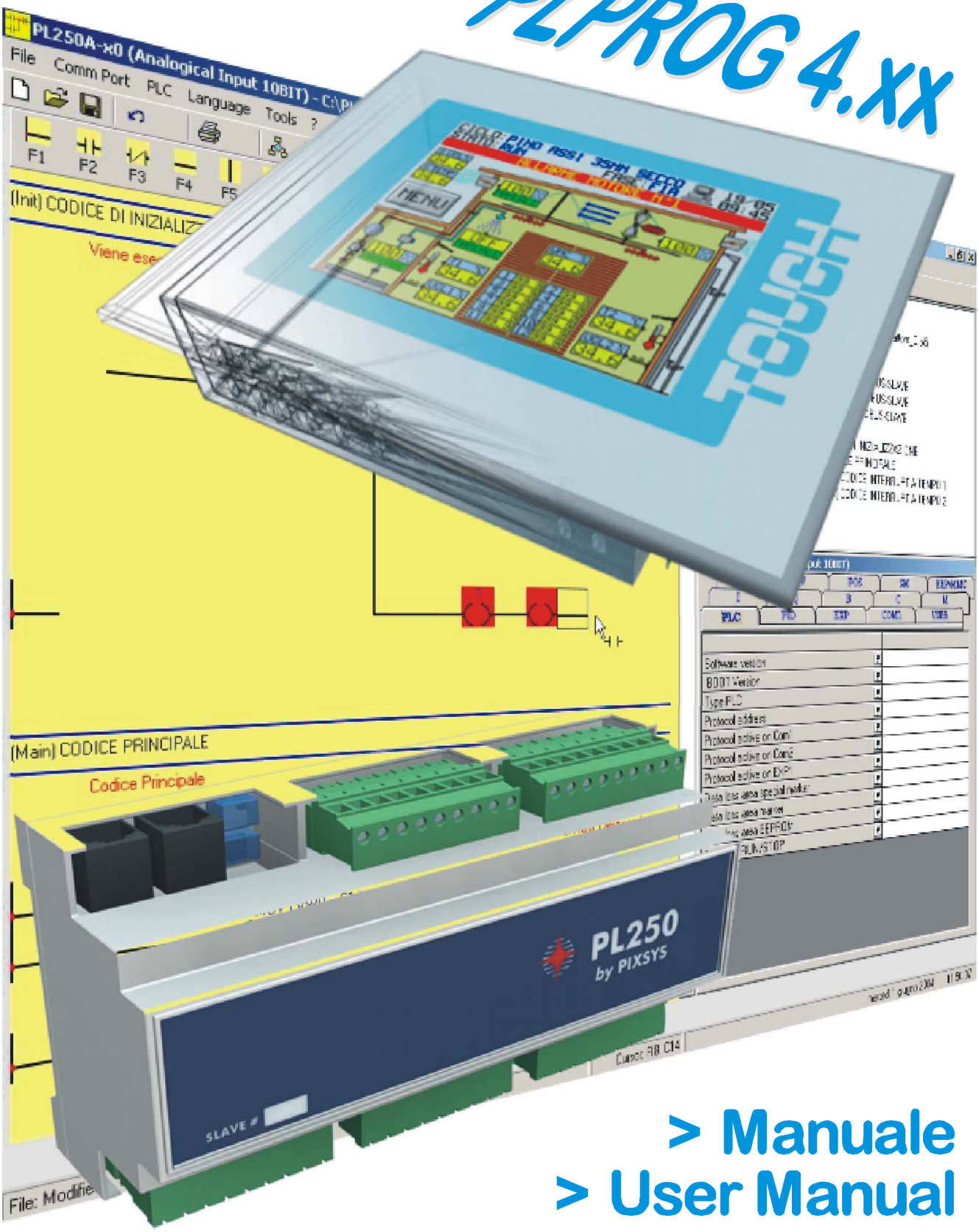
PLPROG 4.XX



> Manuale
> User Manual

Contents

1 ENGLISH

2 Overview

In many cases, technological development over the last few years has made PLC more of a necessity rather than a choice. The complexity of applications required by the automation market has made these devices increasingly more important.
Pixsys has developed a wide range of products that are easy to instal and can be configured from your PC. The development environment provides all the programming and library tools needed to quickly build common industrial automation applications.
The purpose of this manual is to outline the main concepts of PLProg software without going into details on the product or on Ladder programming, both amply covered in other documentation. Commercial details are also dealt with elsewhere.

3 Kit components

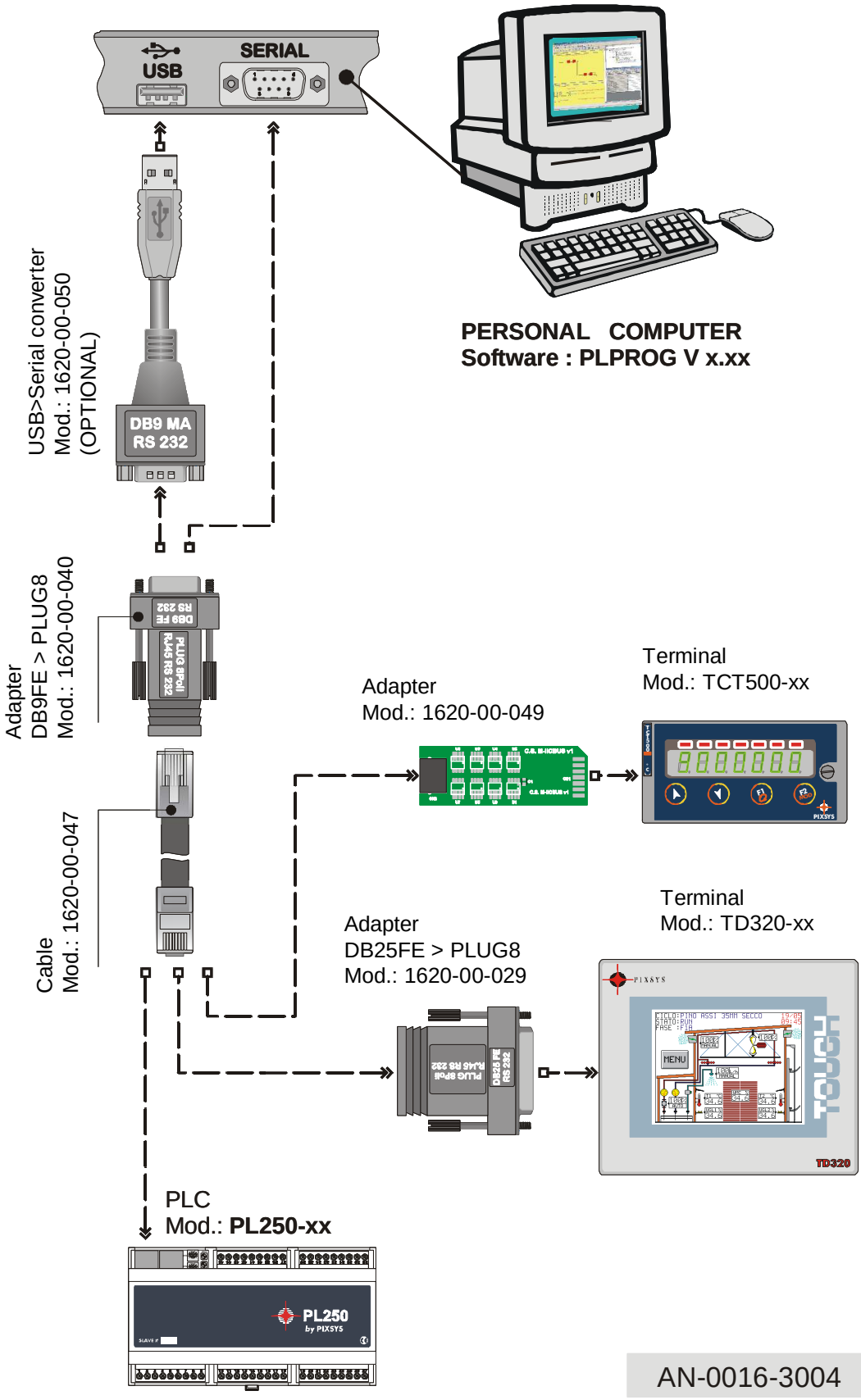


Figure 3.1: Connections between Pixsys devices and PC using cabling provided in the kit.

4 PlProg installation

After inserting the CD, the installation wizard will launch automatically and guide the user through the installation process with a series of dialog windows. A few examples are found below to show the most important steps.

Figure 4.1: The first window asks you to select language during installation.

Figure 4.2: Select installation folder.

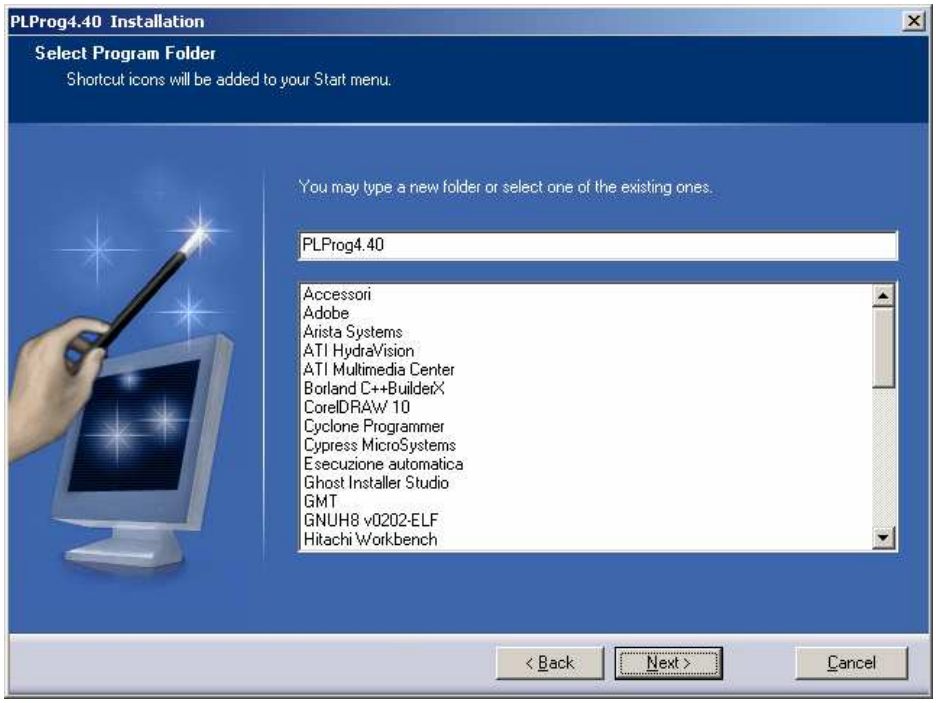


Figure 4.3: Item on start menu.

PLProg is now installed and ready to use and can be launched from the Start/Program menu or from the desktop shortcut.

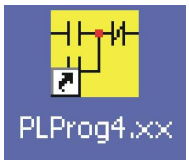


Figure 4.4: The installation wizard puts a shortcut icon on your desktop.

The user can choose between Italian and English from the main menu. For further information see par. 6.1.4.

4.1 Hardware test

With just a few steps you can immediately check that the material works correctly.
First of all, it is necessary to connect the PC and device as illustrated in Figure 3.1. Once connected, launch PLProg and set the serial port number.

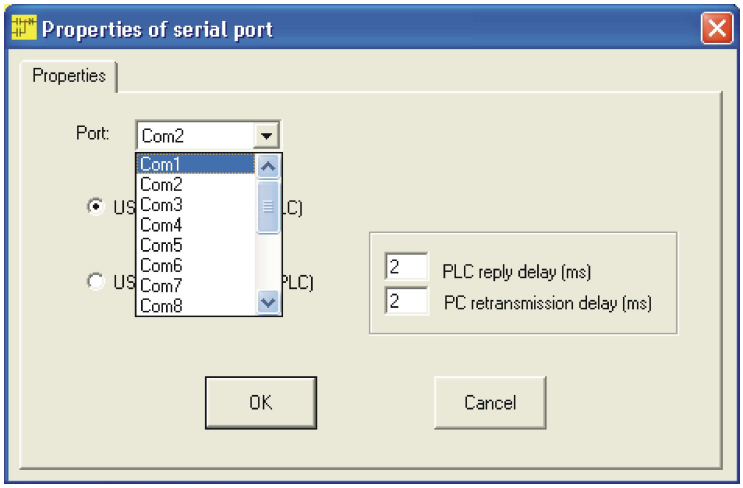


Figure 4.5: From the "Comm port" menu you go to the submenu "Com" where a drop-down box is opened

It's also possible to set which serial line has to be used for PLCs connection and/or programming (from PlProg version 4.63). Connection is possible by RS232 and RS485 (for all PLCs), programming by RS232 (for all PLCs) and RS485 (for PL260-11AD only). In the last case, select **PLC reply delay** (default 2msec) and **PC retransmission delay** (default 2msec).

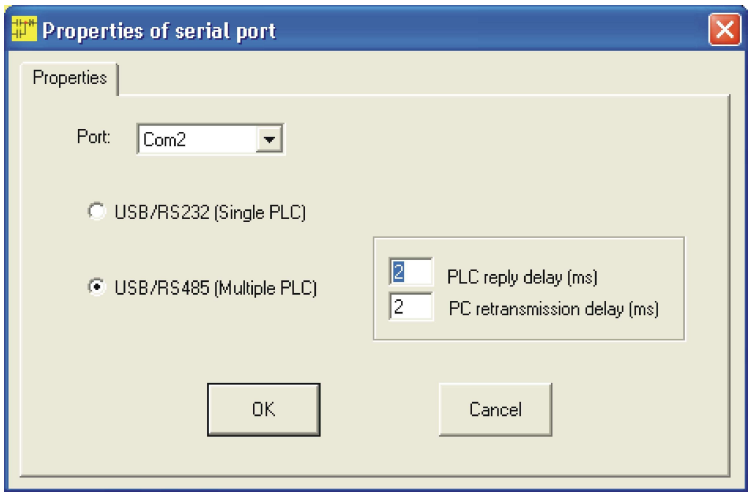


Figure 4.6: Selection connection/programming by RS485 and setting Modbus master parameters for PC

If USB/RS485 (Multiple PLC) is selected, then the number of PLC of the line to poll by PC has to be selected (up to 254 devices).

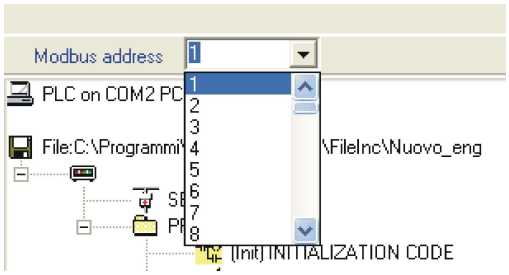


Figure 4.7: Selection of number of PLC to poll in Multiple PLC USB/RS485 modality

With the PLC powered on, make sure the right port has been chosen by clicking on the first line of the state window (see 6.4):

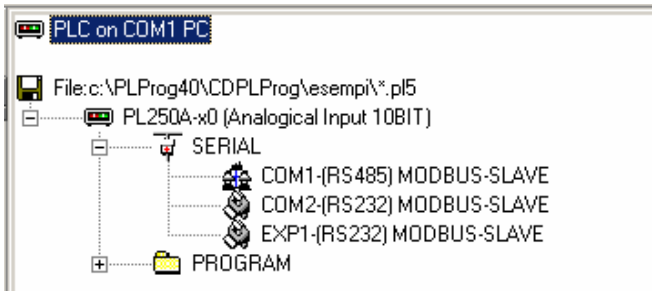


Figure 4.8: By clicking on the highlighted area, PLProg will try to communicate with the PC through Com1

If an error message is received it means the wrong port has been selected, otherwise the program will show some information on the PLC connected.

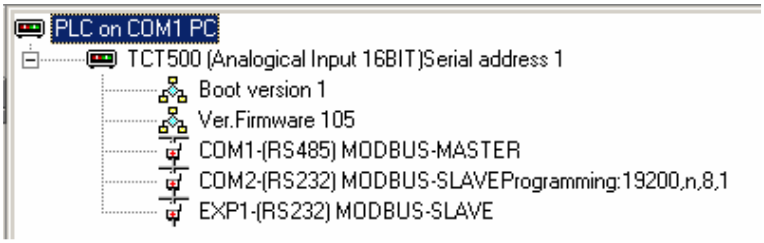


Figure 4.9: The connection and settings are correct.

Once this process has been completed you can go to computer resources by clicking on . In the folder
C:\PLProg40\Examples\TestHardware
some files named
TestHardware_*PLCName* are located:
Open the file corresponding to the device connected.

The program loaded is to be compiled and sent to the PLC by clicking on  and then  on the toolbar: if everything works correctly the output relays will begin to move and, in the case of TCT500, some writing will appear on the display.

5 How to use PLProg

This chapter gives you a general overview on how to use PLProg with a description of some common operations and steps for programming a PLC.
As seen in the previous chapters and in particular in Figure 3.1, PLProg can interface with several tools. For simplicity's sake, from now on, the device connected to the PC with the kit shall be referred to as PLC.
A thorough description of all the buttons and items on the menu is provided in the next chapter.

5.1 Setup

To create a new application the first thing to do is set up the device you intend to use. If the hardware test was successful (see par. 4.1) then the connections and serial port have been configured correctly, otherwise repeat the steps outlined in Figure 4.5 and following ones.
Clicking on the editor area while empty or on the button  on the toolbar will open the window in Figure 5.1.

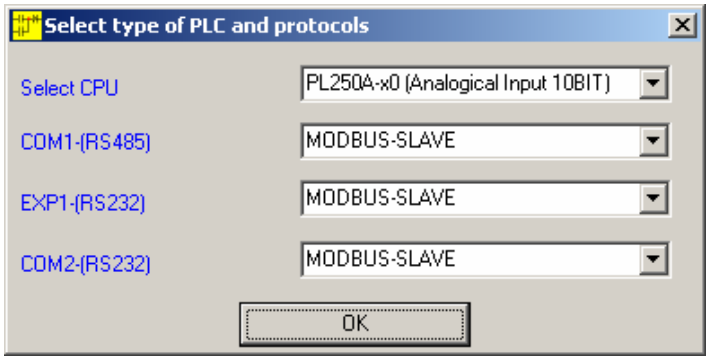


Figure 5.1: Setup window. The setup window appears every time a new file is created and can later be opened by the user with a click on the "file" area of the state window (see paragraph 5.4).

The field "Select CPU" allows you to select the type of PLC while the other three are for selecting the communication protocol of available serial ports. The options for COM1 and EXP1 are "MODBUS-SLAVE" (by default), "MODBUS-MASTER" and "CT-ANSI-MASTER". The protocol assigned to COM2 is always

“MODBUS-SLAVE” because it is the communication port used to program the PLC.
Setup can be carried out even if there is not a PLC physically connected to the PC. Furthermore, when implementing the ladder diagram, PLProg does not check the correspondence between the device connected and the one selected. If any incompatibility is detected an error message is sent whenever an attempt to transfer the program to the PLC memory is made (see par. 5.4).
The state window provides a complete overview on the situation (par. 5.4) and shows all the information related to the device connected to the PC and user settings.

5.2 Create and modify a ladder diagram

The PLProg editor window is full of features and so in order to take complete advantage of it you need an in-depth description as well as some hands-on practice. It can be considered a sheet where you place the classic objects that make up a ladder program: contacts, coils, connections and comment lines.

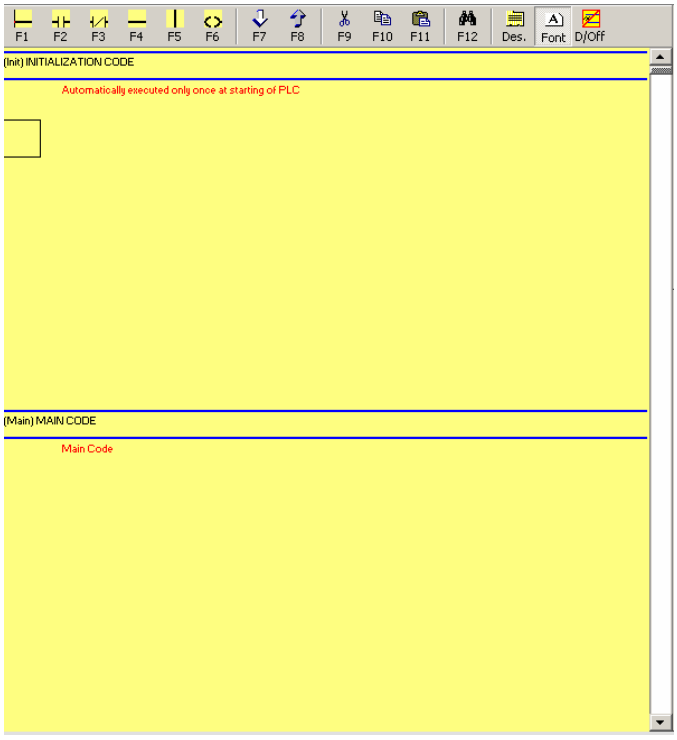


Figure 5.2: PLProg editor window.

These objects are entered using the buttons on the toolbar (Figure 6.19) or using the mouse.

It is immediately noticeable that the area of the editor reacts to any mouse movement: as illustrated in Figure 5.3 and Figure 5.4, the cursor changes according to where it is positioned to indicate that the objects need to be placed alternating contacts and coils with at least one connection part.

Figure 5.3: If the cursor looks like this then that box can host a connection.

Figure 5.4: In this case the cursor is on a box that can host a contact or relay.

