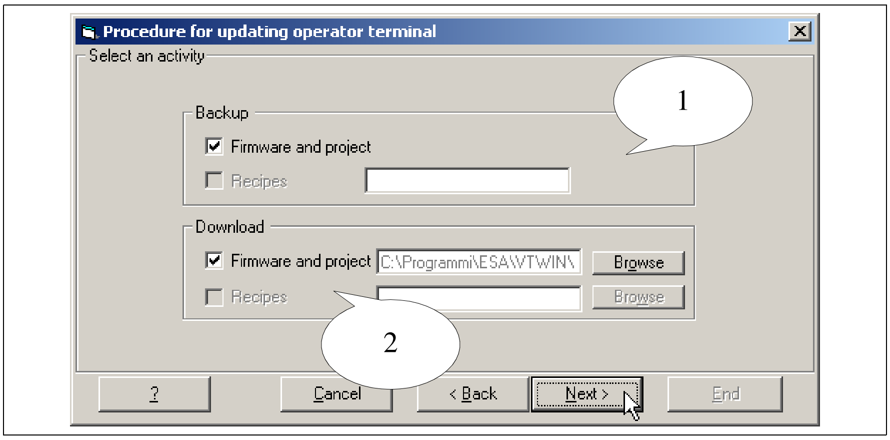
The parameters given above refer to the serial port of the PC to be used for transferring the update.

1) With this you can define whether to make a Backup (of what has been selected) before updating the terminal.

2) Use this to define the type of update and the path of the relevant file.

Select as in fig.

Click on Forward >


Procedure for updating operator terminal

Select an activity

Backup
[✓] Firmware and project
[ ] Recipes

Download
[✓] Firmware and project C:\Programmi\ESA\VTWIN\
[ ] Recipes

If certain options are not selectable, it means that they are not supported by the terminal being used (see Hardware Manual).

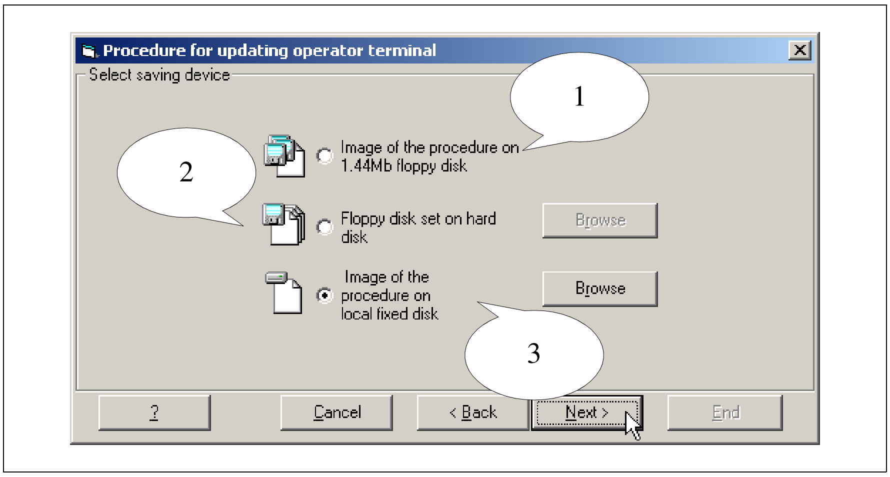
1) Creates update directly onto the floppy disk(s).

2) Creates image of the update subdivided into folders, whose contents are then copied onto floppy disk.

3) Creates update in an appropriately sized folder.

Select as in fig.

Click on Forward >


Procedure for updating operator terminal

Select saving device

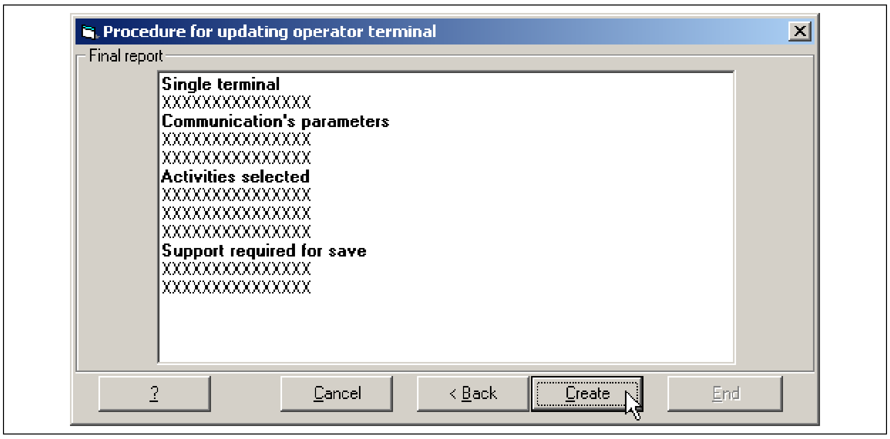
Image of the procedure on 1.44Mb floppy disk

Floppy disk set on hard disk

[✓] Image of the procedure on local fixed disk

There appears a summary of the type of procedure created.

To begin the procedure click on Create.


Procedure for updating operator terminal

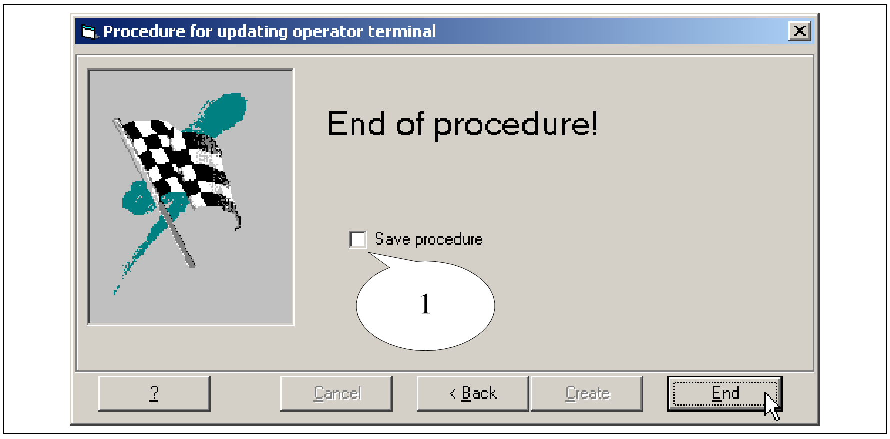
Final report

Single terminal
XXXXXXXXXXXXXXX
Communication's parameters
XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX
Activities selected
XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX
Support required for save
XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX

The procedure is created; respond to any advice that might be displayed.

1) Used to define whether to save the parametrization of the procedure created.

Click on End to finish.


Procedure for updating operator terminal

End of procedure!

☐ Save procedure

Installing the update on the PC

To copy the files needed for the update, just take the support containing the update and put it into the appropriate drive of the PC, click on ***Start > Execute…***

Digit x:\setup.exe and confirm with OK.

Replace “x” with the letter of the unit and the complete path.

Installing the files on the PC means also automatically executing the update procedure. If there is no terminal available, cancel the procedure and execute it later on (see Page 13-5 -> “Updating the terminal“).

The Command Prompt is fetched and this displays the operations that are being executed automatically:

- Copies files required
- Activation of transfer

The files are copied in the following path:

C:\ESA Elettronica\Batch Executor\<Date and time of procedure>

Respond to any questions that appear during the copying phase.

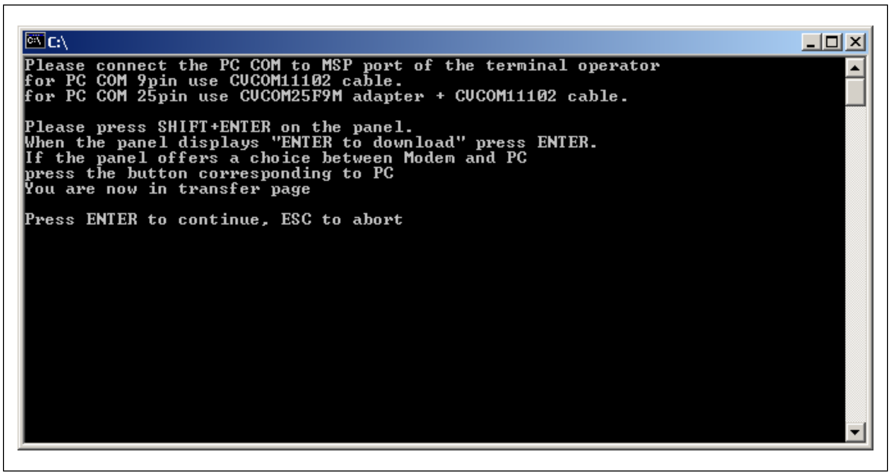
When the copying phase of the files required is over the transfer is activated (see Updating the terminal

Updating the terminal

The updating procedure is activated in automatic mode if the update is being installed onto a PC (see Page 13-4 -> “Installing the update on the PC“), otherwise it is necessary to identify the folder containing the files suitable for the update, identify the file BatchExecutor.bat and run it. For example:

C:\ESA Elettronica\Batch Executor\24-01-2003 1017\BatchExecutor.bat

The Commands Prompt is called up, displaying the operations that should be executed.



```
Please connect the PC COM to MSP port of the terminal operator
for PC COM 9pin use CVCOM11102 cable.
for PC COM 25pin use CVCOM25F9M adapter + CVCOM11102 cable.

Please press SHIFT+ENTER on the panel.
When the panel displays "ENTER to download" press ENTER.
If the panel offers a choice between Modem and PC
press the button corresponding to PC
You are now in transfer page

Press ENTER to continue, ESC to abort
```

Follow the on-screen instructions.

Chapter 14 BOOT update

This chapter consists of a total of 4 pages.

The BOOT of a terminal is the set of instructions required for the starting and management of the basic functions of the VT (e.g. management of the display, keyboard, communication, etc.).

These instructions reside in the VT in a special area of memory usually not accessible to the user.

The BOOT update function offers the user the possibility of restoring /substituting the start-up instructions of the VT in the event of abnormality or when new the terminal functions.

An update of BOOT should be carried out ONLY after first contacting ESA's Customer Care department which, if required, will supply a transfer enabling code.

Do NOT switch off current and do NOT disconnect the transfer cable during the update.

The terminal can be updated in two distinctly different ways:

- Automatic
- Aided

In both cases you need the enabling code (issued by Customer Care) and the exact model of the terminal to be updated.

Automatic

This is the simplest method, because the user only needs to connect the transfer cable to the terminal and switch on, after which the routine proceeds without further intervention on the part of the user.

The terminal is updated in two phases, the first is the transfer to the VT and the second is the self-programming.

Before starting the transfer procedure:

- Switch off the terminal.
- Connect the PC to VT using the transfer cable (see Hardware Manual).
- Switch on the terminal.
- Set the terminal up for transfer (see Hardware Manual).

Once the VT has been set up, click on ***Tools > BOOT Update > Automatic*** on the VTWIN Configurator side; and from this moment on follow the on-screen instructions carefully.

As far as VTWIN is concerned, the transfer procedure is only over after the "End" key on the last screen-page has been clicked; this is when the real update of the VT starts.

Once the transfer is over, follow the instructions set out on the display of the terminal.

Do NOT switch off the current till the display shows the words "Switch off VT and transfer firmware".

The display of the terminal might flash and/or lose contrast during the update phase.

Assisted

This method is only recommended if the Automatic method has not succeeded, because this kind of transfer means removing the back cover and acting on some jumpers, making it more inconvenient or difficult.

Before starting the transfer:

- Set up the transfer cable (see Hardware Manual).

On the VTWIN Configurator side, click on ***Tools* > *BOOT Update* > *Assisted***; and from this moment on follow the on-screen instructions and steps carefully.

The jumpers involved are identified pictorially during the update procedure.

Once the update is terminated it may be necessary to proceed to calibrate the Touch Screen (Only for terminals equipped with this function - See Hardware Manual).

Chapter 15 Multilanguage support

This chapter consists of a total of 6 pages.

Extended Font

A Font is typically composed of 256 characters; but there are special types of font that have more characters (e.g.. Unicode).

VTWIN is designed to support these fonts in Standard (256-character) format or in Extended (65536 character) format, e.g. Unicode or other formats of over 256 characters).

True Type Unicode Fonts (TTF) in Extended mode are used to support those languages that need a considerable number of characters (e.g. Chinese, Japanese, etc.) or, alternatively, to have more than one language in one single type of font.

Extended mode fonts are supported exclusively by Windows 2000 / XP / NT 4.00.

If you are using a version of VTWIN that cannot handle extended mode fonts (Rev. 4.66) or a PC with a Windows 95 / 98 operative system you will need the help of a font editor (not supplied).

Creating a Standard TTF Font

When the number of characters of the language you intend to use is greater than that supported by VTWIN or by the PC's operative system you have to preparare one or more fonts containing the characters needed limiting them to 256 per font.

The resulting fonts must then be put into the project language.

Follow these steps:

1. Use a font editor. In the example given below we have used Font Creator. (Evaluation version. The FONT CREATOR PROGRAM is copyright © 1997-2000 by High-Logic, all rights reserved. Web reference:http//www.high-logic.com/download.html.)