To outline a connection use F1, F4 or F5 on the toolbar or keyboard or drag the cursor using the left mouse button.
To insert a contact use F2 or F3 or double click on a connection part where the cursor appears as in Figure 5.4
To insert a relay use F6.
In any case, if keys from F1 to F6 are used, the corresponding object will be placed in the outlined box (see Figure 5.5). This selection can be made with a mouse click on the sheet or with the arrow keys on the keyboard.

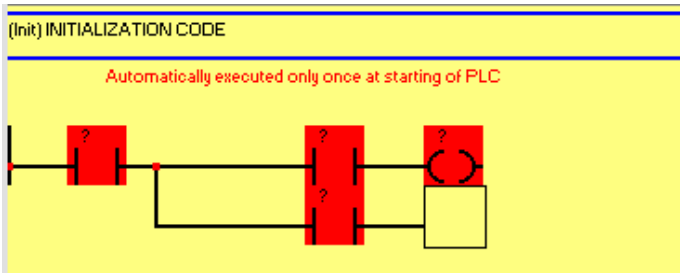


Figure 5.5: Beginning implementation of a ladder diagram. New contacts are visible as well as the box selected to insert a new object.

This first example of a ladder diagram means nothing until some functions or PLC parameters are associated to the contacts and relays. A double click on the contact opens the window “Insert/Modify contact”.

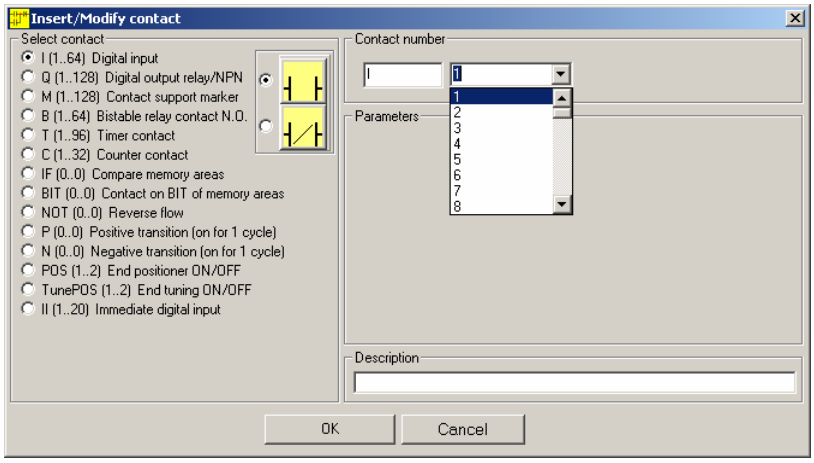


Figure 5.6: This window is for selecting the type of contact (Normally Open or Normally Closed), the associated flag (I, Q, M…) and if necessary a description.

A double click on a coil opens the window “Select/Modify coil”.

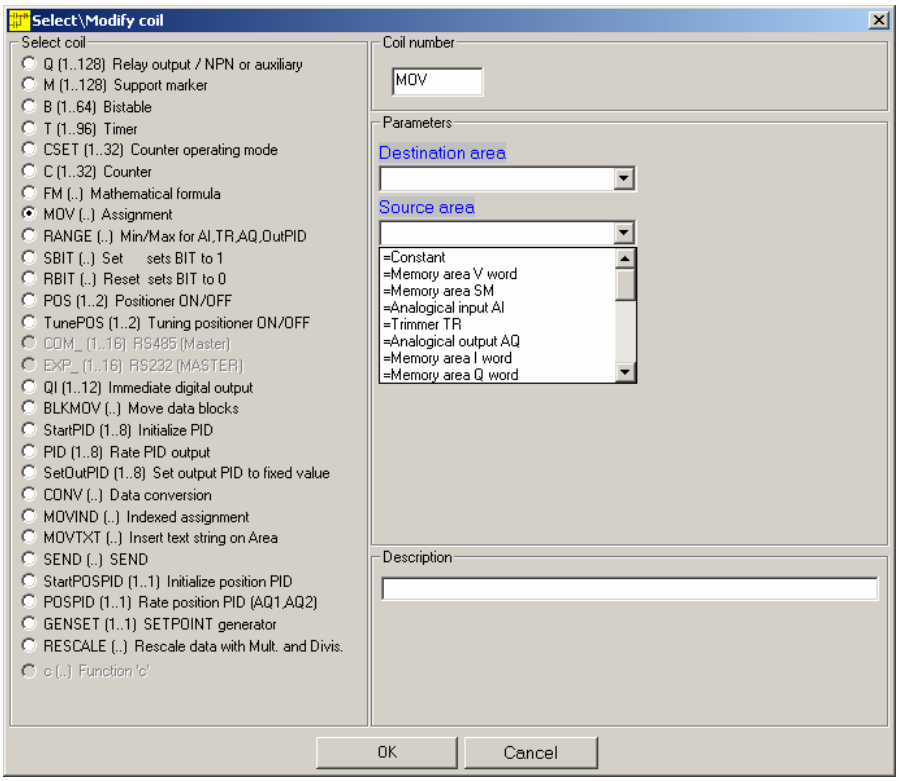


Figure 5.7: This window is for associating a function with the needed parameters and a description to a coil.

In the example of Figure 5.8 three PLC inputs have been associated to the contacts and one output and assignment function have been associated to the relays.

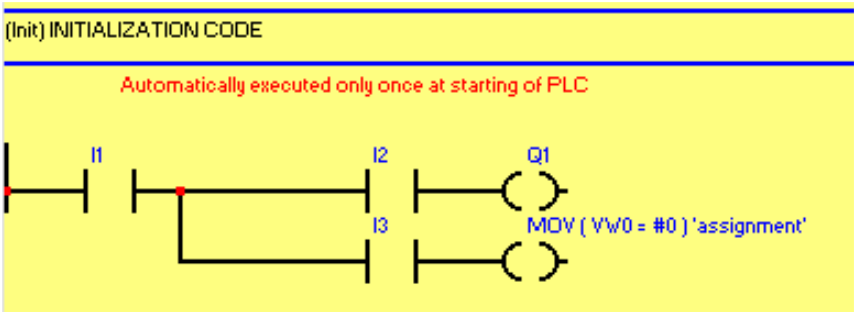


Figure 5.8: Complete diagram of features and parameters inserted.

The diagram can be extended by adding new lines of code with the same procedure described above and can also be changed. With a double click on the relays and contacts the windows in Figure 5.6 and Figure 5.7 open for changing functions and parameters while with a right mouse click you can cancel.

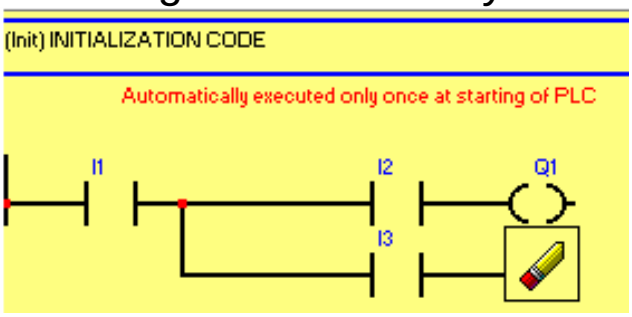


Figure 5.9: Cancelling an object. The cursor changes appearance.

A contact or coil can be dragged horizontally with a drag-and-drop movement.
It is possible to mark an area of a diagram with a title to render the program more legible and organised.
A double click on an empty area of the editor opens the window in Figure 5.10.

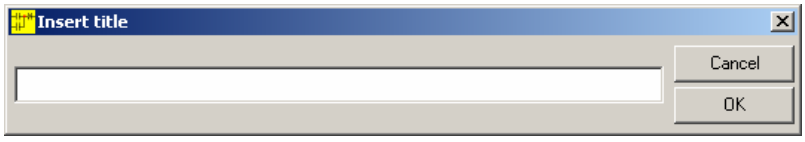


Figure 5.10: Insert a title and click OK.

If titles are placed on the far left of the sheet then they become actual bookmarks which is particularly useful in programs of considerable length.

In that case PLProg places a shortcut in the state window (see par. 6.4) from where at any time you can jump to the area of code that follows the title.

5.2.1 Selecting and copying a block

PLProg allows you to copy (cut) a block of lines to paste onto another part of the program or to save in another file (see paragraphs 6.1.1.5 and 6.1.1.9).
Due to the importance of this feature, a paragraph is dedicated to it and previews the description of the editor toolbar which will come in par. 6.3.
The complete procedure entails four steps:

1. Click on first line – first column of the block you want to copy (cut);
2. Press F10 (F9) or with the mouse on the editor toolbar (Figure 6.19), the cursor changes appearance as shown in Figure 6.6;
3. Click on the last line of the block you want to copy (cut) or save, the column is irrelevant. The cursor turns back to its original appearance;
4. Click on the line from which you want to position the block obtained.
5. Press F11 to paste.

If there are not enough empty lines to host the block, a partial copy will be made without overwriting those already existing.

5.3 Opening and closing a ladder diagram

So far the process to create a ladder diagram has been shown. If you want to start over from the beginning click on  in the toolbar corresponding to the menu item ‘File-New file’.
If instead you want to save the diagram and settings click on  or ‘File-Save file’ in the menu. When you save for the first time a window opens so you can give a name to the file. The extension is always ‘pl5’.
 (‘File-Load file *.pl5’) opens a window where you can select and open a saved file.

5.4 Compilation

To put into practice the logic represented by a ladder diagram you need to make a compilation and load the file into the memory of the PLC. The  on the toolbar launches the compilation process. The result is shown in the state bar of the program, under the editor window (see Figure 6.8).
After compilation it is possible to transfer the program to the PLC: check the physical connection and the serial port ("Comm port" in the menu), making sure the device connected is powered and corresponds to the one selected for the program (in the state window, Figure 5.12), then click on  .

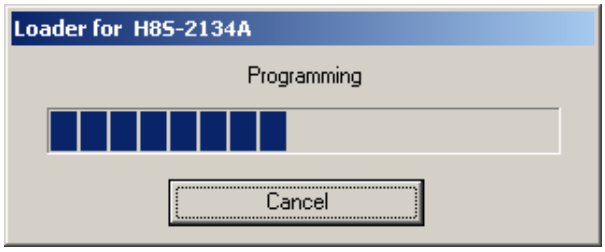


Figure 5.11: Programming the PLC. The entire operation takes just a few seconds.

5.5 Testing hardware and software

Before running the system it is always a good idea to test the PLC program with a simulation. PLProg possesses three tools in particular to check the behaviour of a device and the program loaded: debug, memory areas' monitor and PLC test.

5.5.1 Debug

The debug feature allows you to test the program loaded on the PLC showing the state of contacts and coils on the ladder diagram.
To debug, the PC needs to be connected to a device of the same type specified in the program and communication needs to be active. The information concerning these conditions is in the state window (see paragraph 6.4).

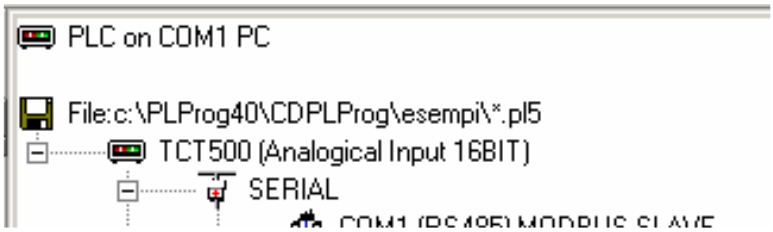


Figure 5.12: The user has selected the PLC TCT500 but a connection has not yet been set.

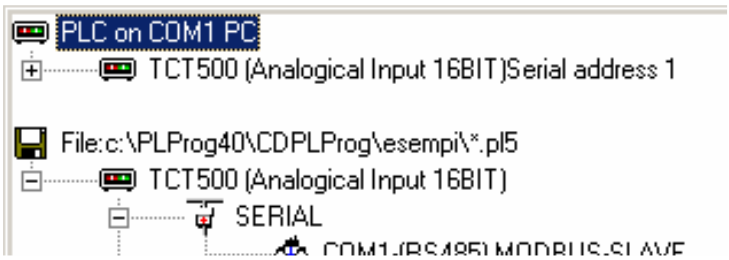


Figure 5.13: With a click on the highlighted area the program detects the PLC. In this case it is possible to continue debugging.

Naturally, the ladder diagram shown needs to correspond to the program loaded in the PLC memory to make the simulation valid.

By clicking on  you go into debug mode. This button will appear disabled if areas of PLC memory are being monitored (see Figure 5.17) because the two functions cannot run at the same time.

From this moment on, the active contacts and coils will be highlighted with a blue box,

Figure 5.14: Active coil.

Furthermore, contacts with IF condition show the values of the compared expressions.

Figure 5.15: The same contact seen in normal mode is being debugged when the condition is verified.

Debugging blocks most PLProg functions. To exit click again on  .

5.5.2 Memory areas monitor

In some cases it is useful to view PLC memory contents dynamically: the window shown in Figure 5.16 and Figure 5.17 collects and organises the various areas into tables. A double click on the grid begins and interrupts.

The table “USER” is empty at first; the user can insert a reference to the location of the memory to be monitored.

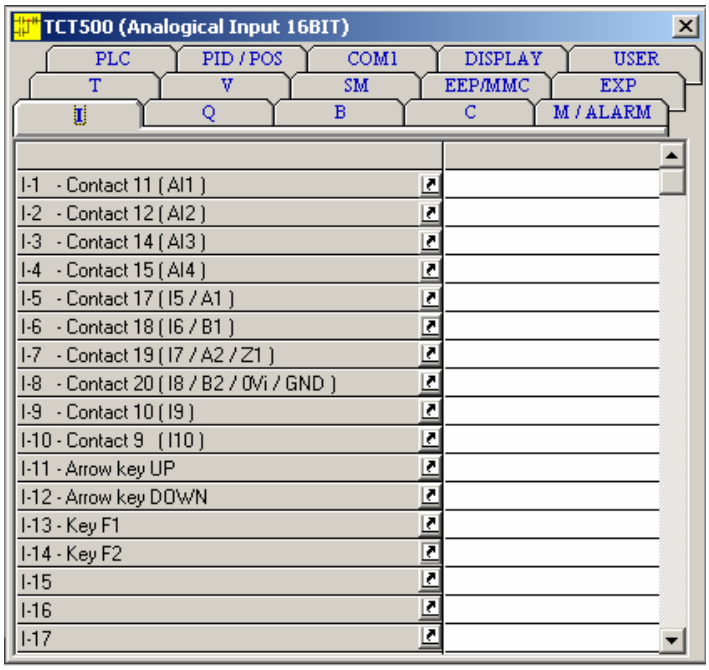


Figure 5.16: Double click on the grid to start reading.

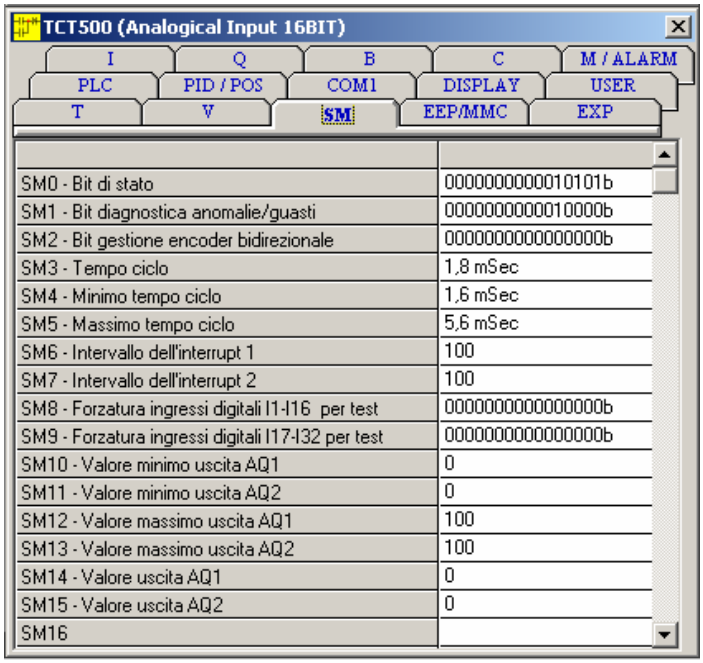


Figure 5.17: Modbus communication in progress. A double click on the grid interrupts reading.

When communication with the PLC is not active (as seen in Figure 5.16), the user can insert a variable in the “USER” table or force a value: a click on  opens the window shown in Figure 5.18 through which you can carry out these actions.

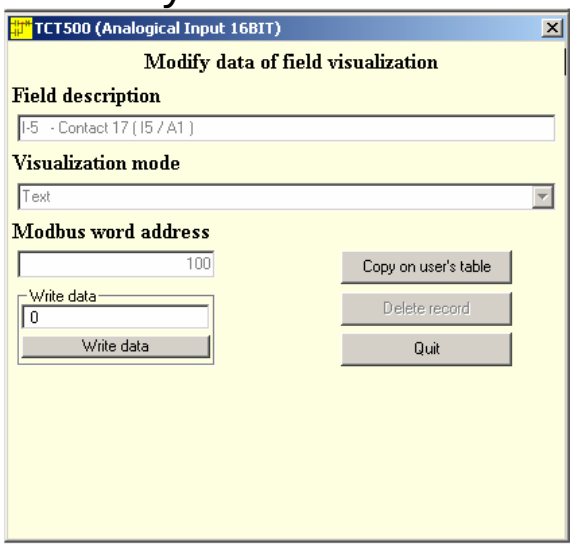


Figure 5.18: “Field description” and “Delete record” can be changed only in the “USER” table, “Write data” is not available for the “PLC” area table.

5.5.3 Test PLC

This tool is able to test PLC functions regardless of the program loaded.

Clicking on  in the toolbar opens the window shown in Figure 5.19 (not available for TD320 and TD240 terminals) thanks to which it is possible to check graphically that the communication is working properly, force inputs and outputs and monitor digital and analogue variables.

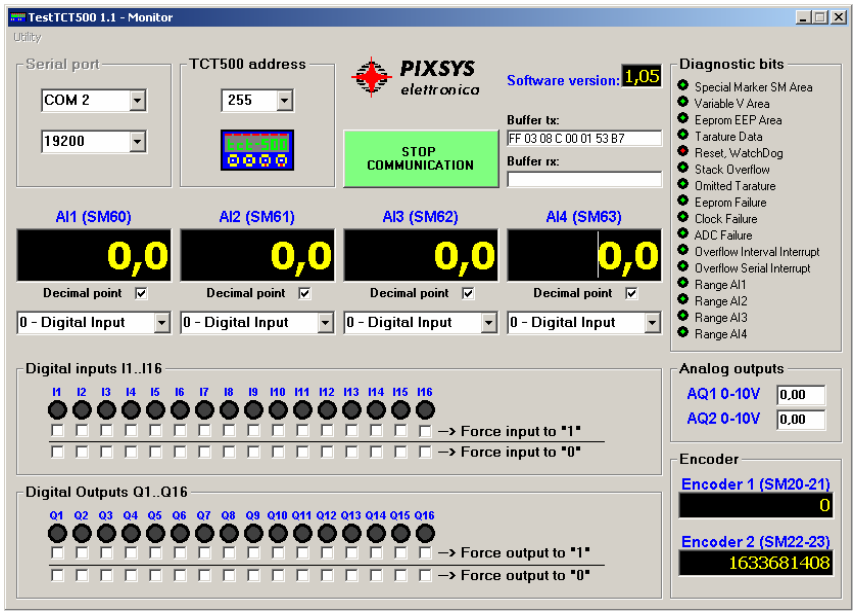


Figure 5.19: Test program for the PLC, in this case TCT500.

5.6 Print

Considering the large amount of information linked to a ladder program, the printing process (launched by the button ) always starts with selecting what needs to be printed.

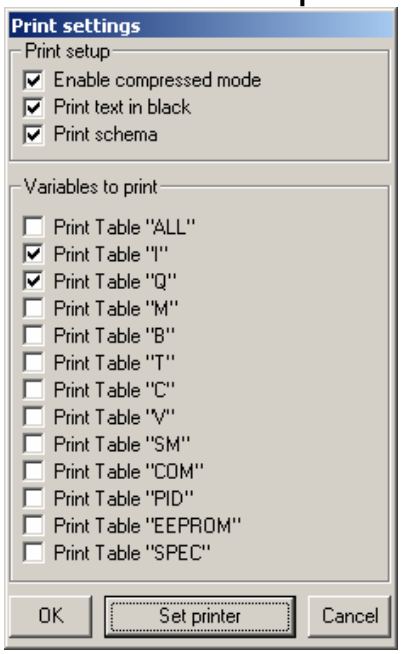


Figure 5.20: Print settings

“Enable compressed mode”: prints the ladder diagram without empty lines.

“Print text in black”: also the strings that are not black in the ladder diagram (comments, formulas…) are all printed in black.

“Print schema”: it is possible to disable this option when printing the ladder diagram.

“Variables to print”: by selecting the fields desired, you can print a table with a list of all the areas of memory used in the program in the style of the window “Variables’ global table” (see paragraph 6.5).

6 Program guide

The goal of the previous chapters is to provide an overview of the main features of PLProg. That is why advanced functions and accessories have been left out.
To complete the description of the software, in this chapter we will review all the windows, menu items and buttons of the program.