2. Follow the step-by-step instructions to install the program.

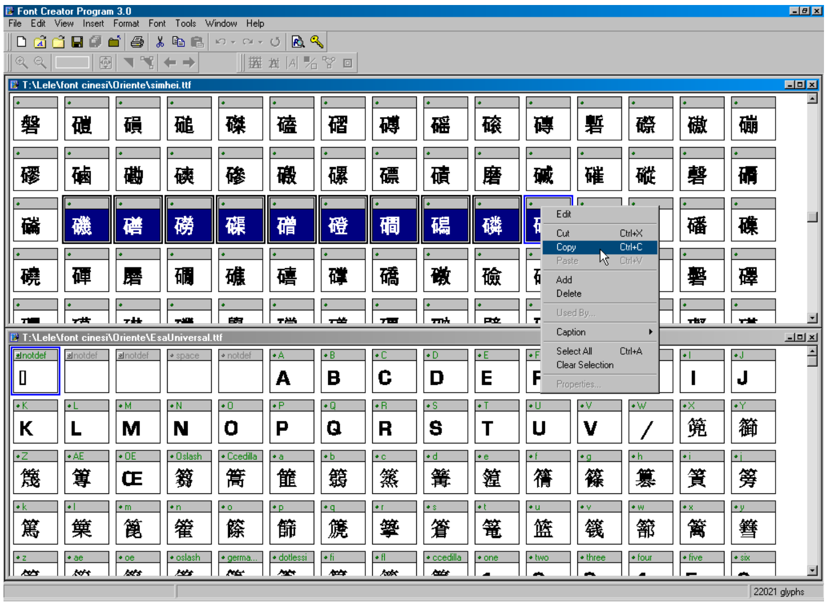
3. Run the program.

4. Click on ***File*** > ***Open*** > ***Font file*** to open the **ESA Universal** font (composed of 256 characters) and use it as a matrix (the path to select on Cd-Rom is: \Language support\Oriental\...), and, using the same procedure, open the Unicode font from which the characters are extracted (e.g. Font Simhei, whose path to select on Cd-Rom is: \Language support\Oriental\...).

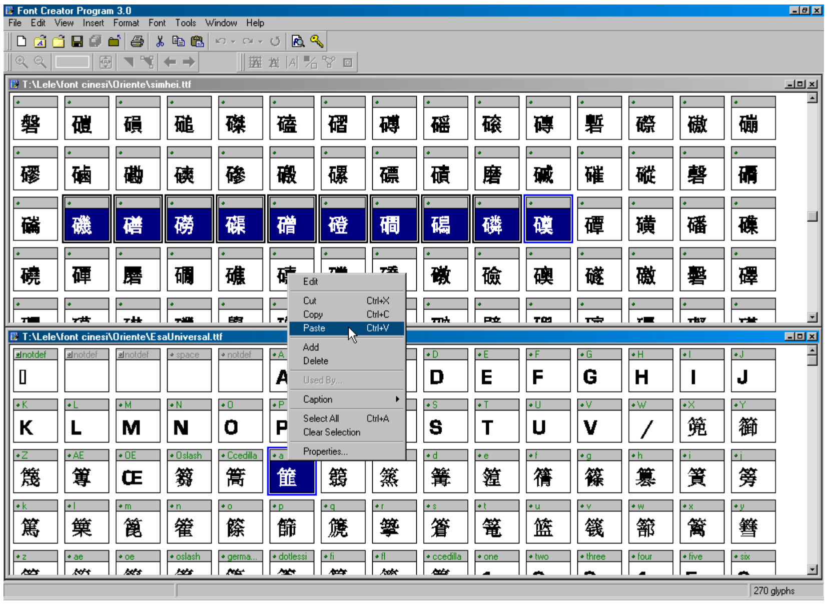
   To be able to open the Unicode fonts, you need to have installed a quantity of RAM memory equivalent to 128Mb or more on your Personal Computer.

5. Select the necessary characters and copy them into the Windows clipboard (by clicking on the right mouse key then on Copy).

The symbols slash (/), space ( ) and colon (:) must not be touched because they function as separators in the date and time. They are therefore NOT to be copied into the compartments.

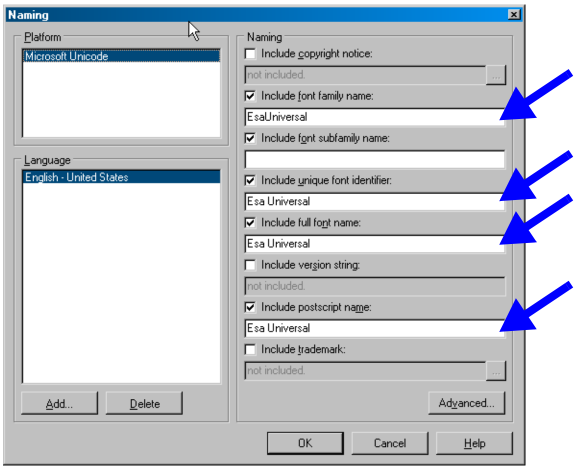


6. Paste the characters copied into the ESA Universal matrix font (by clicking on the right mouse key then on Paste).



7. Once all the necessary characters have been pasted, save the ESA Universal file with another name; to do this click on ***File > Save as...***

8. Click on ***Format > Naming*** and modify the content of the highlighted entries using the same name as assigned to the file.


Naming

Platform: Microsoft Unicode

Language: English - United States

Naming
- [ ] Include copyright notice: not included
- [✓] Include font family name: EsaUniversal
- [✓] Include font subfamily name:
- [✓] Include unique font identifier: Esa Universal
- [✓] Include full font name: Esa Universal
- [ ] Include version string: not included
- [✓] Include postscript name: Esa Universal
- [ ] Include trademark: not included

9. Save the font by clicking on ***File > Save***.

10. Install the font created in Windows. To do this use the item *Type faces* of the control panel.

11. The font is now ready to be importated into VTWIN (See Software Manual "Chapter 6 -> Project language").

Using a non-Western European Standard TTF Font

When the language to be used is not of the western European type (e.g. like Russian, Greek, etc.) the following elements must be installed on the PC:

- Fonts (of 256 characters) in the language to be used (see Readme.txt file on VTWIN CD-ROM).
- Multilanguage support of the operating system used (Windows 95, 98, Me, NT and 2000. Follow the instructions in the Windows "Online Guide" by clicking on the index and inserting as a keyword "multilanguage support").
- Keyboard of the language to be used (Follow the instructions in the Windows "Online Guide" by clicking on the index and inserting as a keyword "Layout of keyboard").

Once the above-mentioned operations have been performed there will be displayed on the tray-bar next to the clock an icon with the active reference language. If you click on this you will be able to select when you wish which keyboard and thus which language to use for editing.