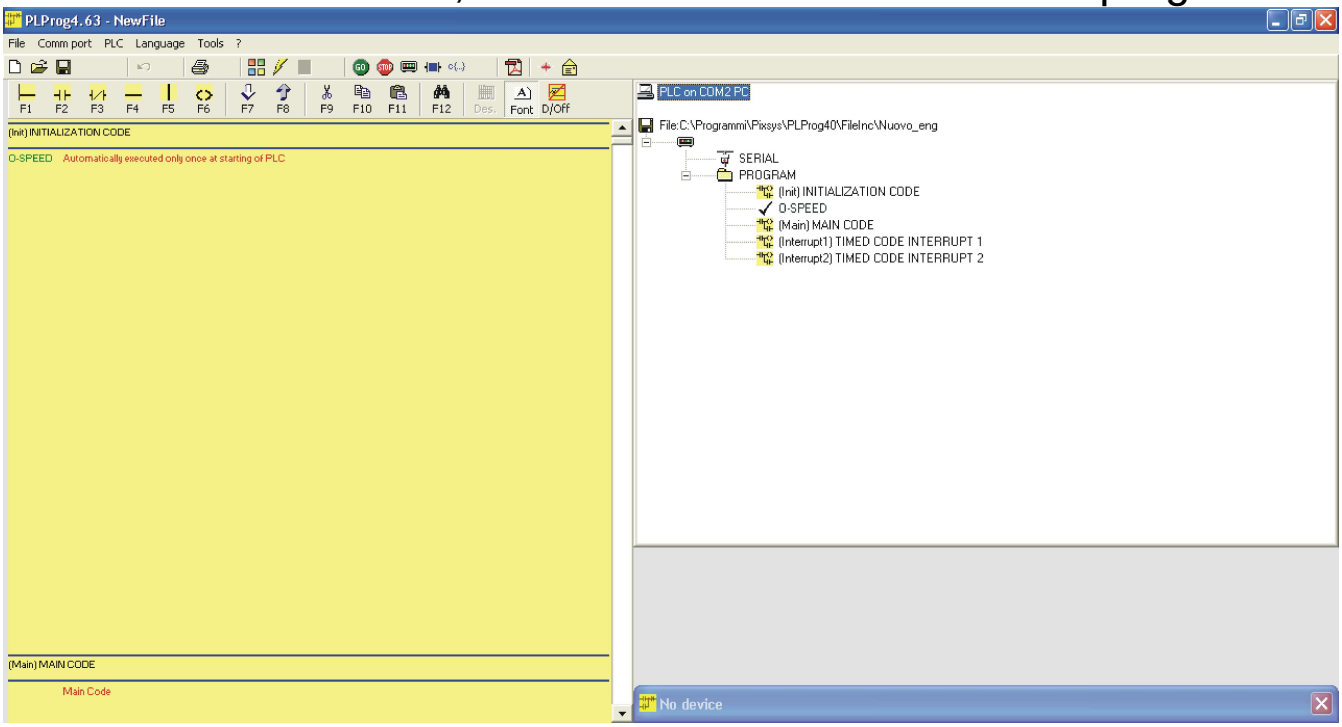


Figure 6.1: Opening view of PLProg

6.1 Main menu

The main menu contains all the tools necessary to make the program work.

Figure 6.2: Detail of main menu.

Figure 6.3: The toolbar. Some of the buttons perform actions found in the main menu.

6.1.1 Files

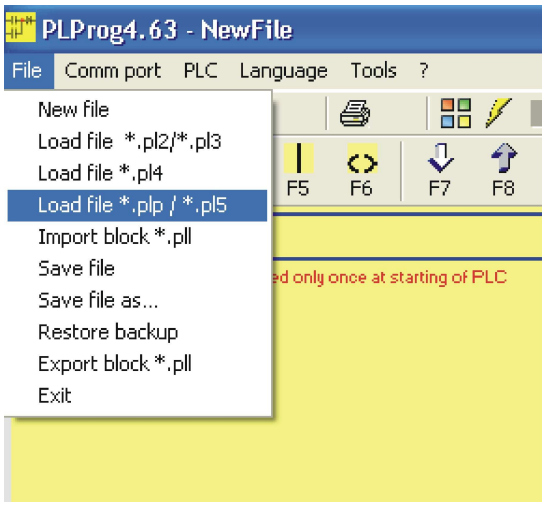


Figure 6.4: "File" tool of main menu

The item "File" presents all the possible operations related to opening and saving entire programs or parts of programs. The buttons on the toolbar:

Figure 6.5: Buttons on the toolbar

Correspond to "New file", "Load file *.plp" and "Save file".

6.1.1.1 New file

Prepares the environment to create a new ladder diagram, in particular it opens a window to configure the PLC where you intend to load the program.

6.1.1.2 Load file *.pl2/pl3

To access programs created with previous versions of PLProg.

6.1.1.3 Load file *.pl4

As above.

6.1.1.4 Load file *.plp / *.pl5

Files with the extension 'pl5' have been created with the current version of PLProg. In comparison to previous versions, this format saves not only the ladder diagram but also further useful

information such as names associated by the user to memory areas (see par. 6.5), PLC settings (Figure 5.1) and the “USER” table (see par. 5.5.2).

6.1.1.5 Import *.pll block

It is possible to save certain lines (a “block”) of a ladder diagram as a file with the extension “pll” and so making it easily accessible and recyclable for other projects.

To import a saved block (see par. 6.1.1.9), the following steps are to be followed:

- Command from menu “” [import block *.pll];
- Select the “pll” file desired and at this point the cursor changes to:

Figure 6.6

- Click on the area of the diagram where you want to insert the block; if there is not enough room this operation will not overwrite program lines but only the available space will be filled.

6.1.1.6 Save file

Save the program as a “pl5” file; files open as *.pl2, *.pl3, *.pl4 and *.pl5 are automatically converted to this format. In addition to the ladder diagram, the PLC settings and the “USER” table in the monitor window of the memory area are also saved.

6.1.1.7 Save as…

As above with the possibility to assign or change the file name.

6.1.1.8 Restore backup

PLProg automatically creates backups of files you are working on. The frequency of automatic saves can be set in the menu “Auto-save options” (see par. 6.1.5.3).

The command “Restore backup” allows you to load these files which are otherwise hidden.

6.1.1.9 Export *.pll block

Saves a part of a ladder diagram in a file and is only possible if a block has been selected (see process to copy a block at 5.2.1).

6.1.1.10 Exit

Closes the application.

6.1.2 Communication port

6.1.2.1 Com

Opens a window where you select the communication port between the PC and PLC assuming it is coherent with physical connections.
N.B.: the baud rate of the communication between the PC and PLC is set automatically.

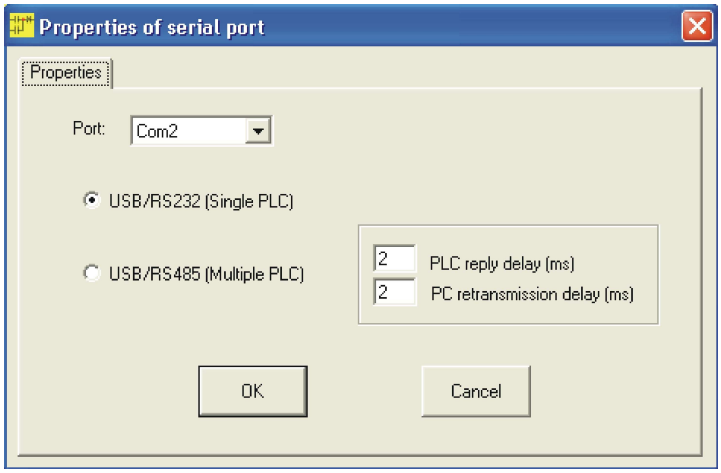


Figure 6.7: From the “Comm port” menu you go to the Properties

6.1.3 PLC

Management operations of PLC connected to PC.
The following buttons correspond to the items on the menu:

for “RUN” and “STOP”,

for “Compile all” and “Send to PLC”.

6.1.3.1 RUN

Reruns the program for the PLC if it is stopped.

6.1.3.2 STOP

Stops the program for the PLC.

6.1.3.3 Compile all

From the ladder diagram, it creates a binary file to put into the PLC memory. The outcome of the compilation is summarised in the state bar at the bottom to the left.

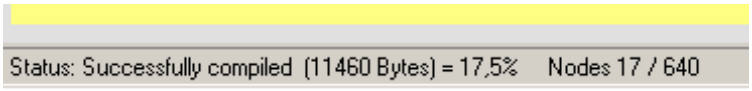


Figure 6.8: PLProg state bar, under the editor window.

6.1.3.4 Send to PLC

Programs the PLC through the serial connection. This operation is allowed only after a compilation.

6.1.3.5 PLC type

Opens the window:

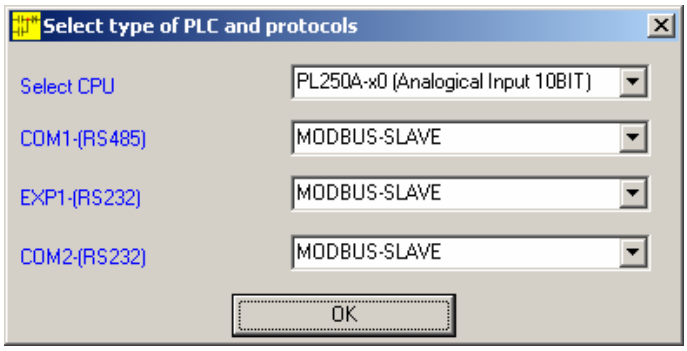


Figure 6.9: To configure the type of device and serial protocols. COM2 is the programming port for the PLC and so it is set as MODBUS-SLAVE.

6.1.3.6 Adjust PLC clock

To synchronise the PLC clock with the PC clock.

6.1.3.7 Create memory card

Programs can be loaded into PLC memory using a memory card as well as PLProg. This command works in a similar way as “send to PLC” (see par. 6.1.3.4). Before beginning, PLProg will show a photo of a memory card and the correct positioning.
To program a PLC, turn off the power, insert the card and turn it on again.

6.1.3.8 PLC maintenance

Set of advanced PLC operations.
The user does not need to worry about the first two “Update firmware version” and ”Update BOOT version”, as PLProg will carry out those actions automatically when necessary.
“DataSave (Read/Write–VW-EEPROM)” opens this window:

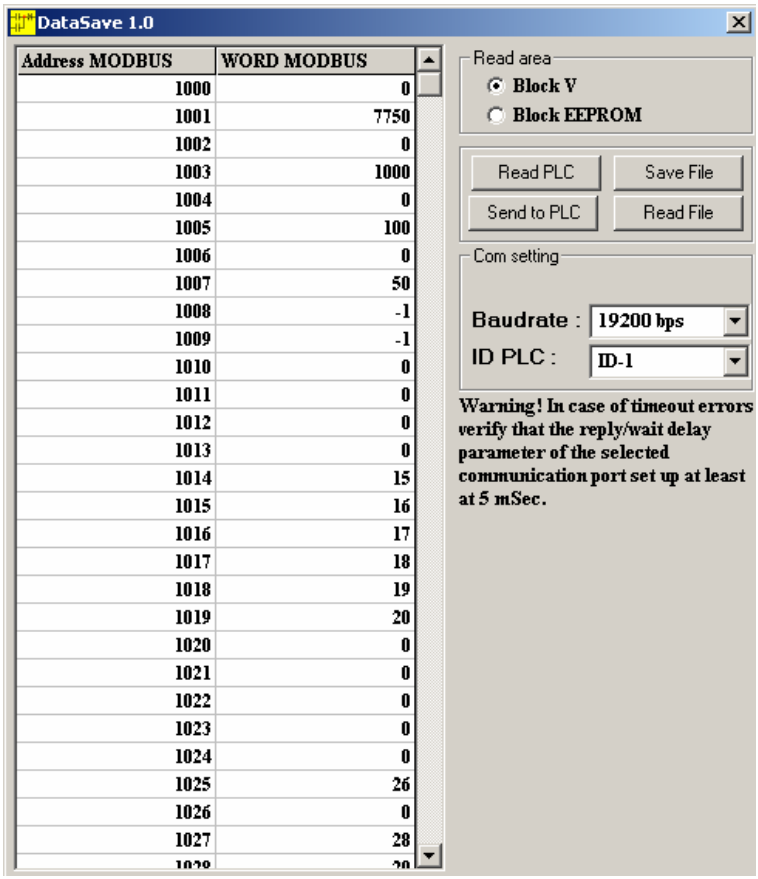


Figure 6.10: Window for VW and EEPROM areas reading and writing.

With this tool the entire contents of VW and EEPROM areas on the PLC can be saved in a file or overwritten with the contents of a file. For instance when you want to clone a device.

6.1.4 Language

The two options available are Italian or English.

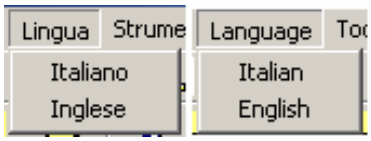


Figure 6.11: The language menu.

6.1.5 Tools

With the tools menu the user can modify the system and configure certain features in the development environment.

6.1.5.1 Editor options

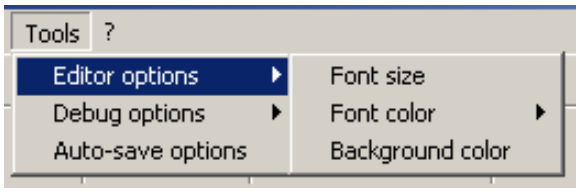


Figure 6.12: Editor options

It is possible to make settings according to your preferences

- **Text format** (type and size of characters);
- **Text colour**, for all types of editor writing (functions, descriptions, code area titles, bookmarks);
- **Background colour**, this option is not available for Win98 operating systems.

6.1.5.2 Debug options

Figure 6.13: Debug options

- Shows active lines
    - If this option is chosen also the connections following active contacts and coils are shown. The figures (Figure 6.14, Figure 6.15) show the same portion of the diagram in the two cases.

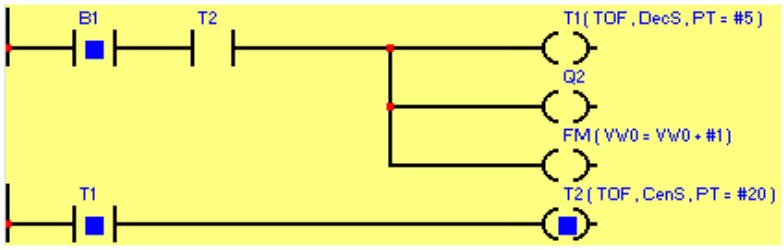


Figure 6.14: option “Highlight active lines” not selected.

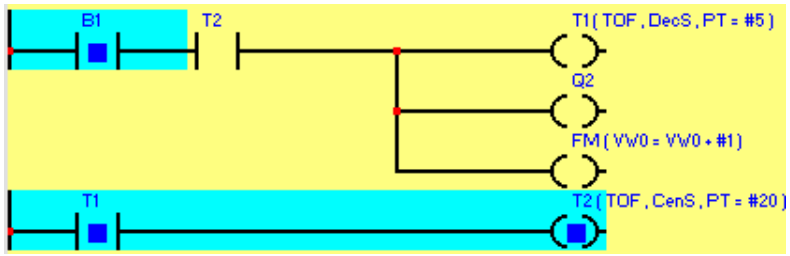


Figure 6.15: option “Highlight active lines” selected.

- Maintain NC symbol
    - The initials NC indicate logic contacts considered active when the corresponding physical contacts are open. When debugging you can choose whether to view them with the same symbol as NA contacts or keep the symbol used in the development phase.

Figure 6.16: Active NC contact. “Maintain NC symbol” not selected on the left and selected on the right.

6.1.5.3 Automatic save options

Set frequency of backup file automatic saving.

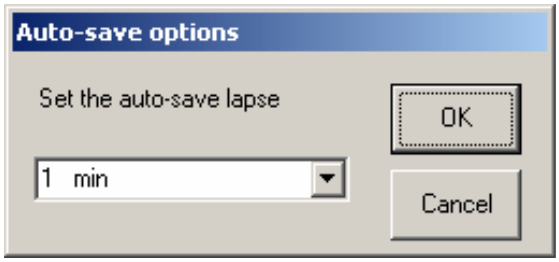


Figure 6.17: Window to set automatic saving.

6.1.6 ? (Help)

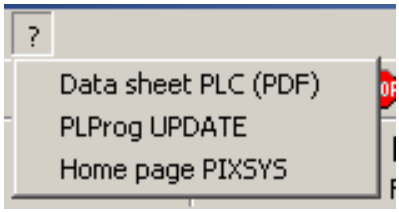


Figure 6.18: "?" menu

6.1.6.1 PLC manual (PDF)

Opens a manual for the PLC or device. If one has not yet been selected it has not effect. If there is no .pdf reader installed, PLProg will automatically install one.

This command corresponds to  on the toolbar.

6.1.6.2 PLProg update

If there is an Internet connection, the most recent versions of PLProg will be downloaded automatically.

6.1.6.3 PIXSYS home page

Go to the Web site http://www.pixsys.net.

6.2 Toolbar

In this chapter an overview of the buttons on the toolbar is provided, in particular the ones that do not correspond to items on the main menu are described.

 New file (see par. 6.1.1.1)

 Open file (see par. 6.1.1.4)

 Save file (see par. 6.1.1.6)

 Undo, cancels changes made to ladder diagram.

 Opens the print window (Figure 5.20).

 Compiles the program (see par. 6.1.3.3).

 Sends the program to the PLC (see par. 6.1.3.4).

 Opens the variables table (see par. 6.5)

 Start PLC (see par. 6.1.3.1).

 Stop PLC (see par. 6.1.3.2).

 Launches test program for connected PLC.

 Debug mode.

 Opens the manual for the selected PLC (see par. 6.1.6.1).

 Connects to the Pixsys home page (see 6.1.6.3).

 For any suggestions and clarifications via e-mail on how PLProg works.

6.3 Editor toolbar

Following is a brief explanation on the toolbar buttons (Figure 6.19).

Figure 6.19: The editor toolbar

F1 : inserts a new line begin symbol in the diagram.
F2 : new contact normally open.
F3 : new contact normally closed.
F4 : horizontal connection, if the box selected is a vertical connection, a curve is outlined.

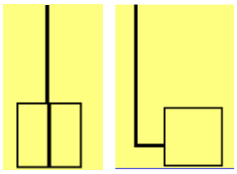


Figure 6.20: effect of the F4 function on a vertical connection.

F5 : vertical connection, if the box selected is a horizontal connection, a node is outlined.

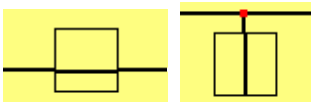


Figure 6.21: effect of the F5 function on a horizontal connection. This pertains only if the cursor appears as in Figure 5.3.

F6 : new coil.
F7 : starting from the line of the box selected, it moves the entire diagram down a line.
F8 : starting from the line of the box selected, it moves the entire diagram up a line.
F9 : cuts a box or a block of lines and keeps it on the clipboard to paste it onto another part of the program.
F10 : copies a block of lines (see par. 5.2.1).
F11 : pastes the last block of lines copied (cut).
F12 : opens a search window.

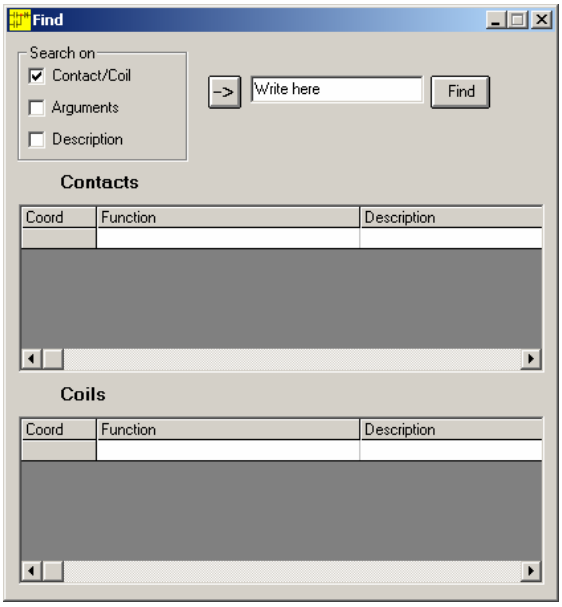


Figure 6.22: Write the string to search for in the text field. The first column of the grid always contains the resulting line-column coordinates.

With the three options available in the "Search on" field the user can choose to search for a string from the names and functions of contacts and coils, from the arguments of functions or from the titles and descriptions. These options are not mutually exclusive.
The search does not distinguish between lower-case and upper-case characters.
If a ladder diagram contact or coil is selected on the editor (mouse click) you automatically produce in the text field the name associated to the function using the right arrow button  .
Searching using the option Contact/Coil works only with complete names. For instance, to find contact I11, it is not enough to insert the string "I" or "i1" but you need to write "i11". The options "Arguments" and "Description" accept any type of string. For instance, searching for the string "sm", you may find **SMW0.1** and **SMW1.2** among the subjects and "tran**sm**ission" among the descriptions.

Des.: enable/disable description view mode (Figure 6.27 and Figure 6.28).
Font: if enabled, select the font desired (see par. 6.1.5.1), otherwise the font is default.
D/Off: hides the description associated to contacts and coils.

6.4 State window

Contains all the information on the device connected to the PC and on the file you are working on. The window is organised so that data is viewed in detail or hidden by clicking on '+' and '-' next to the title

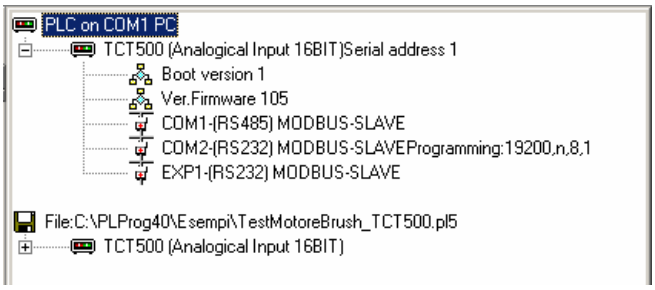


Figure 6.23: State window. The settings shown are for the connected PLC.

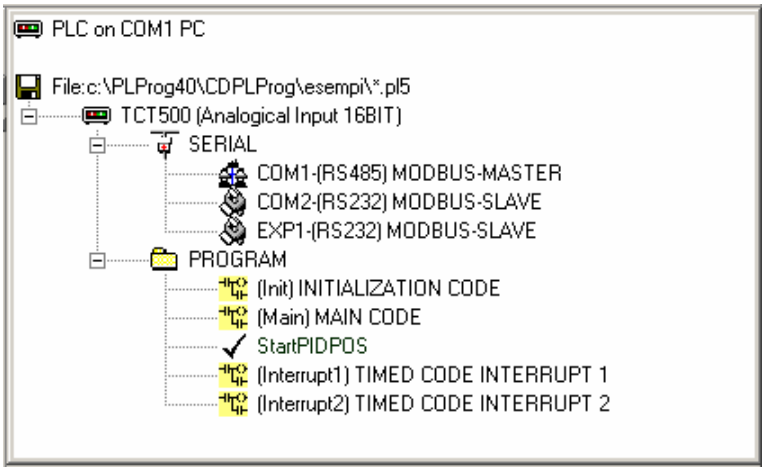


Figure 6.24: The window refers to the same situation as the previous figure. In this case the program settings are shown.

From the top these fields are found.

- The settings of the PLC physically connected to the PC (Figure 6.23).

- The PLC selected by the user and the settings for the serial ports (Figure 6.24). In general the same ladder diagram cannot be compiled and transferred onto two separate PLCs. To change PLCs or the serial settings, open the window shown in Figure 5.1 with a click on a line in this field.

- The index of the ladder diagram: with a click on a line in this field you can view the parts of the programs that follow the corresponding title. As explained after Figure 5.10, you can insert new items marked with a ✓ symbol to distinguish them from those created automatically by PLProg (symbol).

6.5 Variables table

The user can decide to assign to flags and program memory areas a name for the application under development. The button opens this window:

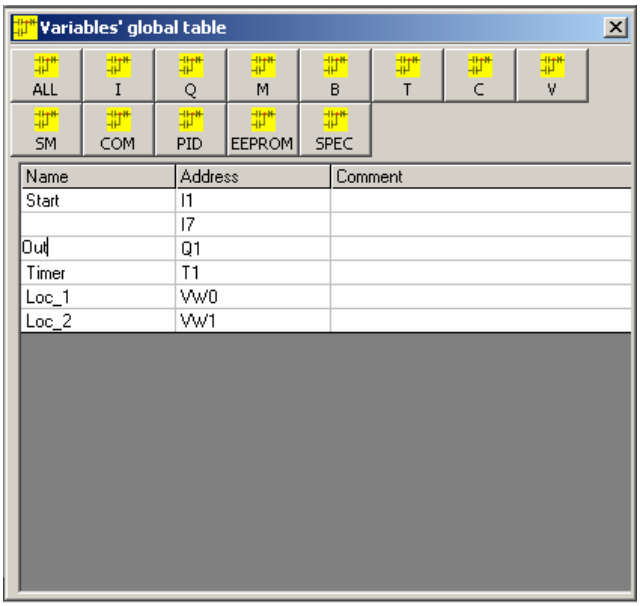


Figure 6.25: The fields "Name" and "Comment" can be changed. Mouse clicks, arrow keys, tab, return, space and backspace can be used on the grid.

Names assigned to "variables" can be viewed on the ladder diagram as well. With the button

Figure 6.26: Button to enable/disable description mode

you can switch from one view to another. The two figures below show the same portion of the diagram, the names are those assigned in Figure 6.25.

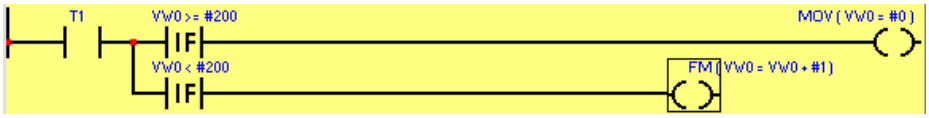


Figure 6.27: Normal view mode

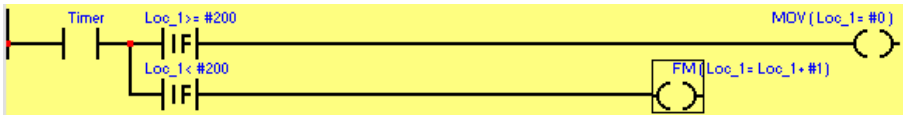


Figure 6.28: Descriptive view mode

7 Examples of ladder programming

7.1 Self-holding

Self-holding is one of the most common logic diagrams in ladder programming. It is a mechanism able to maintain an output active even when the activation signal has stopped.
Figure 7.1 and Figure 7.2 show an example.

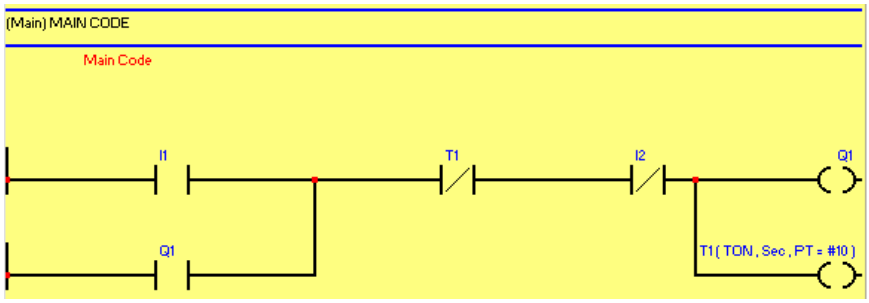


Figure 7.1: Contact I1 is the start, when it is closed Q1 is active. To deactivate the output, the normally closed I2 contact needs to be opened or it is necessary to wait 10 seconds for the timer contact to open.

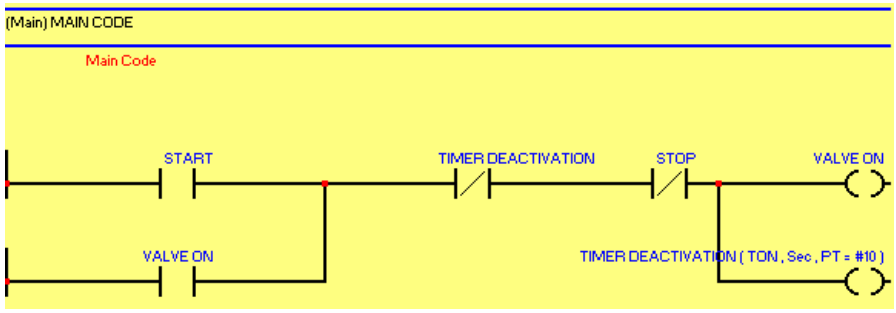


Figure 7.2: Diagram in descriptive mode (view of names assigned by user).

7.2 Adjust temperature with analogue output

In the example below, temperature is adjusted using a PT100 as sensor and a voltage-driven device as actuator.
The first step is to configure the hardware, naturally with the device off, according to the specifications in the PLC manual.
Then, in the initialisation code, select the type of analogue input using the relative fields in the Special Marker area: for a PT100 physically connected to inputs AI3 and AI4 it is necessary to write

a constant #13 in the word SMW42 and SMW43, for other selections see the PLC manual.
The set point for the process is taken from one of the two trimmers on the PLC normalized between 0 and 30℃.
The other trimmer is an offset for the input and is variable between –3 and +3℃.
It is assumed that continuous 0-10V output is desired: in accordance with the coil output that does the PID regulation, the AQ1 output is set between 0 and 10000.
Lastly, the PID action needs initialising, in this case with only one proportional component. Figure 7.3 shows all the initial configurations described above.

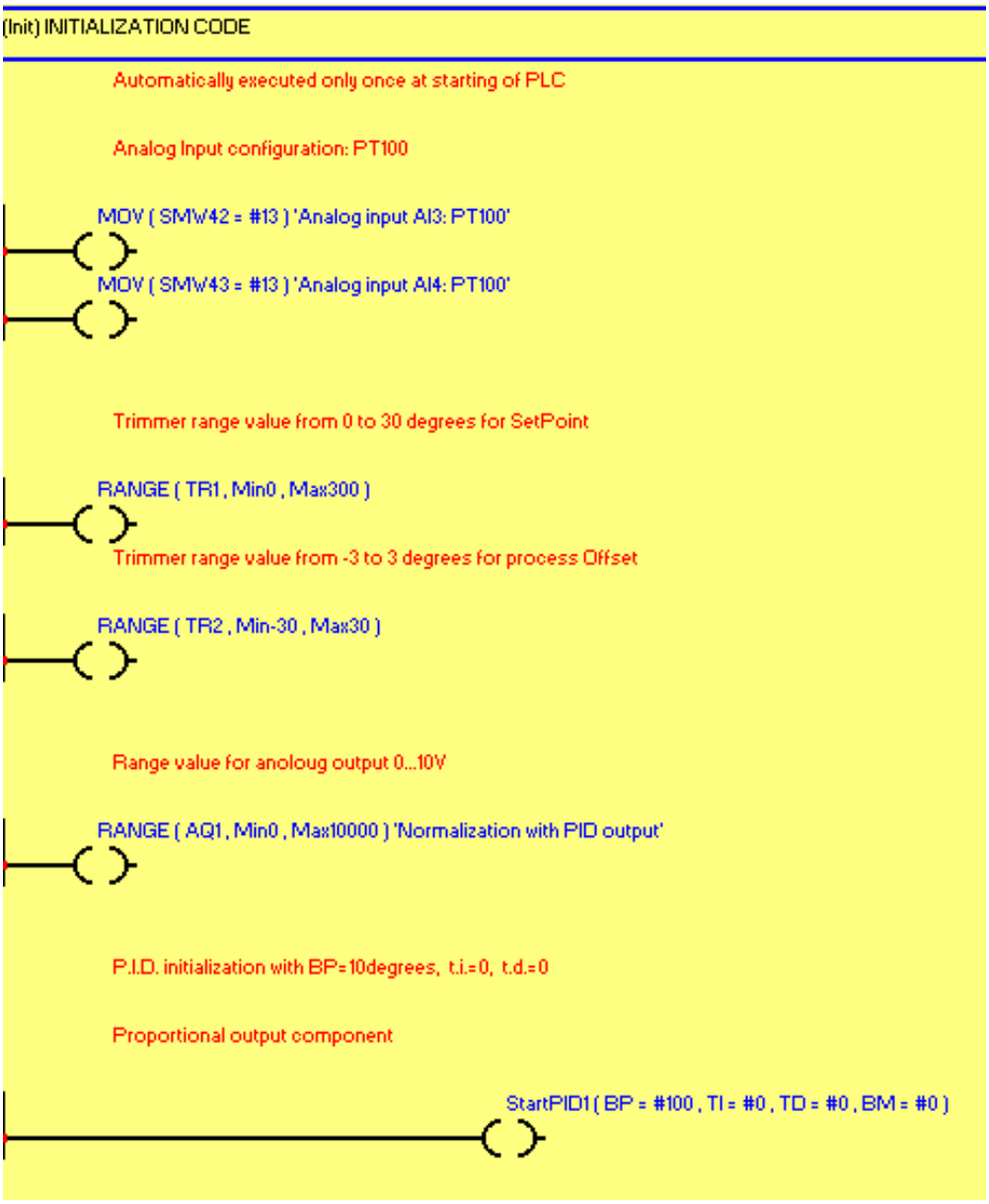


Figure 7.3: Initialisation code for the application. Note: the values given to the RANGE function applied to the trimmers are multiplied by 10 compared to the desired value in degrees. This is because when an analogue input reads a temperature, it is returned multiplied by 10 to obtain a precise reading by a tenth of a degree.

For this application it is sufficient to calculate the PID action once per second, the resulting value is sent to the AQ1 analogue output, available among the clamps specified in the PLC manual. Figure 7.4 shows the main part of the diagram.

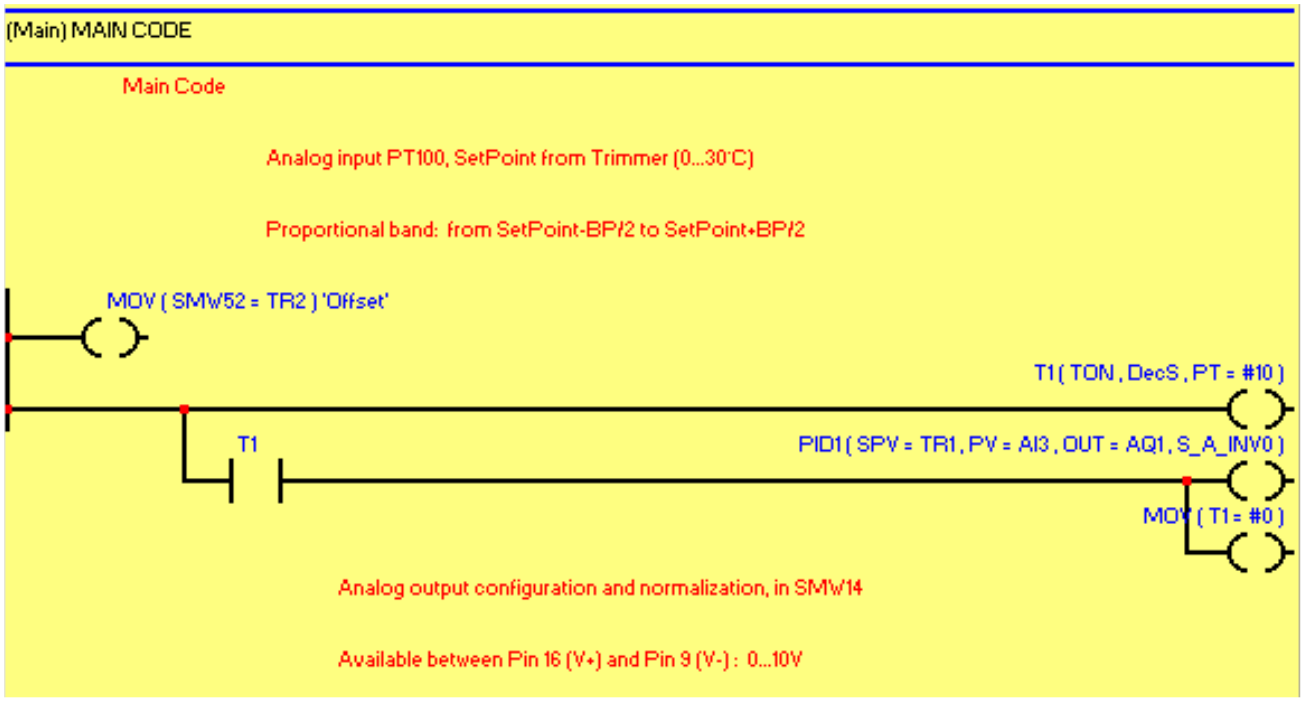


Figure 7.4: At one second intervals, the T1 contact is activated for an instant during which the PID1 function sets the value of the AQ1 analogue output.

7.3 Adjust temperature with relay output

The application of the previous example can be changed slightly to make the actuator ON/OFF. To continue using the PID function, the output needs to be modulated as a ratio between relay off-time and on-time in a set cycle.

The initialisation phase is the same as in Figure 7.3 except that now it is no longer necessary to set the RANGE of the analogue output.

In the main code, the value calculated by the PID function is not sent directly to the AQ1 output but is saved in order to calculate the ratio between relay on-time and cycle time.

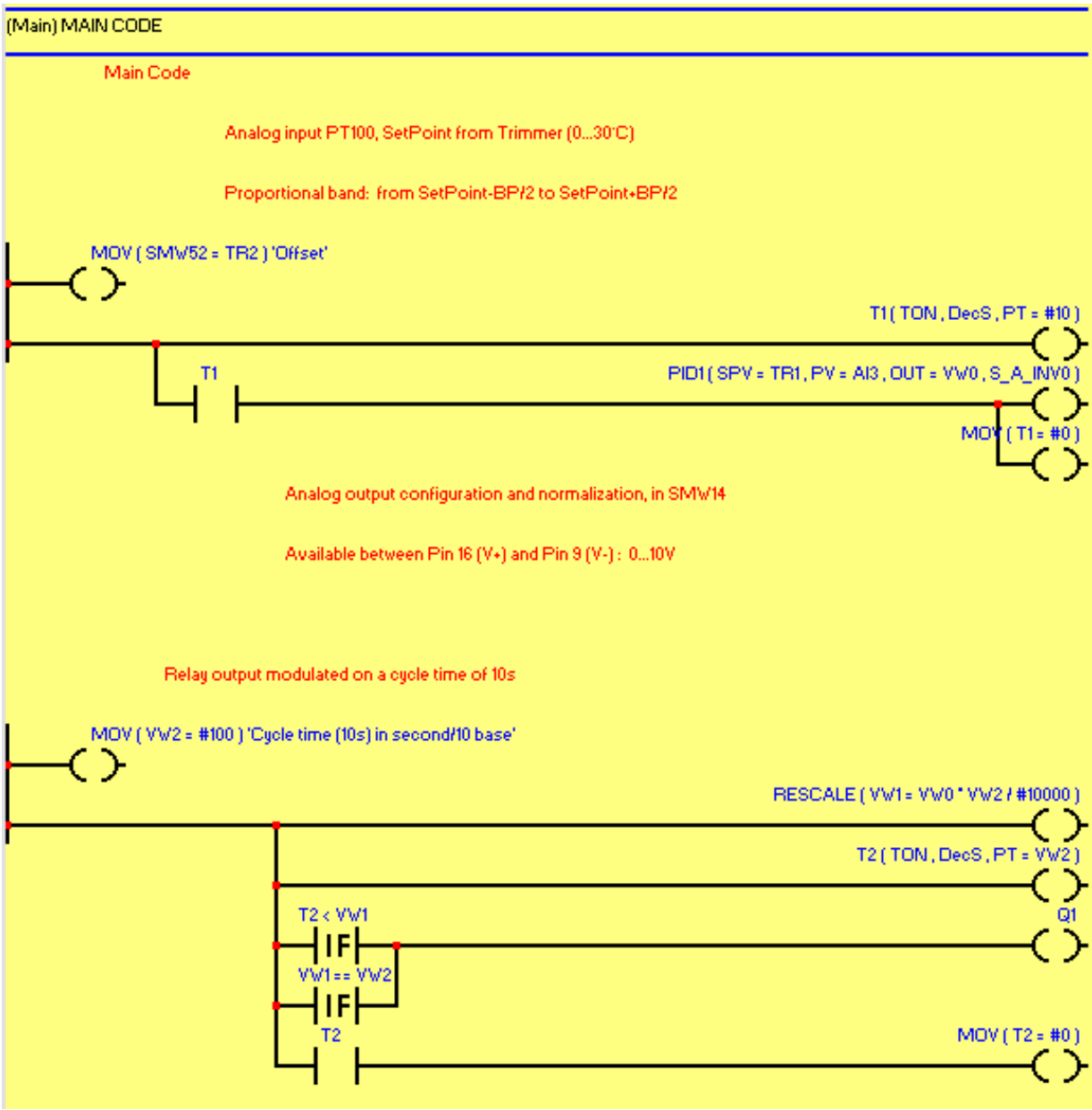


Figure 7.5: The PID output is copied into the VW0 memory. VW1 is the percentage relative to the VW2 cycle time when the Q1 relay needs to be closed.