Exports translation with non-Western European TTF Font

When the Import/Esport translations function is used (see “Chapter 6 -> Export translations“) with a language that is not of the western European type (e.g. Russian, Greek, etc.) the file will have to be edited using Notepad of Windows 98 or Me (Notepad versions 4.10.1998 or 4.90.0.3000) otherwise errors will occur during the phase of importing translations (see “Chapter 6 -> Export translations“).

How to proceed to change languages:

- Open Notepad ( ).
- Select ***Edit > Set font...***
- Now select from *Font* Courier New (other types may create problems of display).
- Select the type of *Script* (e.g. Greek).
- Confirm all by clicking on Ok.
- Change the layout of the keyboards (using the appropriate icon, and set it in the same way as the *Script* selected (e.g. Greek).

Now proceed with the translation.

Notepad manages files of up to 64Kbytes, so you are advised to export the texts to be translated into the various languages NOT in one single file but in separate files (e.g. English, French, Greek, Russian, etc.) or alternatively activate the function *Divide elements in various 64K files.*

Using a TTF Extended Font

To use this type of font proceed as follows:

- Install an Extended Font (e.g. Unicode).
- Using ***Opzioni Internazionali*** in the PC enable the language you intend to use (the mode in the example refers to Windows 2000 English version; if other operative systems are used, other languages and/or you need further details, consult the appropriate “On-line guide”).
    - Click on ***Start > Settings > Control Panel > Regional Options***.
    - Enter the language you intend to run (e.g. Simplified Chinese) in the *General* folder.
    - In the same folder click on the *Set default...* button and set the same language (e.g. Simplified Chinese).
    - Browse the ***Input Locales*** folder and enter the input language and the Keyboard Layout/IME.

Once the above-mentioned operations have been performed an icon with the active reference language appears on the bar beside the clock. Clicking on this allows you to select when required the keyboard and thus the language you wish to edit ing.

⚠ **The Extended Fonts are filed away in the project memory and it is not possible to know how much of this has been used until the project has been compiled.**

⚠ **For those languages that have characters that have to be represented differently depending on the preceding and/or following character, there can be no guarantee that the display is correct.**

Chapter 16 Printers directly connectable to the VT

This chapter consists of a total of 4 pages.

Requirements of the printer

The printers that can be used directly connected to the VT terminals are those thatcan be used to print in an MS-DOS environment; in other words, what you need to manage the printing is in the printer and does not need the support of the operativing system.

Identification of the printer

To identify if the type of printer is suitable for use with VTs, there needs to be a statement of its suitability for working in an MS-DOS environment. In the absence of such information the following test can be carried out (valid only for Windows 95 and 98 with parallel type printers):

1. Connect the printer to be tested to the parallel port of a PC.
2. Switch on the PC and re-start in MS-DOS mode (***Close session... -> Restart system in MS-DOS mode***).
3. Digit *Dir > LPT1* (if the PC has more than one parallel port, check where the printer is connected) and press Return.
4. If there is a print-out, or if the "Ready" LED comes on, or if the "Form feed" LED stays on it means that the operation was successful and that the printer is suitable for direct connection to the VT.
5. If the message "Write error in peripheral LPT1. Cancel, Try again, Ignore, Omit?" it means:
   - the cable is not connected correctly (Check and try again)
   - the printer is not on (Check and try again)
6. If the system is blocked it means that an LPT number has been digited that is not available on the PC (Check and try again)
7. If none of the things indicated in point 4 happens, and there are no error messages, it means that the printer is **not suitable to be** connected directly to the VT.

Printers tested Below is a list of printers tested by ESA elettronica S.p.A.

Printer model	To use with...							
Epson LX-1050Plus	☺							
Epson LX-400	☺							
Epson Stylus Pro XL-Plus	☺							
Fusjitsu DX2250					☺			
HP Deskjet 1120C Professional series							☺	
HP Deskjet 1125C Professional series							☺	
HP Deskjet 690C							☺	
HP Deskjet 840C							☺	
HP Laserjet 2100							☺	
HP Laserjet 4 Plus							☺	
HP Laserjet 4P							☺	
HP Laserjet 4V							☺	
IBM 2381								☺
Lexmark 2381 Plus Forms Printer			☺					
Lexmark 238/239								☺
Lexmark 3200	☹	☹	☹	☹	☹	☹	☹	☹
Olivetti JP350ws					☺			
Siemens PT88 parallel			☺					
Siemens PT88 serial				☺				
VTWIN device	▼	▼	▼	▼	▼	▼	▼	▼
Epson Parallel	┘							
Epson Serial		┘						
ESA elettronica Parallel User-Defined			┘					
ESA elettronica Serial User-Defined				┘				
Fujitsu/Olivetti Deskjet Parallel					┘			
Fujitsu/Olivetti Deskjet Serial						┘		
HP Laserjet							┘	
IBM Proprinter								┘

Key:
☺ - Prints with driver indicated
☹ - Does not print, cannot be used with VT terminals.

Chapter 17 Free terminal

This chapter consists of a total of 2 pages.

This allows the user to activate the transfer mode of the Free terminal communication protocol. (See Hardware Manual.)

During transfer it is possible to choose which version of free terminal to transfer, Standard or Ver. 3.0 (for terminals which support it). The only differences are the number of ▢ to press in order to access the parameter change page: the Standard version needs ▢ while Ver. 3.0 needs ▢.

Ver. 3.0 offers extra safety in case of accidental access to the parameter change page (see Hardware Manual).

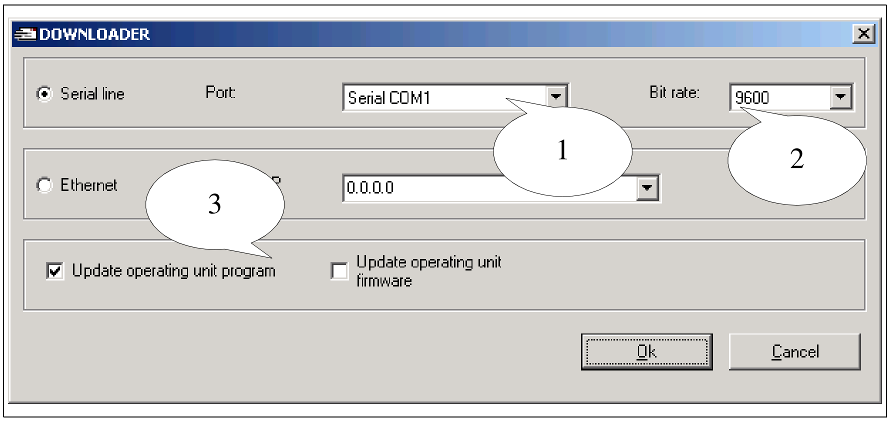
Transfer of free terminal

Click on ***Start*** > ***Programs*** > ***VTWIN*** > ***Free terminal;*** a list of the terminals supporting it appears. When one of these is selected and the user clicks on ▢ Transfer, the following mask appears:

1) Makes it possible to choose the communication port used by the PC.

2) Allows the user to select the data transfer speed.

3) Allows the user to define whether the VT Firmware is also to be sent during the transfer.



DOWNLOADER

(● Serial line) Port: Serial COM1 Bit rate: 9600

(Ethernet) 0.0.0.0

[✓] Update operating unit program [ ] Update operating unit firmware

The mask shows all the parameters needed for establishing the connection between the VT terminal and the PC used for the programming.

Chapter 18 ActiveX

This chapter consists of a total of 2 pages.

When it becomes necessary to connect a device directly to a personal computer or check data of many VTs connected to each other in a network, this can be done by using an adapter (see Hardware Manual) with the related project (see "Introduction -> What is a project?" and/or "Chapter 5 -> Project for Single VT:") and by utilizing the capabilities of ActiveX (see HoL for further details).

User application package

The User application package to be employed on a personal computer must be created in a development environment (outside VTWIN) that supports the OLE Automation technology. The User application package (e.g. Supervisor) is interfaced with the VTWIN project using the capabilities offered by ActiveX components.

⚠ **The user application package takes care of transferring the ADT project (file .OBJ and .FW, see "Introduction -> Files generated by an ADT project") to the adapter.**

For all necessary details and information relating to the capabilities of ActiveX see HoL.

To make this easier to understand, you will find some application packages included for illustration purposes in the ActiveX Samples folder.

Chapter 19 Technical support

This chapter consists of a total of 4 pages.

Problems related to the use of ESA VT terminals should be referred to our Customer Care service. The department is contactable on week days in office hours.

International Customer Care

The International Customer Care service can be contacted by:

Telephone:	++39-031757400
Fax:	++39-031751777
E-Mail:	customer.care@esahmi.com
Web site:	http://www.esahmi.com

International Product Returns

Should it be necessary to return the VT terminal for repair:

- Contact our International Customer Care service to authorize the return.
- Fill in all parts of the form to accompany the product.

Our International Customer Care service will supply all the necessary information for returning a an item.

!!! IMPORTANT NOTE !!!

ESA elettronica S.p.A. will accept:

- goods carriage free / freight prepaid (transport at customer’s cost).
- goods carriage forward / freight collect (transport paid by ESA) **only with the prior authorization of the company**.

ESA elettronica S.p.A. will reject:

- any returned goods carriage forward where there has been no prior authorization.

It is not necessary to send connectors, cables and accessories (unless they are thought to be linked to the problem indicated).

Thank you for your kind co-operation.

REPAIRS RETURN FORM

NOTE: Please complete form in BLOCK CAPITALS.

Product: .... **S/N:** ☐☐-☐☐-☐☐☐☐☐

Customer details (must be filled in)

Compiled by: ....

Company : ....

Full address : .... Post Code: ....

Town : .... County: ....

Tel. no. : .... Fax: ....

Contact person (where different from above)

Name : ....

Tel. no. : .... Fax: ....

Information regarding problem (must be filled in)

Device connected: ....

Detailed description of the problem and the circumstances under which it occurs:

....

....

....

....

....

....

....

Notes: ....

....

....

....

....

Customer Care worker contacted: ....

Date of compilation: ...../...../........ Signature: ....

Appendix A

This chapter consists of a total of 4 pages.

The tables below show the contents of the DEMO project.

Table A.1: Project images - Graphic (Part 1 of 3)

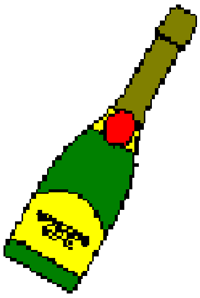
		▶	◀
Bottle	Esa	Arrow right	Arrow left
▶	◀	◉	⊙
Arrow right short	Arrow left short	Previous menu 1	Previous menu 2
⊙			
Previous menu 3	Mixer 1	Mixer 2	Mixer 3
Mixer 4	Mixer 5	Mixer 6	Mixer 7

Table A.1: Project images - Graphic (Part 2 of 3)

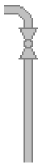
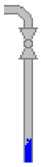
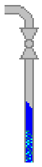
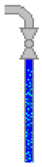
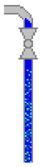
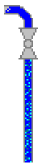
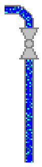
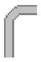
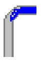
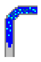
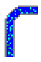
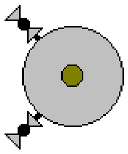
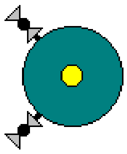
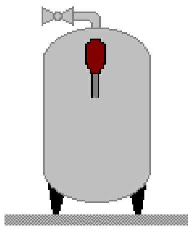
Mixer 9	Mixer 10	Mixer 11	Mixer 12
Mixer 13	Mixer 14	Mixer 15	Mixer 16
			
Cool right 1	Cool right 2	Cool right 3	Cool right 4
			
Cool right 5	Cool right 6	Cool right 7	Cool left 1

Table A.1: Project images - Graphic (Part 3 of 3)

			
Cool left 2	Cool left 3	Cool left 4	Heat 1
Heat 2	Heat 3	Heat 1	Heat 2
			
Heat 3	Heat 4	Autoclave empty	Autoclave full
			
Autoclave background			

Appendix B

This chapter consists of a total of 4 pages.

Structure of AWL and ASC files

This types of files are generated by the development software of the program for the device, further information must therefore be sought in the maker's manual.

A list of the devices which support the importing of variables using AWL and ASC files is given below.

Table B.1: Devices supporting the importation of variables

Make	Description	Notes
SAIA	PCD 1/2 xx7	Step 7 Rev. ≥ 5.00
SIEMENS	S7 300/400	Step 7 Rev. ≥ 5.00

Structure of CSV and TXT files

These files both use a tabular structure of rows and columns to identify all the elements contained in them.

Table B.2: Control characters

Part of the file	CSV	TXT
Column separator (one at a time)	Semi-colon (;)	Tab
		Space
		Comma (,)
		Semi-colon (;)
		Defined by user
Field	May be enclosed by "". If the value includes the character " this is duplicated in "". E.g. "cam""po" -> cam"po	May be enclosed by "". If the value includes the character " this is duplicated in "". E.g. "cam""po" -> cam"po
Line end	<CR><LF>	<CR><LF>

The positions of the individual elements in the files relating to the variables. Some columns are always present while others are optional.