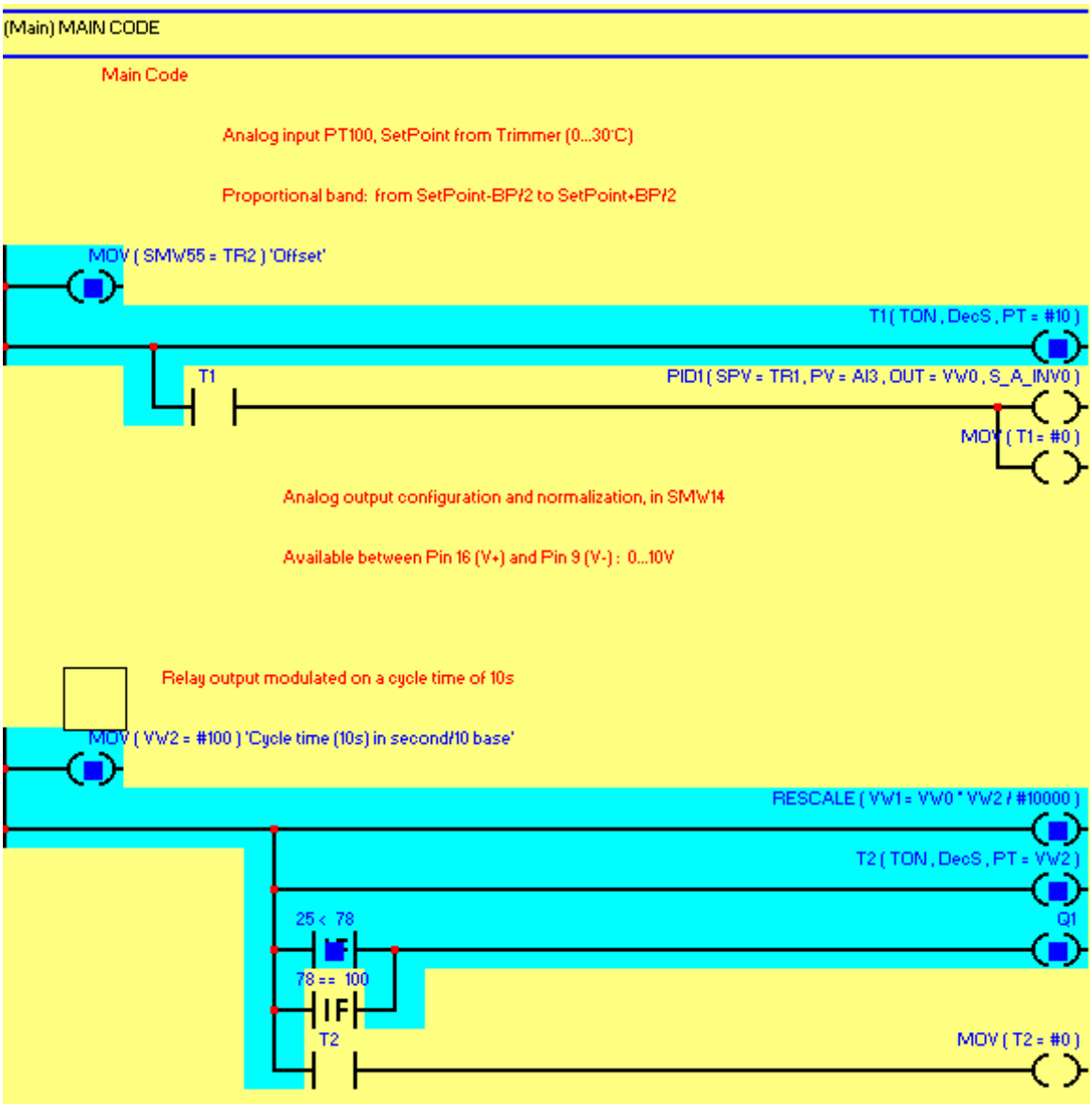


Figure 7.6: operating in run-time. The output calculated by the PID is 7800, so in a cycle time of 10 seconds, Q1 is closed for 7.8 seconds and open for 2.2. This image shows that T2 equals 2.5s so Q1 is closed.

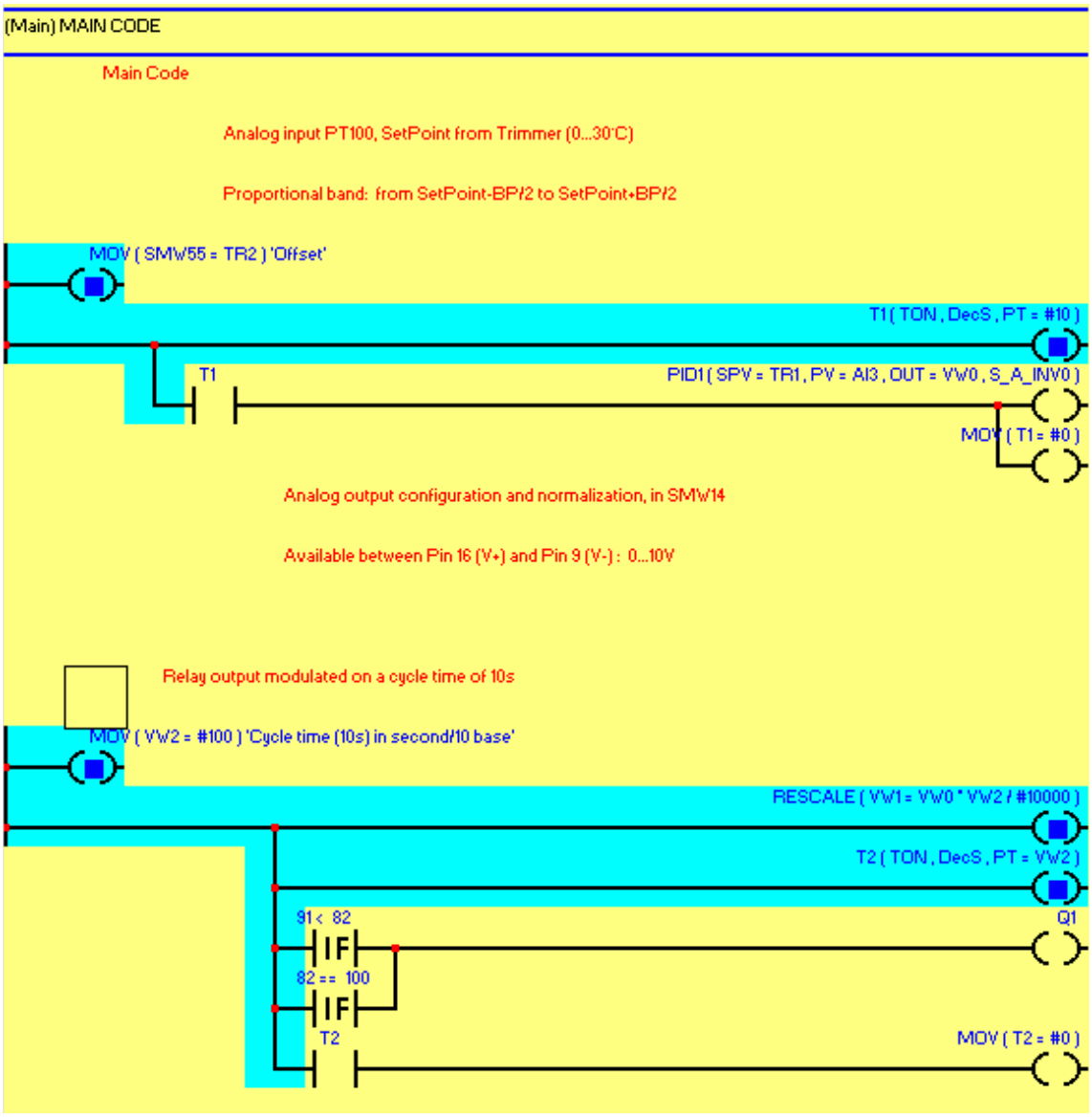


Figure 7.7: Operating in run-time. In this case T2 equals 9.1s, so Q1 is open.

8 TdDesigner

8.1 Introduction

TdDesigner is the development environment to configure graphics in the Pixsys TD320 terminal. This device acts as both PLC and HMI (Human Machine Interface) so generally the development of an application requires the simultaneous use of PLProg and TdDesigner.

The WYSIWYG (What You See Is What You Get) approach of TdDesigner makes it also particularly easy to create and manage complex projects: once transmitted to the terminal the graphical objects and displays appear exactly the same on the PC screen.

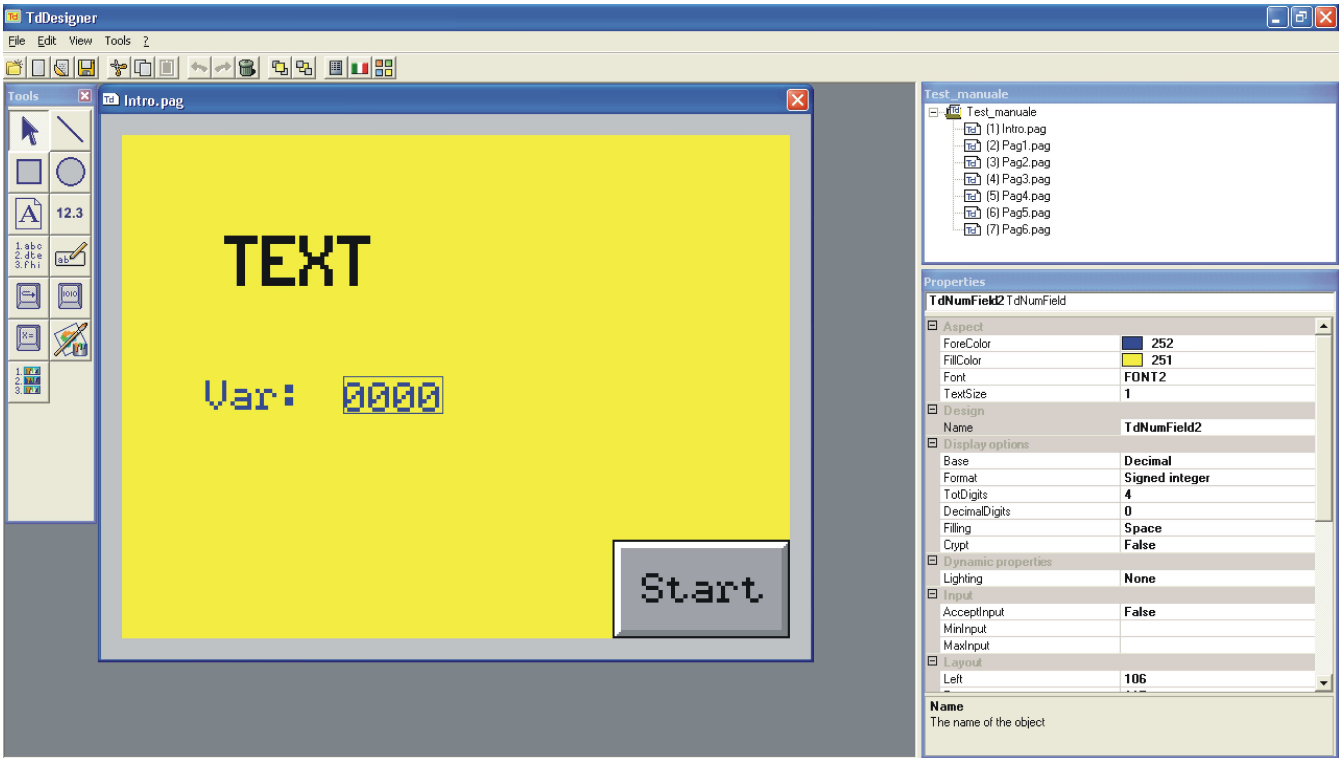


Figure 8.1: A typical TdDesigner screenshot. The Toolbar (left) is seen as well as a page (centre), the Project Management window (upper right) and the Properties window (lower right).

8.2 Definitions

The person using the development environment shall be referred to as “user”. “Operator” shall be the person in front of the graphical terminal configured with TdDesigner.

Items marked (*) are not yet fully supported at the moment this manual is published.

8.2.1.1 Graphical object

Graphical unit contained in a terminal display that can be configured and when necessary can react to operator actions. Typical graphical objects are buttons, geometric figures and variable fields.

8.2.1.2 Variable

Terminal memory area that can be accessed during the design phase using a name assigned by the user.

8.2.1.3 Page

The entire content of a graphical terminal display generally includes an image or a background colour and a series of graphical objects.

8.2.1.4 Project

A group of pages and variables that describe terminal functions. Every project is saved in a group of files, one with the “pag” extension for each page and one with the “tdproj” extension that counts references to the pages and variables.

8.2.1.5 Compile

An automatic operation that allows TdDesigner to translate a project into a configuration file to send to the terminal through PlProg.

8.3 Menu

The TdDesigner menu is similar to most software packages on the market.

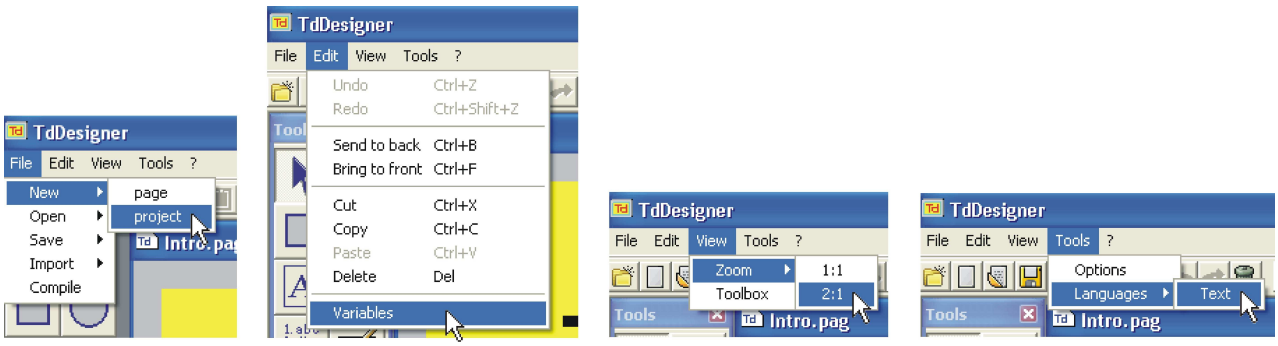


Figure 8.2: TdDesigner menu items.

Some features will be dealt with in later paragraphs so for now just take note that these features are available:

- create, open and save file.
- change open file: “Undo” and “Restore” change, “Cut-Copy-Paste” single objects, groups of objects and entire pages, each with their respective keyboard shortcuts.
- possibility of importing pages from other TdDesigner projects.
- setting of environment “Tools”: table sizing and display, as well as language management.
- project compilation.

The most important menu items also available on the toolbar:

 **New project:** Create a new TdDesigner project.

 **New page:** Create a new page in the project.

 **Open project:** Open an existing project.

 **Save:** Save an open project with the same name.

 **Cut:** Cut the selected object/s. Only activates when at least one object is selected (Ctrl+X).

 **Copy:** Copy the selected object/s. Only activates when at least one object is selected (Ctrl+C).

 **Paste:** Paste the selected object/s. Only activates when at least one object is copied (Ctrl+V).

 **Undo:** Undo the last operation made (Ctrl+Z).

 **Redo:** Redo the last operation undone (Ctrl+Shift+Z).

 **Delete:** Delete the selected object/s. Only activates when at least one object is selected.

 **Bring to front:** Change the position of the overlapped objects. Move forward the selected object (Ctrl+F).

 **Send to back:** Change the position of the overlapped objects. Move backwards the selected object (Ctrl+B).

 **Modify variables:** Open the settings editor and modify variables.

 **Traductions:** Open the **TdLinguist** editor to select and set languages.

 **Compile:** Compile the open project (see paragraph 8.7)

8.4 Properties window

All the properties of the pages and the graphical objects are visible and modifiable in the Properties window (Figure 8.1 and Figure 8.3). In particular, the last object selected (with a click) by the user will show its properties in that window. Just under the properties table a brief message appears with a summarised explanation of the item selected.

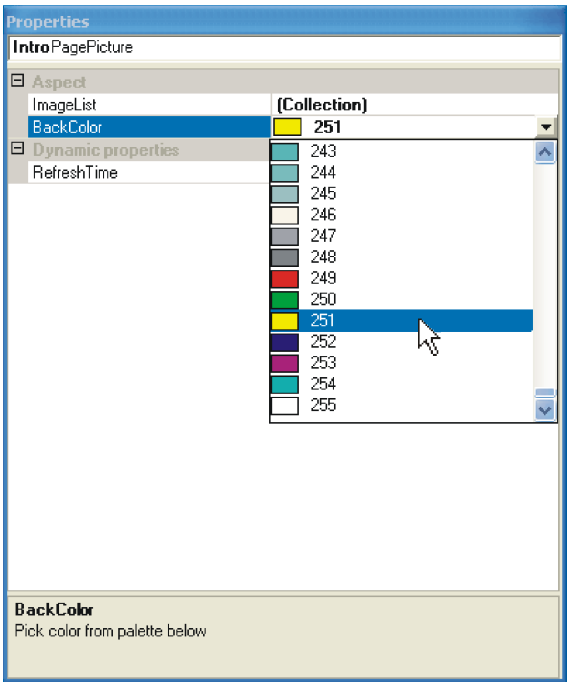


Figure 8.3: Properties window. The object and the property selected are a page and its background colour, respectively.

In Figure 8.3 the name of the **page** object is **Intro**. PagePicture indicates the **page** type.

TdDesigner assigns a progressive default label to the objects. In this example it is **TdText0**, while TdText indicates the data type.

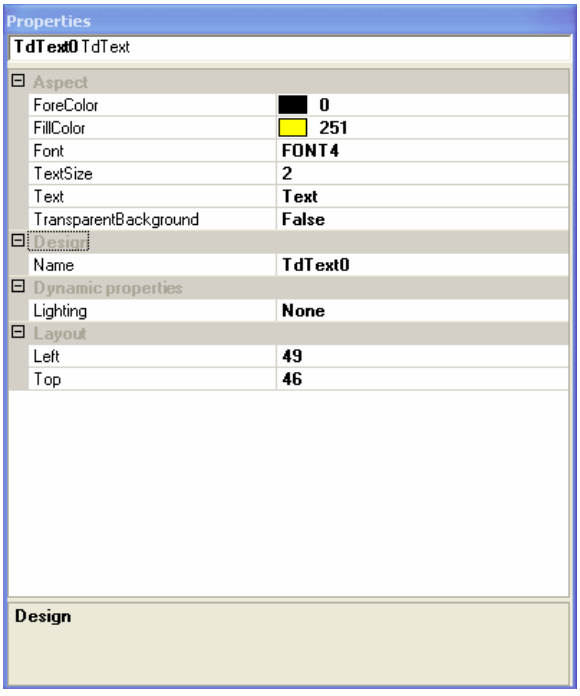


Figure 8.4: Properties window. The object selected is the "Text" of the Intro page.

In the **Design** field the user can assign a name that is different from the default name, as reported in Figure 8.5. Now the “Text” field is called **Free Text**.

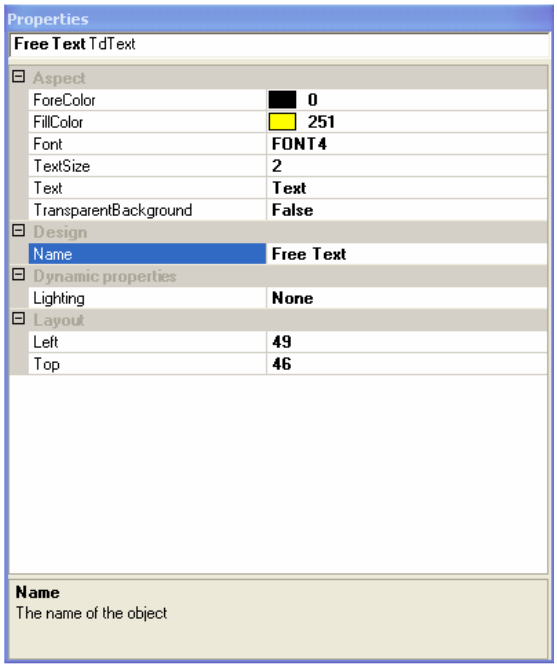


Figure 8.5: Properties window. Object name change.

8.5 Create new project

Select the menu item “**File\New\Project**”, as shown in Figure 8.2 to the left. Select terminal to program.

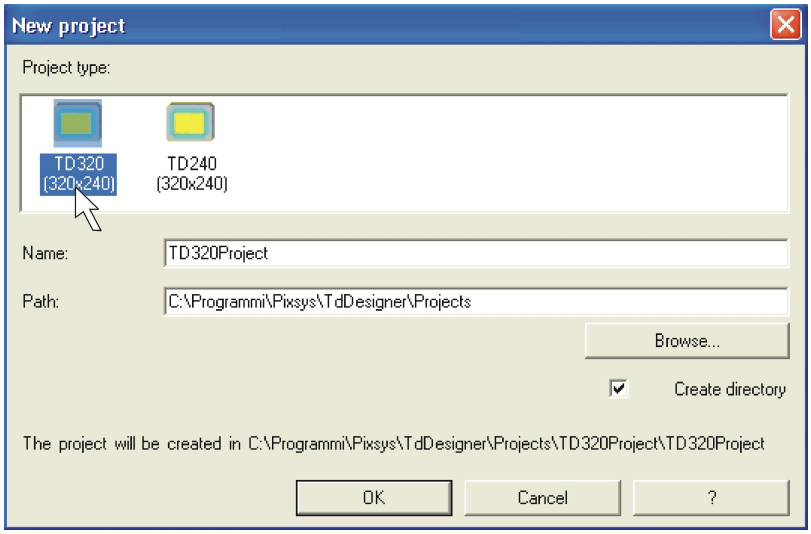


Figura 8.6: Terminal selection

The development environment can put new project in a directory automatically created or in a folder chosen by user.

8.5.1 Language management

The number of languages associated with the application can be modified at any time. By clicking once on the left button of the mouse in the Management Window, above the project name, the Properties window will appear as in Figure 8.7

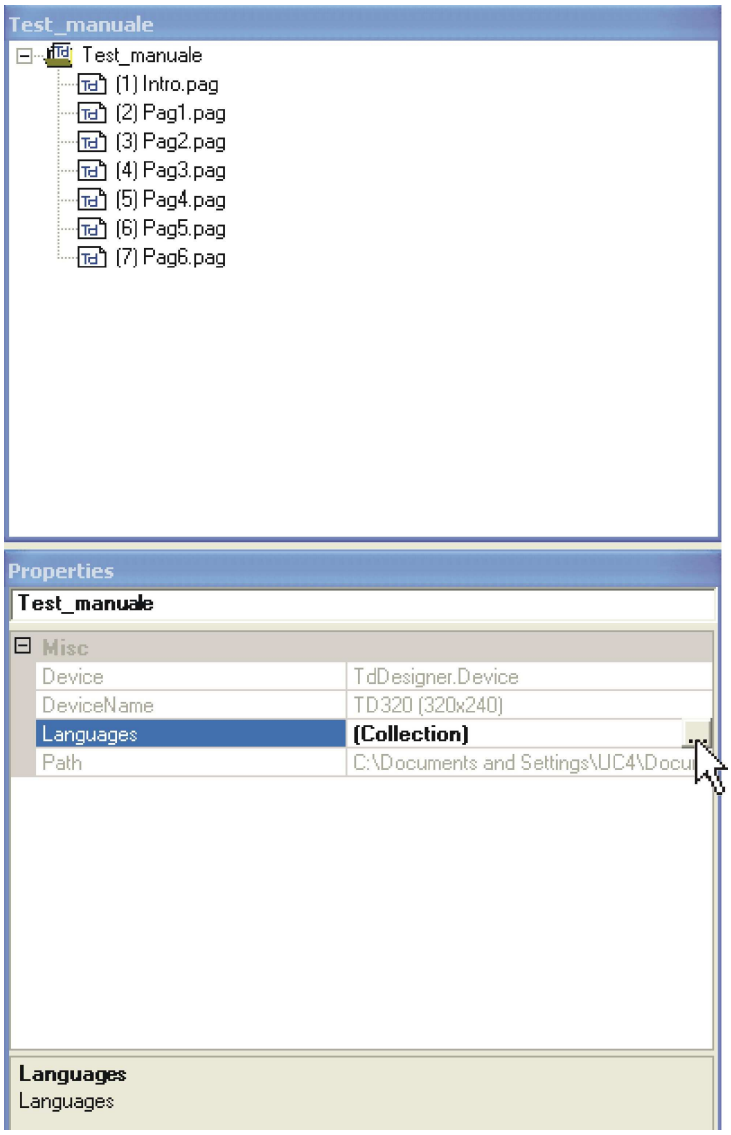


Figure 8.7: Change number of languages.

Still referring to the figure, by clicking on **Collection,** you access the languages editor. The default language is the first on the list.

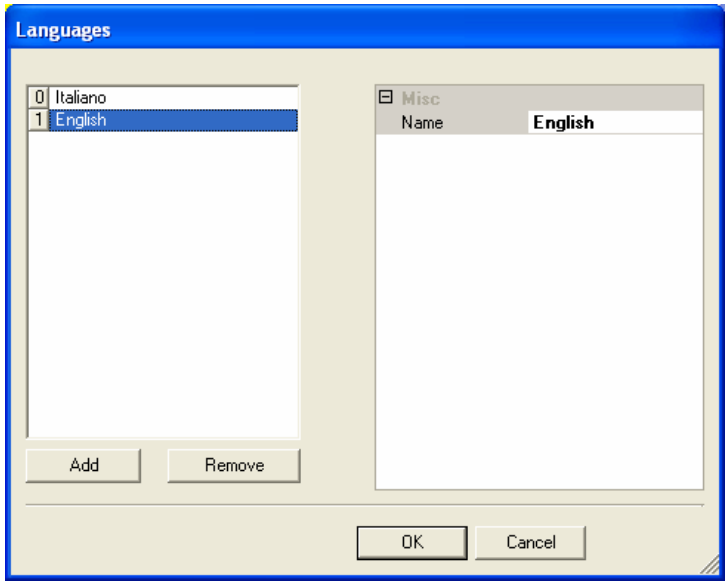


Figure 8.8: Languages editor. To add a new language, press the Add button. The language should be typed to the right of Name. To remove a language press Remove.

To add the translations to associate with the objects containing text, click in the area **Tools / Languages / Text**, or click on the icon in Figure 8.9 to access the **TdLinguist** window.

Figure 8.9: "Traductions" icon.

Select the language in which you want the translations, as reported in Figure 8.10.

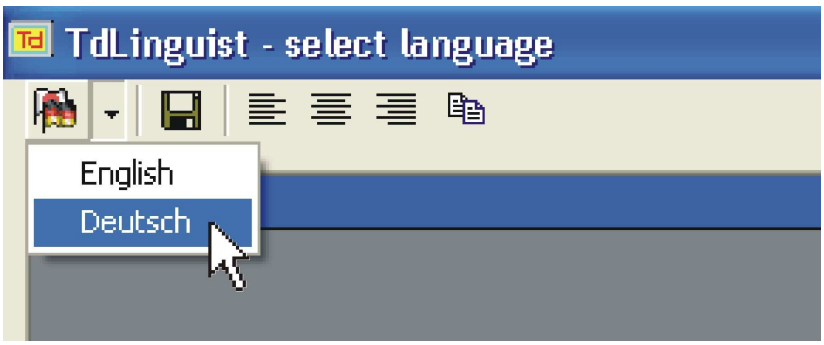


Figure 8.10: TdLinguist window for translations.

By clicking on one of the languages available (there must be at least 2 languages in the **Languages** window of Figure 8.8), the following message appears:

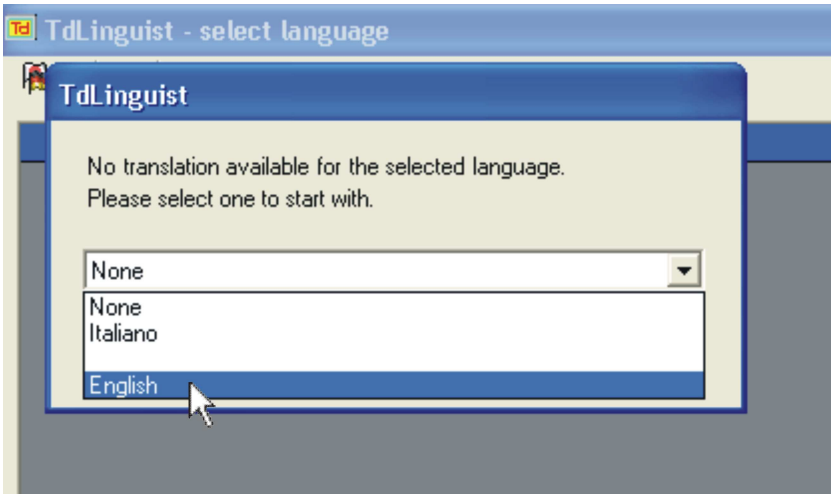
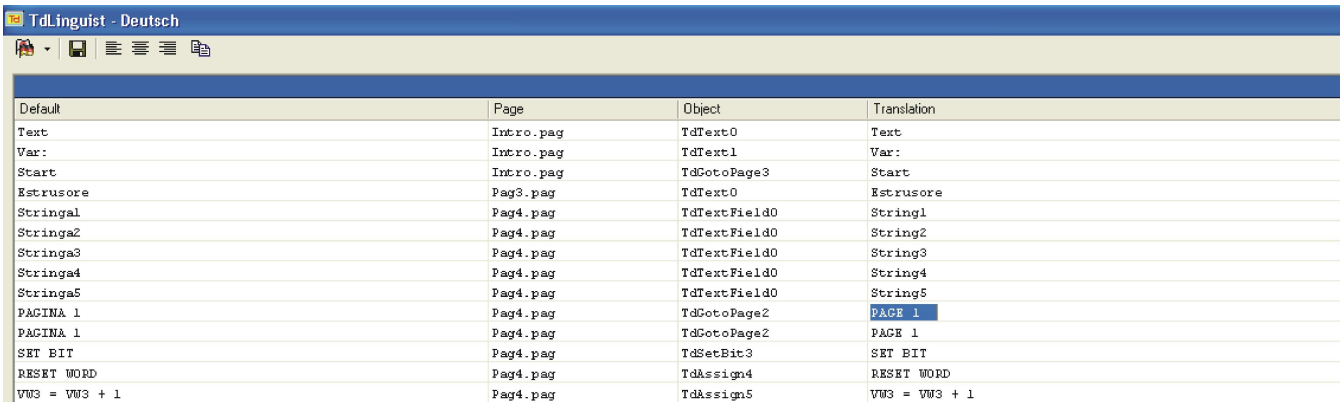


Figure 8.11: TdLinguist window, warning message.

Based on the option selected, a table will be created in which each text will be assigned:

- an empty string (“”) if **None** is selected
- the text associated with the language selected (**English** in the example reported)



Default	Page	Object	Translation
Text	Intro.pag	TdText0	Text
Var:	Intro.pag	TdText1	Var:
Start	Intro.pag	TdGotoPage3	Start
Estrusore	Pag3.pag	TdText0	Estrusore
Stringa1	Pag4.pag	TdTextField0	String1
Stringa2	Pag4.pag	TdTextField0	String2
Stringa3	Pag4.pag	TdTextField0	String3
Stringa4	Pag4.pag	TdTextField0	String4
Stringa5	Pag4.pag	TdTextField0	String5
PAGINA 1	Pag4.pag	TdGotoPage2	PAGE 1
PAGINA 1	Pag4.pag	TdGotoPage2	PAGE 1
SET BIT	Pag4.pag	TdSetBit3	SET BIT
RESET WORD	Pag4.pag	TdAssign4	RESET WORD
VW3 = VW3 + 1	Pag4.pag	TdAssign5	VW3 = VW3 + 1

Figure 8.12: Table of TdLinguist translations – English

The **Default** column reports all the strings for all the objects contained in the project, as they were inserted at the time of definition (the example shows Italian).
The **Page** column reports the page in which the proprietary object of each string is located.
The **Object** column reports the name of the objects to which the strings belong.
The **Translation** column will report the translations of the strings.

The user can customise translations, using possible text alignment tools in the toolbar.
Industrial applications often contains objects with equal strings, and traductions writing can be very boring.  button allows to associate the same traductions to equal strings automatically.

The memory area that indicates the language to use is **SMW13**
SMW8 = 0 Language1 Italian
SMW8 = 1 Language2 English
SMW8 = 2 Language3 Deutsch
... ...

Following the example, when **SMW13** becomes value **2**, all the objects with text will display their strings by retrieving the content from the **Translation** column of Figure 8.12.
Follow the same procedure explained in this paragraph to insert another language. You must choose another source language and associate the new translations for the object strings. The new language will be associated with value **3** of **SMW13**.

8.5.2 Variables management

The set of variables can also be modified at any time using the editor, represented in Figure 8.14, from the "**Edit / Variables**" menu item (Figure 8.2 in centre) or the button in Figure 8.13.

Figure 8.13: "Modify variables" icon

The **Filter** field selects whether to display all the project variables (first option *****) or only the variables contained in a particular page. In the field **Add variable** select the type of variable memory area you want to insert (# indicates numeric constants). The button → inserts the only variable selected, **[...]**→ inserts a maximum group of 100 adjacent variables.

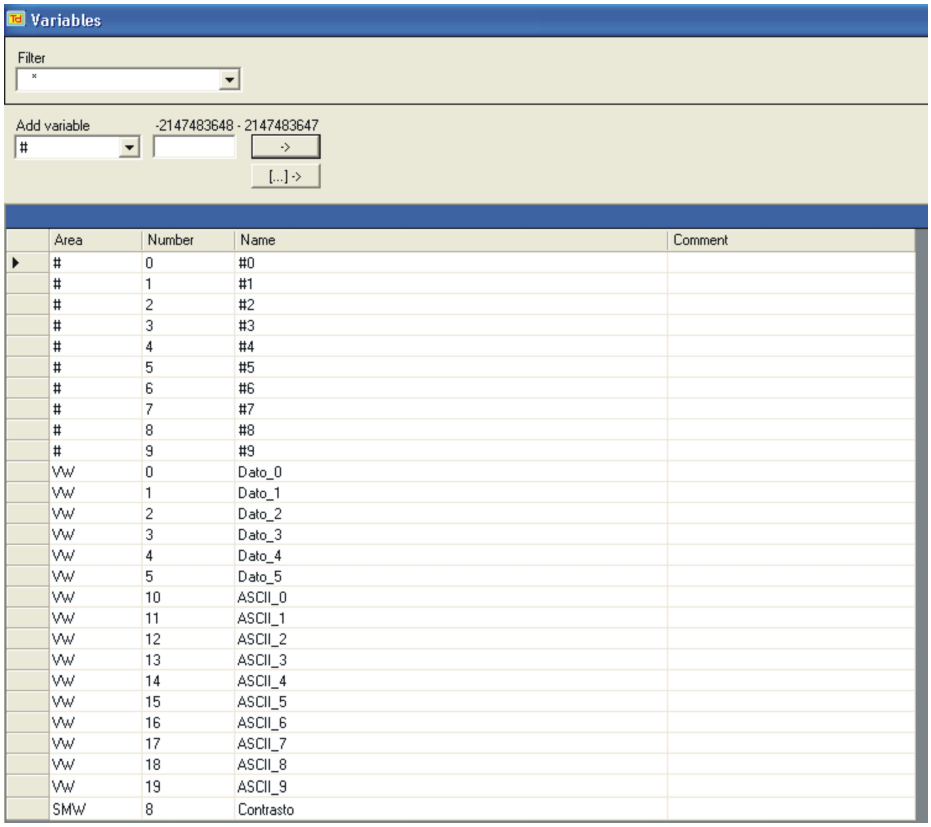


Figure 8.14: Editor of Variables set. The list can be displayed in alphabetical order based on the name assigned by the user by clicking on the "Name" box or in progressive order per memory area (default) by clicking on the "Area" or "Number" boxes.

The variables must always be declared in the table before being used in the graphical objects.

8.5.3 Page management

The set of project pages with the relevant progressive number is outlined in the Project Management Window (Figure 8.1). To open a previously created page, just double click on the name. To add a new page, select the "**File \ New \ Page**" menu item or the corresponding button (see par. 8.3).

The order of the pages is important because when you start-up the terminal the first page is always displayed. Each page can at any time be set as the first using a specific menu, as shown in Figure 8.15

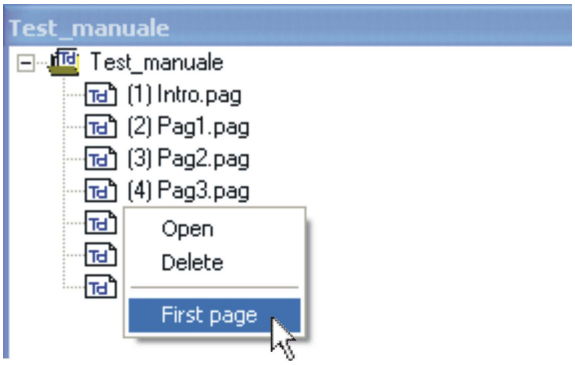


Figure 8.15: Context menu, opens by clicking on the right hand button on a page name.

Once a page is created, you must set its properties: the Refresh time that can vary from 100ms to 5s and the background that can be of uniform colour (see Figure 8.3), or a bitmap[3] image in dimensions equalling the resolution of the display (320x240). In particular, for the latter operation refer to par. 8.6.7 on bitmap objects.

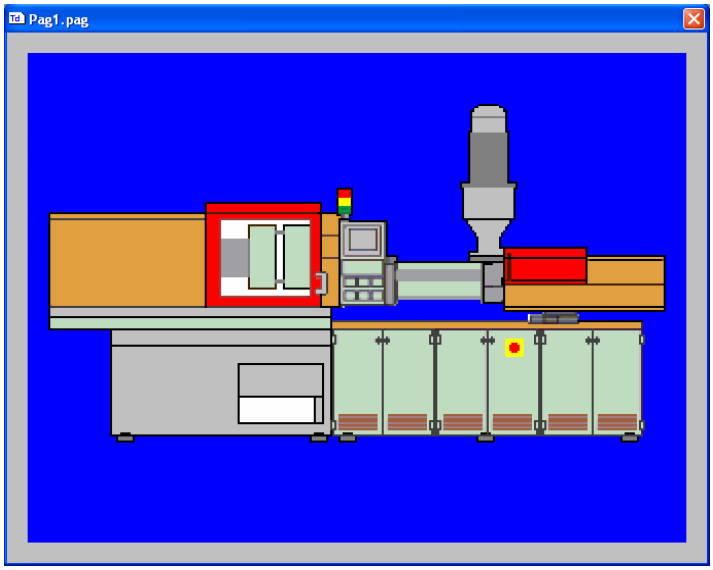


Figure 8.16: A page with a background loaded with a bitmap image.

[3] The TD320 terminal can display images with 256 colours with a set palette.

8.6 Graphical objects

The configuration of a page is subject to the setting of its properties and the insertion of graphical objects. To introduce a new object just select it from the Toolbar and position it in the window that represents the page. The terminal is sensitive to the order of insertion: if two objects are overlapped, only the last will be entirely displayed. For this reason, the "**Edit**" menu (Figure 8.2 in centre) contains the items "**Send to back**" (also using the keys **Ctrl+B** on the keyboard) and "**Bring to front**" (also using **Ctrl+F**), that allow you to edit the position of the objects. The same function is available with the buttons on the toolbar:

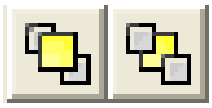


Figure 8.17: "Bring to front" to the left, "Send to back" to the right

Figure 8.18: Toolbar. Apart from the pointer, it contains all the objects that can be used on a page.

8.6.1 Geometric shapes

8.6.1.1 Line object

Once selected from the Toolbar, to draw a line on a page, using the mouse keep the left button clicked and drag the pointer to the desired length.

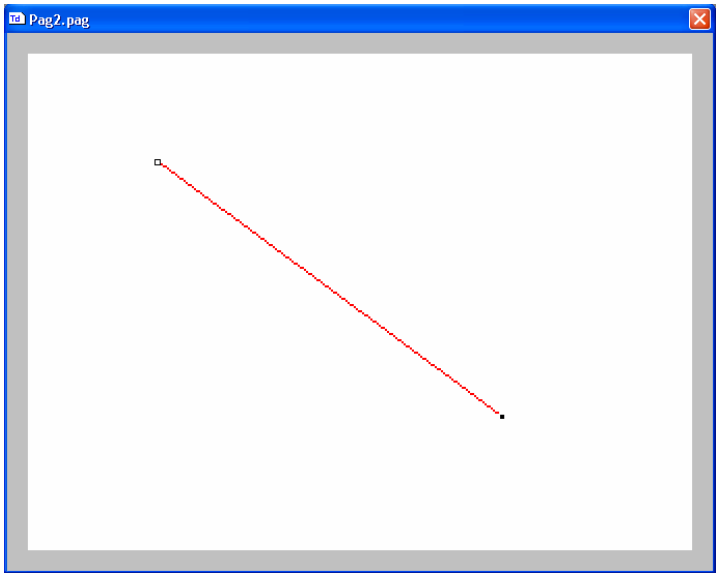


Figure 8.19: Line object on page Pag2.

The object can be moved or resized, even after insertion using the mouse or from the Properties window, which now looks like that of Figure 8.20.
As well as the **ForeColor**, the source coordinates (**X1**, **Y1**), the target coordinates (**X2**, **Y2**) and the object name (**TdLine1**, assigned by default) can be changed.

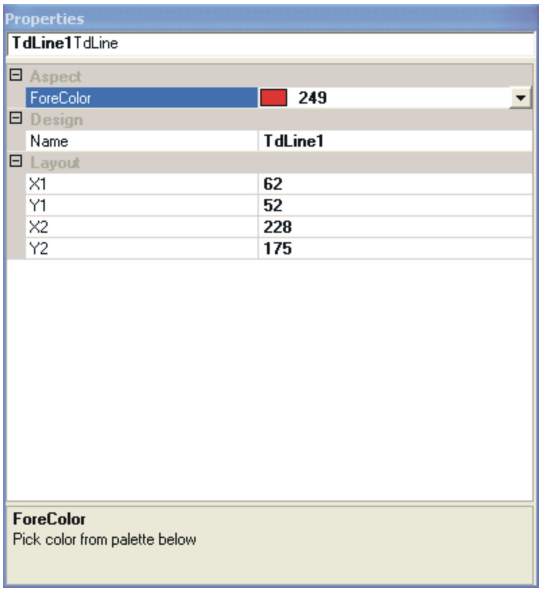
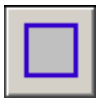


Figure 8.20: Properties table of Line object on page Pag2.

8.6.1.2 Rectangle object



Once selected from the Toolbar, to draw a rectangle on a page keep the left button of the mouse clicked and drag the pointer to define its dimensions.

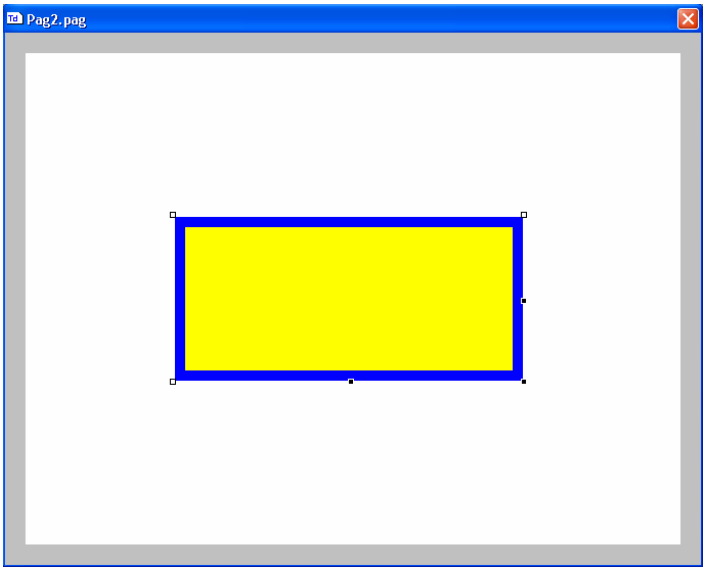


Figure 8.21: Rectangle object on page Pag2.

The object can also be moved and re-sized after insertion both using the mouse or the Properties window, which now resembles that of Figure 8.22.
As well as the **Forecolor** (in blue in the example, the perimeter of the rectangle), the following can be changed: the **FillColor** (in yellow in the example, inside the rectangle), the **Filled** field (**True** in the example, with a full rectangle, to have only the perimeter with an empty inner select **False**), the coordinates of the left upper top (**Left** X axis, **Top** Y axis), the dimensions (**Width** and **Height**) and the name of the object (**TdRectangle0**, assigned by default).