Table B.3: Meaning of the fixed columns (Part 1 of 2)

Column	Field	Description
1	Name of variable	Name of the variable
2	Name of device	Name of the device to which the variable is connected
3	Address	Text representation of the address obstained by joining up the labels represented in VTWIN by usingthe character comma (,). Example: Given an address with label DB and DBW and respective values 10 and 11, the result is DB10,DBW11

Table B.3: Meaning of the fixed columns (Part 2 of 2)

Column	Field	Description
4	Type	STRING(<#characters>: String length(<#characters>) BIT: 1 bit W: Integer without sign (16 bits) DW: Integer without sign (32 bits) I16: Integer with sign (16 bits) I32: Integer with sign (32 bits) BCD: Integer BCD (16 bits) BCD8: Integer BCD (32 bits) B: Byte B8: Byte with sign FLOAT: floating point A field that is empty when imported is replaced with one pre-set for that data area.
5	Description of type	Description in VTWIN of format
6	Name of data area	Name of data area of the device connected

The format of the optional columns is made up of:
<Name of field>=<Value of field >

Table B.4: Meaning of optional columns

Column	Field	Description
Any (after number 6)	MIN	Limit of lowest value setting. May be a constant or a variable. The name of the variable cannot begin with a number.
Any (after number 6)	MAX	Limit of highest value setting. May be a constant or a variable. The name of the variable cannot begin with a number.
Any (after number 6)	VTVAL1	Minimum linear scale value of the terminal. May be a constant or a variable. The name of the variable cannot begin with a number.
Any (after number 6)	PLCVAL1	Minimum linear scale value of the device. May be a constant or a variable. The name of the variable cannot begin with a number.
Any (after number 6)	VTVAL2	Maximum linear scale value of the terminal. May be a constant or a variable. The name of the variable cannot begin with a number.
Any (after number 6)	PLCVAL2	Maximum linear scale value of the device. May be a constant or a variable. The name of the variable cannot begin with a number.

An example of a CSV file follows.

```
VAR001;ABB:Modbus_b;MW0,2;DW
VAR002;ABB:Modbus_b;MW1,15;STRING(16)
ASS_ALL;”S7 300 Assembler”;DB100 DBW12;W;MIN=0;MAX=300
```

Appendix C

This chapter consists of a total of 30 pages.

Using VTWIN you can change the type of device connected to the VT even after the project has been started or finished. In some cases this operation causes a partial loss of the variables, in others the loss is total. Below we list the devices that support such conversion and with which the loss of the variables is therefore minimized or, in some cases, avoided.

⚠ **Conversion when using a device not listed in Table C.1 will cause all the variables to be lost.**

Meaning of the tables

In order to understand which type of variable is maintained after the conversion and which lost, the following points need to be clarified:

- "Table C.1 on Page AC-4" lists the devices supporting conversion; it is made up of two columns: Group and Device. By "Group" we mean a collection having common features; the term "Device" is used to refer to the driver used in VTWIN for making the connection.

  Example.

  Group A consists of a series of devices produced by Allen & Bradley.

- From "Table C.1 on Page AC-5" onwards we give a detailed list of the areas that are converted, relating these to the Group the Device belongs to. The number after the hyphen in the Type column (E.g. Dword - 4) indicates the length of the area in Bytes (4 Bytes).

Conversion mechanism

To ascertain how the conversion will turn out, the tables relating to the device to be converted must be chosen and its correspondence checked using the number of the line.

Example.

Assume we want to convert from SIEMENS S7/300 to SAIA PCD2.

- Identify the tables relating to these devices
- Check the first entry for the device to be converted (Siemens) for line number (8), kind of Data Area (Counter) and Position (1).
- Check if line No.8 of the table of the device you want to convert to corresponds in terms of the kind of Data Area; if it does procede to check if the position corresponds (for Saia the position is 5, not 1); if it corresponds conversion is possible, otherwise the variable related to that item of data will be lost.

In the example the tables are limited to 12 lines for simplicity.

Group M			
SIEMENS S7/300-400			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	**Counter**	**1**	**Counter - 2**
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	**DB/DBW**	1	Byte - 1
		2	Word - 2
		3	**Dword - 4**
		4	**String - 0**
		5	**Floating Point - 4**
		6	Timer 1/100 sec - 2
		7	Timer 1/10 sec - 2
		8	Timer 1 sec - 2
		9	Timer 10 sec - 2
11	Merker	1	--
		2	Byte - 1
		3	Word - 2
		4	Dword - 4
12	--	1	--
		2	--
		3	--
		4	--

Group F			
SAIA PCD / S-BUS			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	**Counter**	1	--
		2	--
		3	--
		4	--
		5	**Counter - 4**
9	--	--	--
10	**Register**	1	--
		2	--
		3	**Dword - 4**
		4	**String - 0**
		5	**FloatingPoint - 4**
		6	--
		7	--
		8	--
		9	--
11	Flag	1	Flag - 0
		2	--
		3	--
		4	--
12	--	1	--
		2	--
		3	--
		4	--

As can be seen, the area of line No.8 cannot be converted, while on line No.10 it is possible to convert positions 3-4-5.

Devices supported in conversion

Below is a list giving the devices supporting the conversion of variables.

Table C.1: List of devices supporting conversion

Group	Device
A	ALLEN BRADLEY DH485 ALLEN BRADLEY Micrologix 1500 ALLEN BRADLEY PLC5 ALLEN BRADLEY SLC500 5/03 - 5/04 DF1
B	ALLEN BRADLEY Micrologix 1000
C	GE-FANUC Series 90-30
D	OMRON H / Host Link
E	OMRON CS1
F	SAIA PCD SAIA S-BUS
G	SAIA PCD1/2 xx7
H	SAIA Profibus
I	SIEMENS 115-CPU 945 SIEMENS 135/SLOT PLC SIEMENS 90,95,100,115 SIEMENS S5 Interbus SIEMENS S5 Profibus
L	SIEMENS S7 200 SIEMENS S7 200 PPI Network SIEMENS S7-200 PPI Network 187500
M	SIEMENS S7 300/400 SIEMENS S7 Interbus SIEMENS S7 Profibus
N	TELEMECANIQUE ALTIVAR-MODBUS TELEMECANIQUE UNITELWAY TSX 07/37/47 PREMIUM TELEMECANIQUE UNITELWAY TSX 17
O	TELEMECANIQUE Reglage TSX17/20 TELEMECANIQUE Reglage TSX47

Table C.1: Type of conversion ordered by Group (Part 1 of 2)

Group A			
ALLEN BRADLEY DH485, Micrologix 1500, PLC5, SLC500 DF1			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter Acc (file, El)	1	Counter Acc - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	Integer:File/Element	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	--
		6	--
		7	--
		8	--
		9	--
11	Bit (File, Element)	1	--
		2	--
		3	Bit -2
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