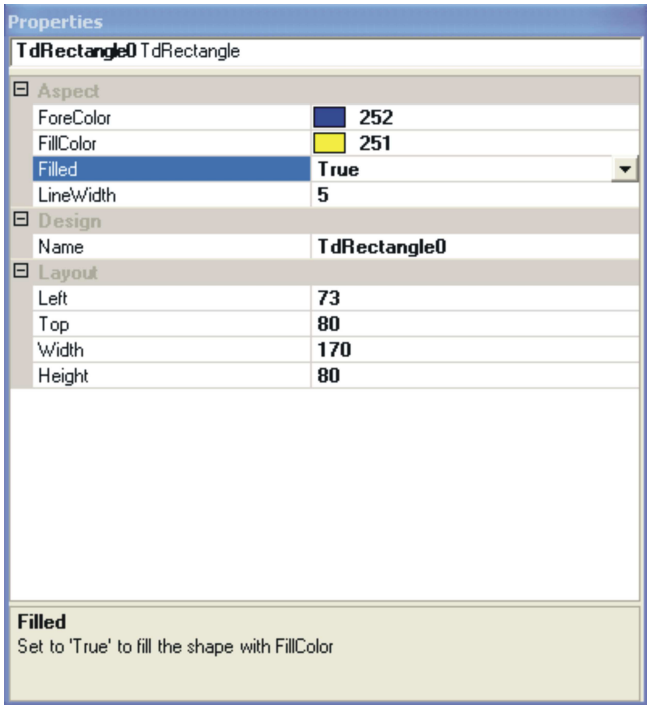
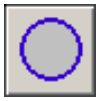


Figure 8.22: Properties window of Rectangle object of page Pag2.

8.6.1.3 Ellipse object



It can be drawn in the same way and has the same properties as the Rectangle object. To get a circumference set the **Width** and **Height** fields the same.

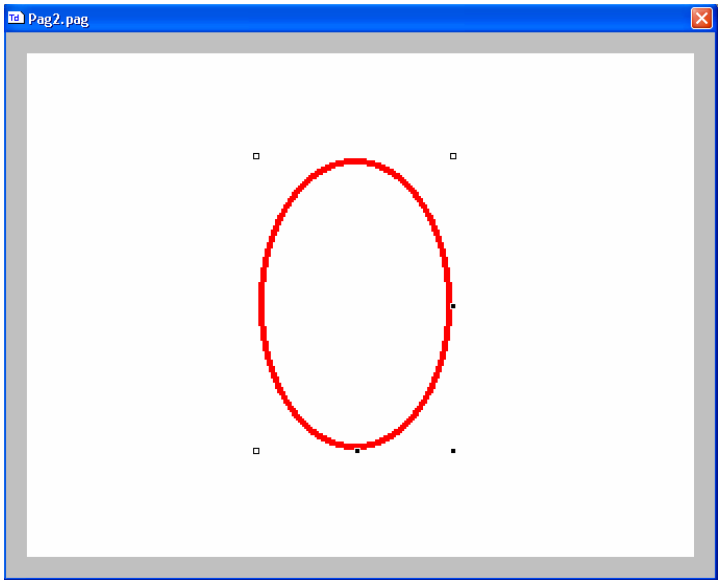


Figure 8.23: Ellipse object on page Pag2.

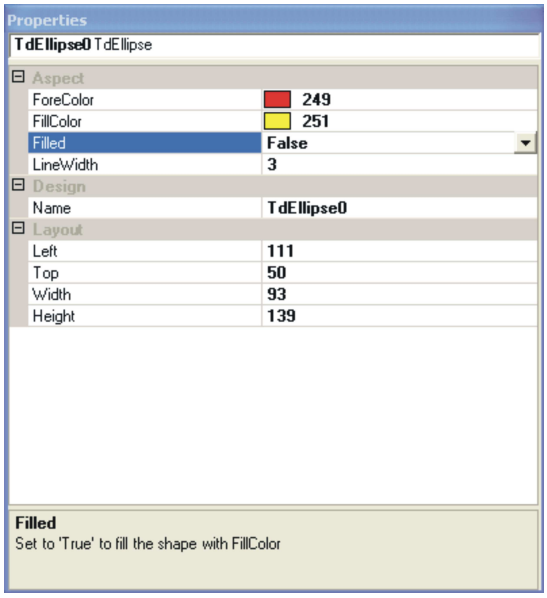


Figure 8.24: Properties window of Ellipse object on page Pag2.

8.6.2 Label object

Once the object is selected from the Toolbar, position it on the page with a simple click. The text can be directly added from the keyboard if the label has already been inserted or selected, or using the Properties window in the "**Text**" field. With reference to

the **Text** writing, displayed in Figure 8.1, the corresponding Properties table is:

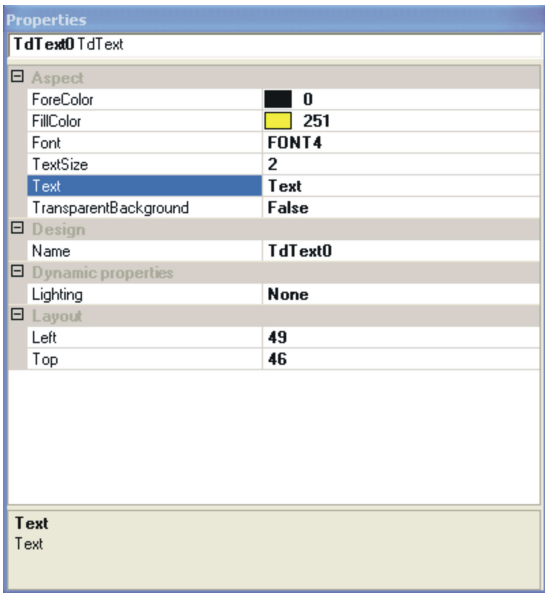


Figure 8.25: Properties window of "Text" object on page Pag1

The following can be changed: the **ForeColor** (colour of text on page), the **FillColor** (rectangular background colour of text), the **Font** (5 types of font are available), the **TextSize** (multiplier by 1, 2 or 4 of the original font sizes), the upper left angle (**Left** X axis, **Top** Y axis) and the name of the object (**TdText0**, assigned by default).
The **TransparentBackground** field was created to overlap an image with text (think of a synoptic), displaying only the writing without the rectangular border, as in Figure 8.26.

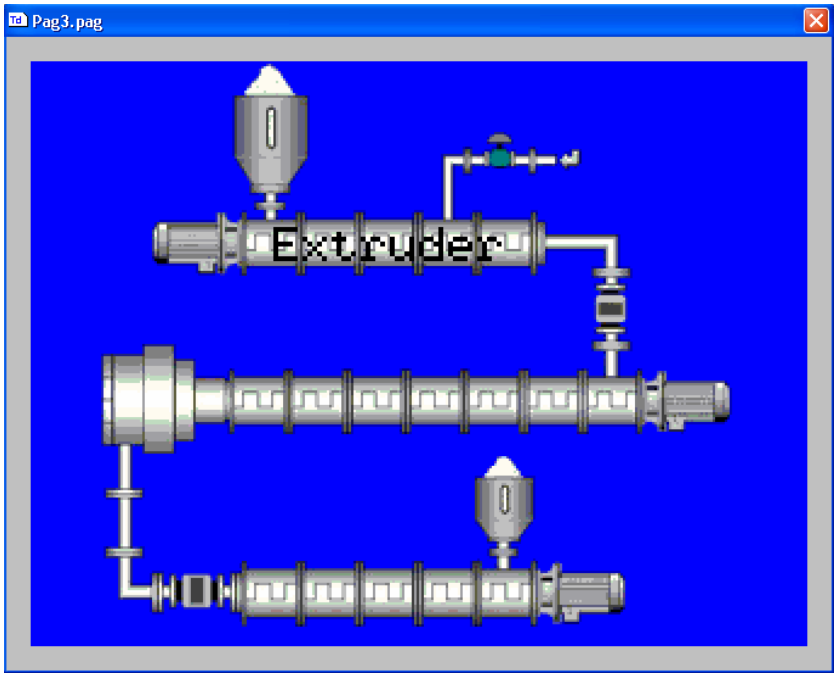


Figure 8.26: The "Extruder" label overlaps the background image of the page with TransparentBackground True.

The Properties table relating to the object described is as follows:

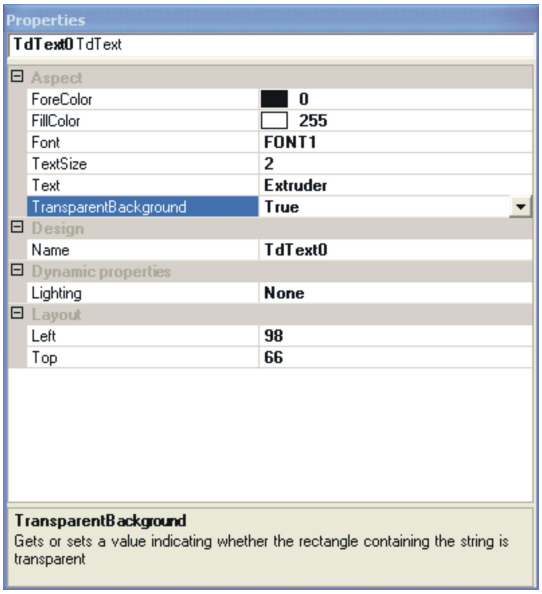


Figure 8.27: Properties table of "Extruder" label on page Pag3.

Lighting: The text of the Label object can blink (set 1 second period). There are 4 types of blinking:

1. **None**: no blinking
2. **Text**: text only blinking and the rectangular background is fixed
3. **Back**: the text is fixed and the rectangular background flashes

4. **All**: both blinking

8.6.3 Numeric Field object

This allows you to display and modify the value of a variable. Once the object is selected from the Toolbar, position it on the page with a simple click. To connect a numeric field with a variable you must check the variable is already present on the list associated with the project (see par. 8.5.2, in this example it is the memory area **VW0**, called **Dato_0**). With reference to numeric field **0000**, displayed in Figure 8.1, the corresponding properties table is:

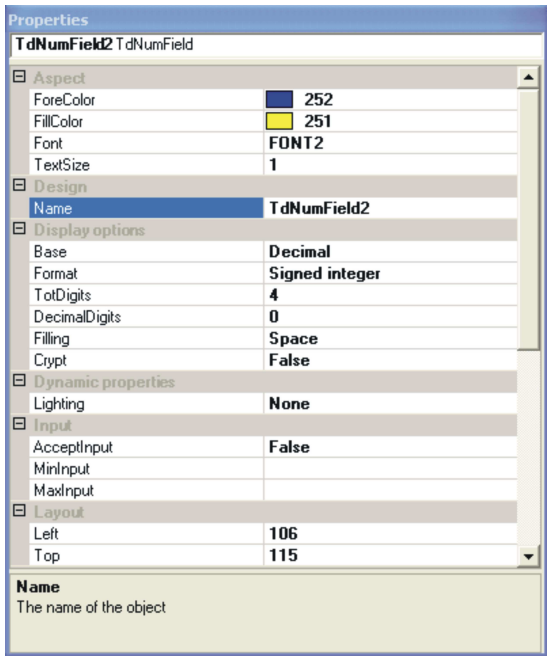


Figure 8.28: First part of Properties table of Numeric Field object on Intro page.

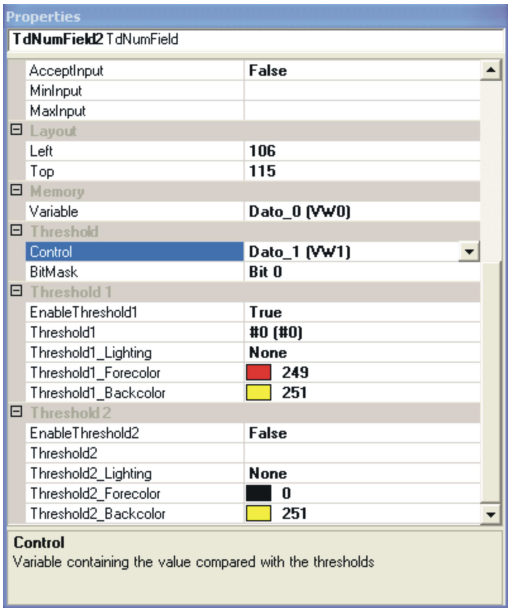


Figure 8.29: Second part of Properties table of Numeric Field object on Intro page. Threshold1 enabled, Threshold2 disabled.

The fields **ForeColor**, **FillColor**, **Font**, **TextSize**, **Left**, **Top, Name** and **Lighting** are similar to those of the Label object (see par. 8.6.2).

Memory: the **Variable** field must have a connected previously set memory area (see par. 8.5.2).

Input: this field defines whether the variable must be editable or not. If the **AcceptInput** field is **False**, as in the example, the variable cannot be edited by the operator. If it is **True**, the operator can modify the value of the variable within the **MinInput** - **MaxInput** (constants or variables) restrictions, using a specific page loaded by pressing on the object.
Display Options: the **Format** field identifies the type of data and therefore its display on the terminal. **Signed Integer** and **Unsigned Integer** must be used for variable words (16bit), **Signed Long** for double words (32bit).
The **Base** field indicates the basis mathematics of the data (**Decimal**, **BCD**, **Exadecimal** and **Binary**).
The number of **TotDigits** must also take into account possible decimal points and sign. For example, with 4 **TotDigits** and 1 **DecimalDigits**, if the value of the variable VW0 is 8375, the

terminal would display 37.5, not displaying the most significant digit (the correct display of the number of **TotDigits** should be 5).
The **Filling** field allows you to fill the most significant empty digit with a **Space** or **Zero**.
The **Crypt** field is generally **False** for variables that must be displayed and modified and **True** for variables whose value must not be known (useful when entering a password, for example). In this later case, the digits displayed will be replaced with ***

Threshold, Threshold1, Threshold2: the **Control** field should be associated with a variable responsible for the possible variation of certain display properties.
Based on the **BitMask** value, the **Control** variable will be entirely evaluated (**None**) or one of its bit (**bit0**, **bit1**, ... , **bit31**).
For each of the two Thresholds (Threshold1 and Threshold2), the control can be enabled (**EnableThreshold True**) or disabled (**EnableThreshold False**).
The **Threshold** field is the comparison value, intended as ≥ (greater or equal).

- In the case of no mask (**BitMask = None**), the comparison has a positive outcome if **Control ≥ Threshold**
- In the case of a bit mask (**BitMask = bit N**), the comparison has a positive outcome if **bit N** of **Control = 1** (**Threshold** has no significance)

In the case of a positive comparison outcome, the object will change its display characteristics in relation to blinking (**Threshold_Lighting**), the colour of the writing (**Threshold_Forecolor**) and the background colour (**Threshold_Backcolor**).

Based on the Properties tables in Figure 8.28 and Figure 8.29, as long as **bit 0** of the **VW1** variable is equal to **0**, the **VW0** variable will be displayed in blue on a yellow background. When bit **0** of the **VW1** variable equals **1**, the **VW0** variable will be displayed in red with a yellow background.

8.6.4 Text Field object

Starting with a list of strings, display a text indexed by the value of a variable. The string in the development environment is always that associated with the first on the list, while the string displayed by the terminal is that of a corresponding value:

Variable = 0 → String1
Variable = 1 → String2
Variable = 2 → String3
Variable = 3 → String4
Variable = 4 → String5
...

Figure 8.30 contains a Text Field object displayed as **String1**.

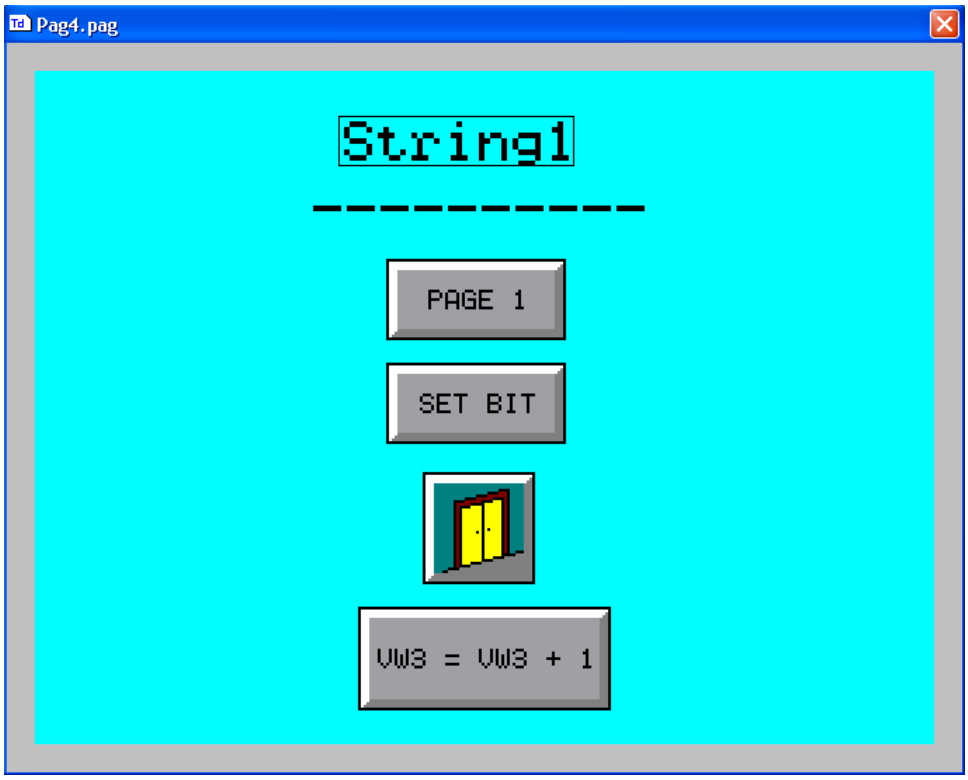


Figure 8.30: Text Field object of "String1" on page Pag4

The list of strings can be accessed from the **Aspect / StringList** field and looks like the list in Figure 8.31

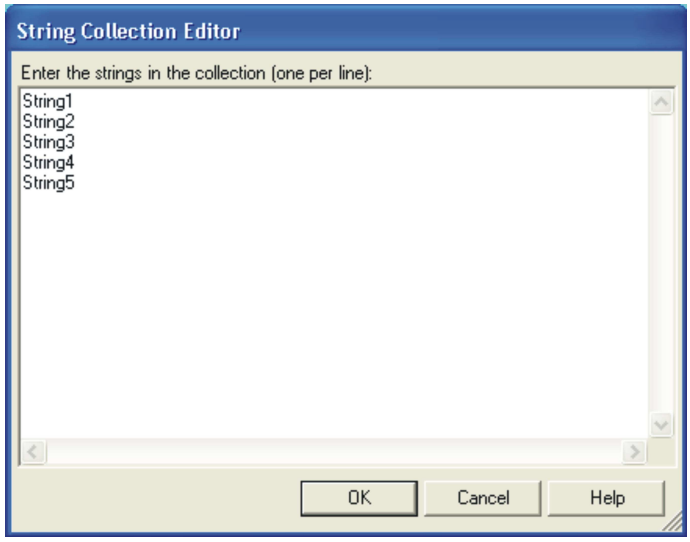


Figure 8.31: Setting of StringList for Text Field of “String1”

The length of all the strings on the list is automatically reported to that of the first string, therefore when strings have a different length the longest must be calculated and if necessary fill the first with spaces.

With reference to the Text Field of **String1**, the corresponding Properties table is as follows:

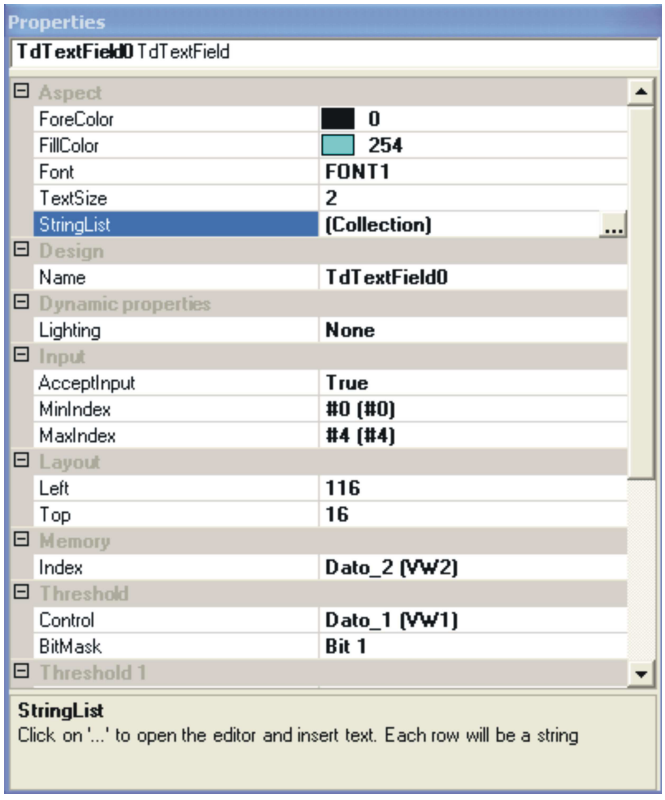


Figure 8.32: First part of Properties table of the Text Field object for "String1" on page Pag4.