* not Micrologix 1500 and PLC5, -- not subject to conversion

Table C.1: Type of conversion ordered by Group (Part 2 of 2)

Group A			
ALLEN BRADLEY DH485, Micrologix 1500, PLC5, SLC500 DF1			
No.	**Data Area**	**Pos.**	**Type**
16	Timer Acc (File, Element)	1	Timer Acc - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input (File, Element)	1	--
		2	--
		3	Input - 2
		4	--
20	Output (File, Element)	1	--
		2	--
		3	Output - 2
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	Counter Pre (File, Elem)	--	Counter Pre - 2
25	Timer Pre (File, Element)	--	Timer Pre - 2
26	Floating *	1	Dword - 4
		2	FloatingPoint - 4
27	Ascii *	--	String - 0
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

* not Micrologix 1500 and PLC5, -- not subject to conversion

Table C.2: Type of conversion ordered by group (Part 1 of 2)

Group B			
ALLEN BRADLEY Micrologix 1000			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter Acc (elem)	1	Counter Acc - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	Integer (Element)	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	--
		6	--
		7	--
		8	--
		9	--
11	Bit (Element)	1	--
		2	--
		3	Bit -2
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.2: Type of conversion ordered by group (Part 2 of 2)

Group B			
ALLEN BRADLEY Micrologix 1000			
No.	**Data Area**	**Pos.**	**Type**
16	Timer Acc (Elem)	1	Timer Acc - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input (element)	1	--
		2	--
		3	Input - 2
		4	--
20	Output (element)	1	--
		2	--
		3	Output - 2
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	Counter Pre (elem)	--	Counter Pre - 2
25	Timer Pre (elem)	--	Timer Pre - 2
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.3: Type of conversion ordered by Group (Part 1 of 2)

Group C			
GE-FANUC Series 90-30			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	--	1	--
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	Register	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	--
		6	--
		7	--
		8	--
		9	--
11	Discrete Internal	1	--
		2	--
		3	Word - 2
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.3: Type of conversion ordered by Group (Part 2 of 2)

Group C			
GE-FANUC Series 90-30			
No.	**Data Area**	**Pos.**	**Type**
16	--	1	--
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Discrete Input	1	--
		2	--
		3	Word - 2
		4	--
20	Discrete Output	1	--
		2	--
		3	Word - 2
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.4: Type of conversion ordered by Group (Part 1 of 2)

Group D			
OMRON H / Host Link			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	Auxiliary Relay	1	Auxiliary relay - 2
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	Counter - 2
		2	Counter preset - 2
		3	Rev Counter preset - 2
		4	Counter CNTW preset - 2
		5	--
9	--	--	--
10	DM	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	--
		6	--
		7	--
		8	--
		9	--
11	Relay	1	--
		2	--
		3	Relay - 2
		4	--
12	Holding Relay	1	Holding Relay - 2
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.4: Type of conversion ordered by Group (Part 2 of 2)

Group D			
OMRON H / Host Link			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	Timer - 2
		2	Timer preset - 2
		3	Timer TMS preset - 2
		4	Timer TIMW preset - 2
		5	Timer TMHW preset - 2
		6	Speed timer preset - 2
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	--	1	--
		2	--
		3	--
		4	--
20	--	1	--
		2	--
		3	--
		4	--
21	--	--	--
22	--	--	--
23	Link Relay	--	Link Relay - 2
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.5: Type of conversion ordered by Group (Part 1 of 2)

Group E			
OMRON CS1			
No.	**Data Area**	**Pos.**	**Type**
1	Auxiliary area bit	--	Auxiliary area bit (A) - 0
2	Auxiliary area bit RO	--	Auxiliary area bit RO - 0
3	Auxiliary area Word	1	Word - 2
		2	Dword - 4
		3	String - 0
		4	FloatingPoint - 4
4	Auxiliary area Word RO	1	Word - 2
		2	Dword - 4
		3	String - 0
		4	FloatingPoint - 4
5	Core Input/Output bit	--	Core Input/Output bit (CIO) - 0
6	Core Input/Output word	1	Word - 2
		2	Dword - 4
		3	String - 0
7	Core Completion flag	--	Counter completion flag (C) - 0
8	Counter current value	1	Word - 2
		2	--
		3	--
		4	--
		5	--
9	Data Register	--	Word - 2
10	Data Memory Area	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	FloatingPoint - 4
		6	--
		7	--
		8	--
		9	--
11	Holding Area bit	1	Holding area bit (H) - 0
		2	--
		3	--
		4	--
12	Holding Area Word	1	Word - 2
		2	Dword - 4
		3	String - 0
		4	FloatingPoint - 4
13	Index Register	--	Dword - 4
14	Task Flag Area	--	Task flag area (TK) - 0
15	Timer Completion Flag	--	Timer Completion flags - 0

-- not subject to conversion

Table C.5: Type of conversion ordered by Group (Part 2 of 2)

Group E			
OMRON CS1			
No.	**Data Area**	**Pos.**	**Type**
16	Timer Current Value	1	Word - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	Work Area Bit	--	Work Area Bit (W) - 0
18	Work Area Word	1	Word - 2
		2	Dword - 4
		3	String - 0
		4	FloatingPoint - 4
19	--	1	--
		2	--
		3	--
		4	--
20	--	1	--
		2	--
		3	--
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.6: Type of conversion ordered by Group (Part 1 of 2)

Group F			
SAIA PCD / S-BUS			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	--
		2	--
		3	--
		4	--
		5	Counter - 4
9	--	--	--
10	Register	1	--
		2	--
		3	Dword - 4
		4	String - 0
		5	FloatingPoint - 4
		6	--
		7	--
		8	--
		9	--
11	Flag	1	Flag - 0
		2	--
		3	--
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.6: Type of conversion ordered by Group (Part 2 of 2)

Group F			
SAIA PCD / S-BUS			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	--
		2	--
		3	--
		4	--
		5	--
		6	--
		7	Timer - 4
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input	1	Input - 0
		2	--
		3	--
		4	--
20	Output	1	Output - 0
		2	--
		3	--
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	-	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	Data block	1	Dword - 4
		2	String - 0
		3	FloatingPoint - 4

-- not subject to conversion

Table C.7: Type of conversion ordered by Group (Part 1 of 2)

Group G			
SAIA PCD 1/2 xx7			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	Counter - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	DBW	1	Byte - 1
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	FloatingPoint - 4
		6	Timer 1/100 sec - 2
		7	Timer 1/10 sec - 2
		8	Timer 1 sec - 2
		9	Timer 10 sec - 2
11	Merker	1	--
		2	Byte - 1
		3	Word - 2
		4	Dword - 4
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.7: Type of conversion ordered by Group (Part 2 of 2)

Group G			
SAIA PCD 1/2 xx7			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	Timer - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input	1	--
		2	Byte - 1
		3	Word - 2
		4	Dword - 4
20	Output	1	--
		2	Byte - 1
		3	Word - 2
		4	Dword - 4
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	-	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.8: Type of conversion ordered by Group (Part 1 of 2)

Group H			
SAIA Profibus			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	--
		2	--
		3	--
		4	--
		5	Dword - 4
9	--	--	--
10	Register	1	--
		2	--
		3	Dword - 4
		4	String - 0
		5	FloatingPoint - 4
		6	--
		7	--
		8	--
		9	--
11	Bit Flag	1	Bit - 0
		2	--
		3	--
		4	--
12	Word Flag	1	Word - 2
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.8: Type of conversion ordered by Group (Part 2 of 2)

Group H			
SAIA Profibus			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	--
		2	--
		3	--
		4	--
		5	--
		6	--
		7	Dword - 4
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Bit Input	1	Bit - 0
		2	--
		3	--
		4	--
20	Bit Output	1	Bit - 0
		2	--
		3	--
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	Word Input	--	Word - 2
29	Word Output	--	Word - 2
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.9: Type of conversion ordered by Group (Part 1 of 2)

Group I			
SIEMENS S5			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	Counter - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	DB/DBW	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	--
		6	Timer 1/100 sec - 2 *
		7	Timer 1/10 sec - 2 *
		8	Timer 1 sec - 2 *
		9	Timer 10 sec - 2 *
11	Merker	1	--
		2	Byte - 1**
		3	Word - 2
		4	Dword - 4
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

* no S5 Interbus, ** no S5 Interbus and Profibus, -- not subject to conversion

Table C.9: Type of conversion ordered by Group (Part 2 of 2)

Group I			
SIEMENS S5			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	Timer - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input	1	--
		2	Byte - 1 **
		3	Word - 2
		4	Dword - 4
20	Output	1	--
		2	Byte - 1 **
		3	Word - 2
		4	Dword - 4
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

* no S5 Interbus, ** no S5 Interbus and Profibus, -- not subject to conversion

Table C.10: Type of conversion ordered by Group (Part 1 of 2)

Group L			
SIEMENS S7/200			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	Word - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	Register	1	Byte (VB) - 1
		2	Word (VW) - 2
		3	Dword (VD) - 4
		4	String (VB) - 0
		5	FloatingPoint (VD) - 4
		6	--
		7	--
		8	--
		9	--
11	Merker	1	Bit - 0
		2	--
		3	--
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.10: Type of conversion ordered by Group (Part 2 of 2)

Group L			
SIEMENS S7/200			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	Word - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input	1	Bit - 0
		2	--
		3	--
		4	--
20	Output	1	Bit - 0
		2	--
		3	--
		4	--
21	High Speed Counter	--	Word - 2
22	Special Marker	--	Bit - 0
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Table C.11: Type of conversion ordered by Group (Part 1 of 2)

Group M			
SIEMENS S7/300-400			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter	1	Counter - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	DB/DBW	1	Byte - 1 **
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	Floating Point - 4
		6	Timer 1/100 sec - 2 *
		7	Timer 1/10 sec - 2 *
		8	Timer 1 sec - 2 *
		9	Timer 10 sec - 2 *
11	Merker	1	--
		2	Byte - 1**
		3	Word - 2
		4	Dword - 4
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

* no S5 Interbus, ** no S5 Interbus and Profibus, -- not subject to conversion

Table C.11: Type of conversion ordered by Group (Part 2 of 2)

Group M			
SIEMENS S7/300-400			
No.	**Data Area**	**Pos.**	**Type**
16	Timer	1	Timer - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Input	1	--
		2	Byte - 1 **
		3	Word - 2
		4	Dword - 4
20	Output	1	--
		2	Byte - 1 **
		3	Word - 2
		4	Dword - 4
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

* no S5 Interbus, ** no S5 Interbus and Profibus, -- not subject to conversion

Table C.12: Type of conversion organized by Group (Part 1 of 2)

Group N			
TELEMECANIQUE Unitelway			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	--	1	--
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	Word	1	--
		2	Word - 2
		3	Dword - 4 *
		4	String - 0 *
		5	Floating Point - 4 **
		6	--
		7	--
		8	--
		9	--
11	--	1	--
		2	--
		3	--
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

* no T.U. Altivar-Modbus, ** no T.U. Altivar-Modbus and TSX17, -- not subject to conversion

Table C.12: Type of conversion organized by Group (Part 2 of 2)

Group N			
TELEMECANIQUE Unitelway			
No.	**Data Area**	**Pos.**	**Type**
16	--	1	--
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	--	1	--
		2	--
		3	--
		4	--
20	--	1	--
		2	--
		3	--
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	--	--	--
25	--	--	--
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

* no T.U. Altivar-Modbus, ** no T.U. Altivar-Modbus and TSX17, -- not subject to conversion

Table C.13: Type of conversion organized by Group (Part 1 of 2)

Group O			
TELEMECANIQUE Reglage			
No.	**Data Area**	**Pos.**	**Type**
1	--	--	--
2	--	--	--
3	--	1	--
		2	--
		3	--
		4	--
4	--	1	--
		2	--
		3	--
		4	--
5	--	--	--
6	--	1	--
		2	--
		3	--
7	--	--	--
8	Counter Value	1	Counter Value - 2
		2	--
		3	--
		4	--
		5	--
9	--	--	--
10	Register	1	--
		2	Word - 2
		3	Dword - 4
		4	String - 0
		5	--
		6	--
		7	--
		8	--
		9	--
11	Bit	1	Bit - 0
		2	--
		3	--
		4	--
12	--	1	--
		2	--
		3	--
		4	--
13	--	--	--
14	--	--	--
15	--	--	--

-- not subject to conversion

Table C.13: Type of conversion organized by Group (Part 2 of 2)

Group O			
TELEMECANIQUE Reglage			
No.	**Data Area**	**Pos.**	**Type**
16	Timer Value	1	Timer Value - 2
		2	--
		3	--
		4	--
		5	--
		6	--
		7	--
17	--	--	--
18	--	1	--
		2	--
		3	--
		4	--
19	Bit Input	1	Bit Input - 0
		2	--
		3	--
		4	--
20	Bit Output	1	Bit Output - 0
		2	--
		3	--
		4	--
21	--	--	--
22	--	--	--
23	--	--	--
24	Counter Preset	--	Counter Preset - 2
25	Timer Preset	--	Timer Preset - 2
26	--	1	--
		2	--
27	--	--	--
28	--	--	--
29	--	--	--
30	--	1	--
		2	--
		3	--

-- not subject to conversion

Index