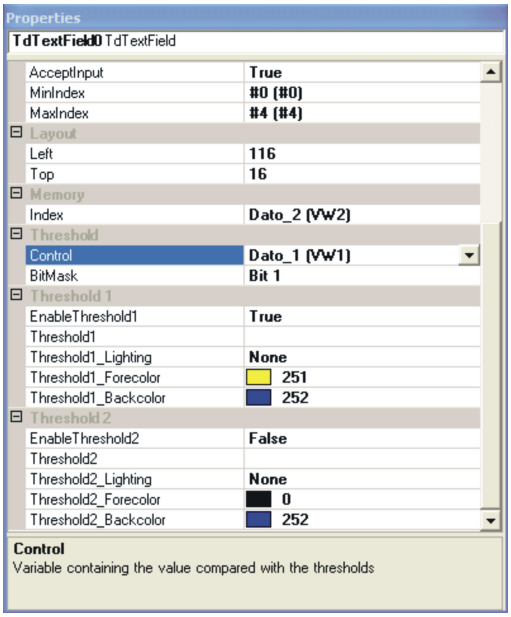


Figure 8.33: Second part of Properties table of the Text Field object of "String1" on page Pag4.

The fields **ForeColor**, **FillColor**, **Font**, **TextSize**, **Left**, **Top, Name**, **Lighting, Control, BitMask, EnableThreshold, Threshold, Threshold_Lighting, Threshold_Forecolor, Threshold_Backcolor** are similar to those of the Numeric Field and Label (see par. 8.6.2 and 8.6.3).

Input: the **AcceptInput** field defines whether the object can be modified by the operator (**True**) or displayed only (**False**).
The **MinIndex** and **MaxIndex** fields restrict the set of strings that can be selected by the operator.

Memory: the **Index** field contains the previously set variable that indexes the list of strings inserted. To correctly display all the N strings on the list, the **Index** variable must vary from 0 to N-1.
With reference to the Properties table in Figure 8.32 and Figure 8.33, the Text Field object is a string indexed from the **VW2** variable and modifiable by the user who can go through them from first (**VW2=0**) to last (**VW2=4**). The colour of the string displayed will be black on a blue background until bit **1** of the **VW1** variable is equal to **0** and yellow on a blue background when bit **1** of the **VW1** variable is equal to **1**.

8.6.5 ASCII Field object

This object enables the operator to insert free text, formatted by letters, numbers and other characters. The maximum number available is decided by the programmer during the object definition phase.
During the development environment, this is how a 10 character ASCII field looks.

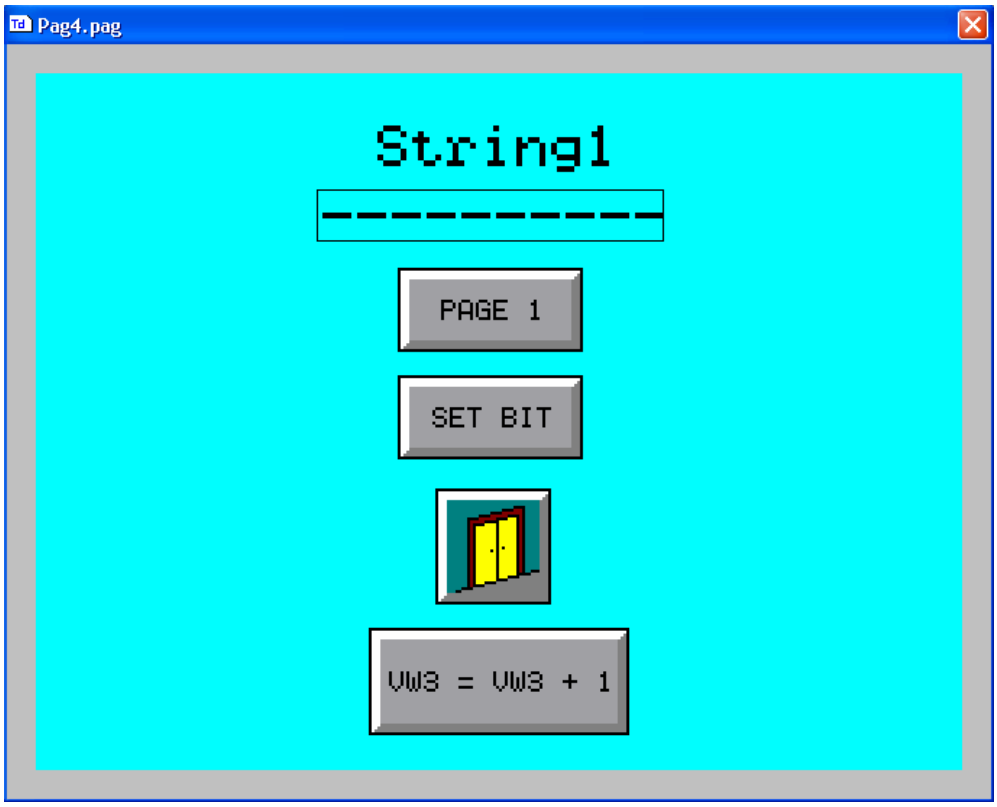


Figure 8.34: ASCII Field on page Pag4 (ten consecutive dashes).

With reference to the ASCII field displayed in Figure 8.34, the corresponding Properties table is as follows:

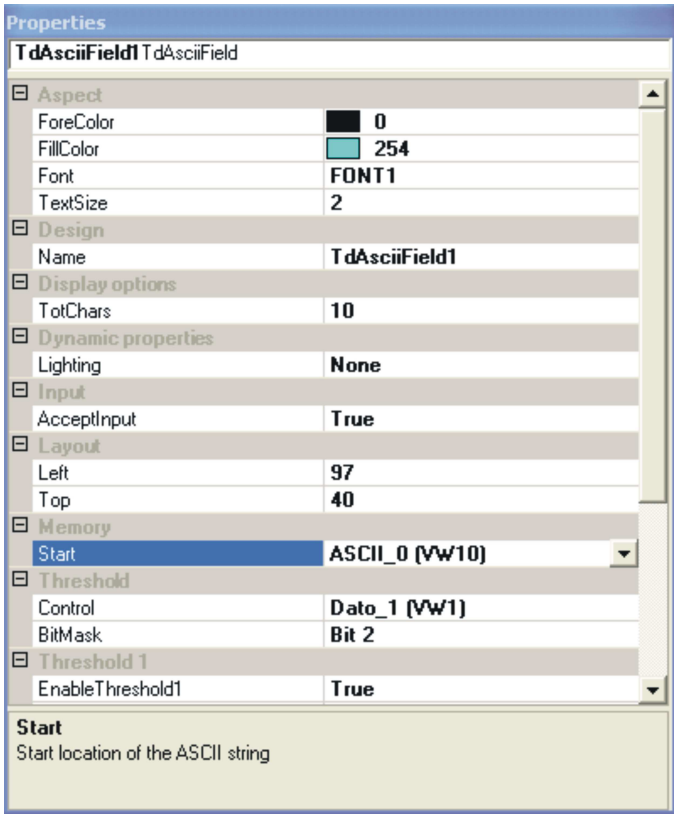


Figure 8.35: First part of Properties table of ASCII Field object.

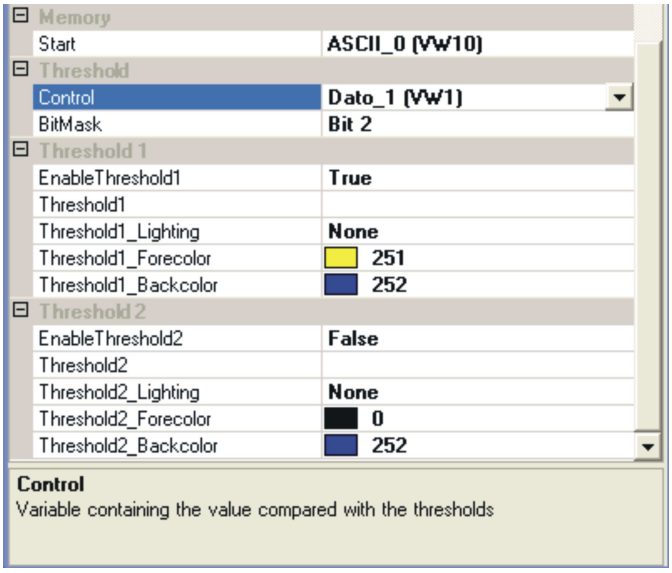


Figure 8.36: Second part of Properties table of ASCII Field object

The fields **ForeColor**, **FillColor**, **Font**, **TextSize**, **Left**, **Top, Name**, **Lighting, Control, BitMask, EnableThreshold, Threshold, Threshold_Lighting, Threshold_Forecolor, Threshold_Backcolor** are similar to those of Numeric Field, Label and Text Field (see par. 8.6.2, 8.6.3 and 8.6.4).

Input: the **AcceptInput** field defines whether the object can be modified by the operator (**True**) or displayed only (**False**).

Memory: the **Start** field sets the memory area that will be used to save the first ASCII character.

Display options: the **TotChars** field determines the total length of the ASCII field and can only be a numeric constant.
If the **Start** field has an associated $\mathbf{VW_x}$ variable and the **TotChars** are 10, the memory areas from $\mathbf{VW_x}$ to $\mathbf{VW_{x+9}}$ will contain the ASCII codes of the characters inserted.

With reference to the Properties table in Figure 8.35 and Figure 8.36, the ASCII field takes up the 10 words from **VW10** to **VW19**. The writing will be black on a blue background until bit **2** of the **VW2** word is **0** and yellow on a blue background when bit **2** of the **VW2** word is **1**.

8.6.6 Button objects

"**Go To page" button**

"**Set bit" button**

"Assign" button

From a display point of view, the buttons are the same. In particular, you can set the graphic style (3D, flat or hidden), the text or the image on the button surface, the line colour and filling.
These objects also allow you to change certain characteristics based on the comparison of a control variable with two possible threshold values: the colours, text, image and enabling of the function associated with it can vary.

8.6.6.1 “Go to page” button

The “Go to page” button is characterised by the **Action/ PageNumber** field which should have an associated constant or variable that identifies the page the terminal must display on pressing the button. Figure 8.37 reports the **PAGE 1** button, while Figure 8.38 and Figure 8.39 report its Properties table.

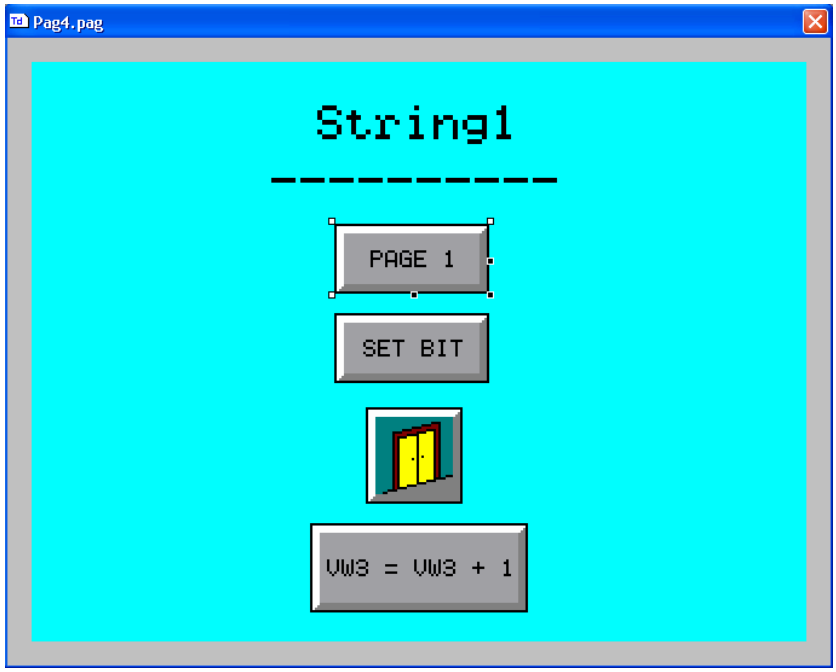


Figure 8.37: “Go to page” button for "PAGE 1" of page Pag4

The fields **ForeColor**, **FillColor**, **Font**, **TextSize**, **Left**, **Top, Name, Control, BitMask, EnableThreshold, Threshold, Threshold_Forecolor, Threshold_Backcolor** are similar to those of Numeric Field, Label, Text Field and ASCII Field (see par. 8.6.2, 8.6.3, 8.6.4 e 8.6.5).

Aspect: the writing you want to display on the button should be written in the **Text** field.
You can associate one or more bitmap images to display in the **ImageList** field (see note 3 on page 57). To import them, simply click with the left button of the mouse on the point indicated in Figure 8.40. The editor in Figure 8.41 appears.

The **ImageList** property has higher priority than the **Text** property.

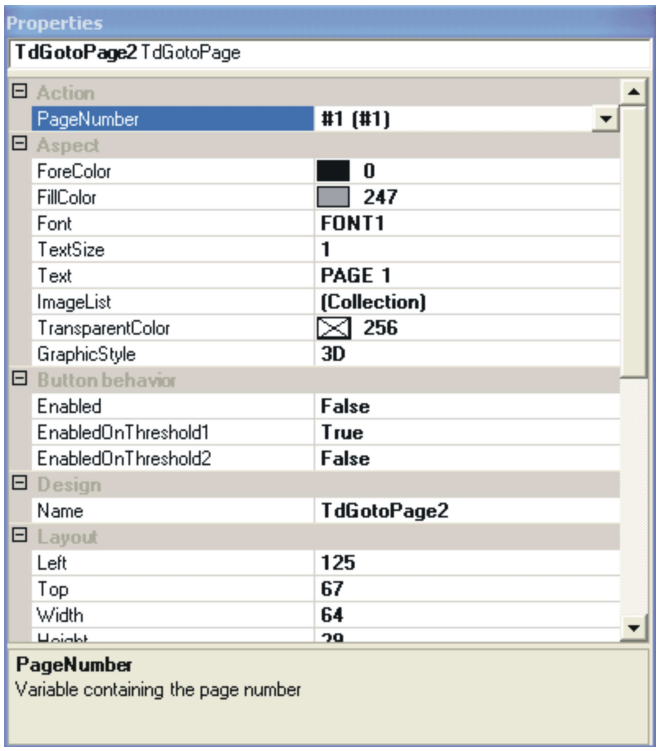


Figure 8.38: First part of Properties table of “Go to page” button object of "PAGE 1" on page Pag4

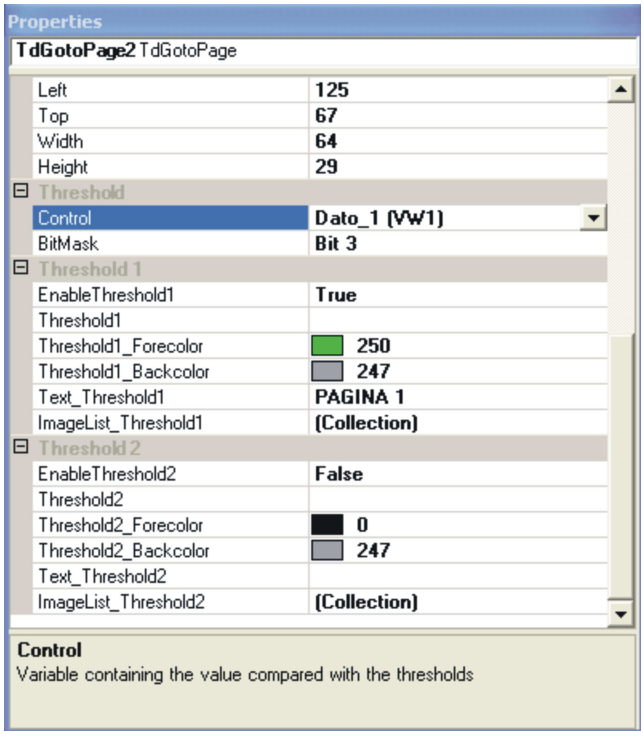


Figure 8.39: Second part of Properties table of “Go to page” button object of "PAGE 1" on page Pag4

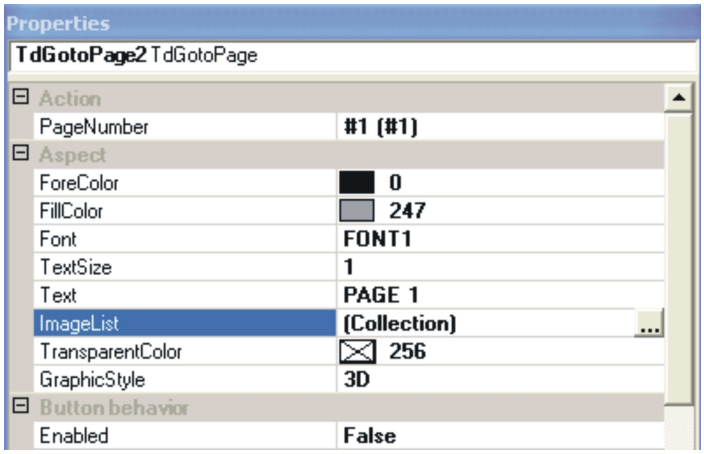


Figure 8.40: Opening of editor for ImageList of "Go to page" button object.

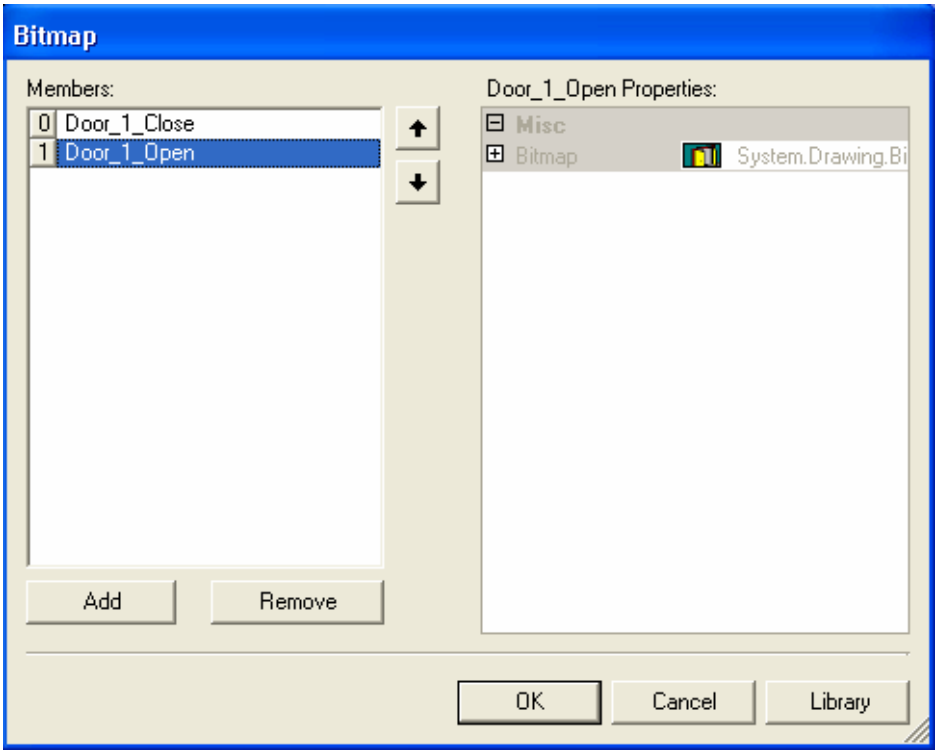


Figure 8.41: Editor. To insert a new image press the "Add" button and select a file in bitmap format. The arrows to the right of the list are used to change the order of the images inserted. The "Library" button has the same function as "Add", but directly opens a folder of default images.

The **TransparentColor** field has the same importance as the Label field. The colour chosen as transparent will no longer be a colour of the bitmap image and will be replaced with the colour underneath it (the colour inside the button). This is useful to delete the rectangular border of the imported bitpmap and for the optimal "fusion" of the image above the button (**256**, default, means no transparent colour).

The **GraphicStyle** field enables the choice of 3 types of display:

1. **Hidden**: the button is not visible on the terminal.
2. **Flat**: a two-dimensional button that is flat and without a visual pressing effect.
3. **3D**: a three-dimensional button, with a visual effect on pressing the terminal.

Action: the **PageNumber** field should be associated with the constant or variable that identifies the page the terminal must display on pressing the button.
Button behaviour: the button can be enabled (**Enabled True** field) or disabled (**Enabled False** field) also with reference to the **Control** variable and the threshold values (**EnabledOnThreshold1** and **EnabledOnThreshold2**).

Layout: the **Width** and **Height** fields represent the button dimensions. In the case of an overlapped image, the dimensions must ensure they can contain the bitmap.

Threshold, **Threshold1**, **Threshold2**: other than the colours of the writing and the background, the button also allows you to change the text (**Text_Threshold**) and the associated bitmap image (**ImageList_Threshold**), on events linked to two thresholds (**Threshold1** and **Threshold2**) (as the Numeric Field object, see par. 8.6.3).

With reference to the Properties table in Figure 8.38 and Figure 8.39, the "**PAGE 1**" object is a Go to page button, which is normally disabled. It is enabled on verification of the event linked to **Threshold1**, i.e. when bit **3** of the **VW1** word is equal to **1** and the writing "**PAGE 1**" appears in green instead of black. Under such condition, on pressing the button the terminal will display page **1** (**#1**).

8.6.6.2 “Set Bit” button

The button works exclusively on certain bit (masks) of a variable. The mask is the binary representation:
$38_{10} = 0010\ 0110_2$
bit 0 = 0
bit1 = 1
bit2 = 1
bit3 = 0
…
when a mask value equal to 38 corresponds to a modification of bit 1, 2 and 5.
Figure 8.42 shows the “**SET BIT**” button, while Figure 8.43 and Figure 8.44 show its Properties table.

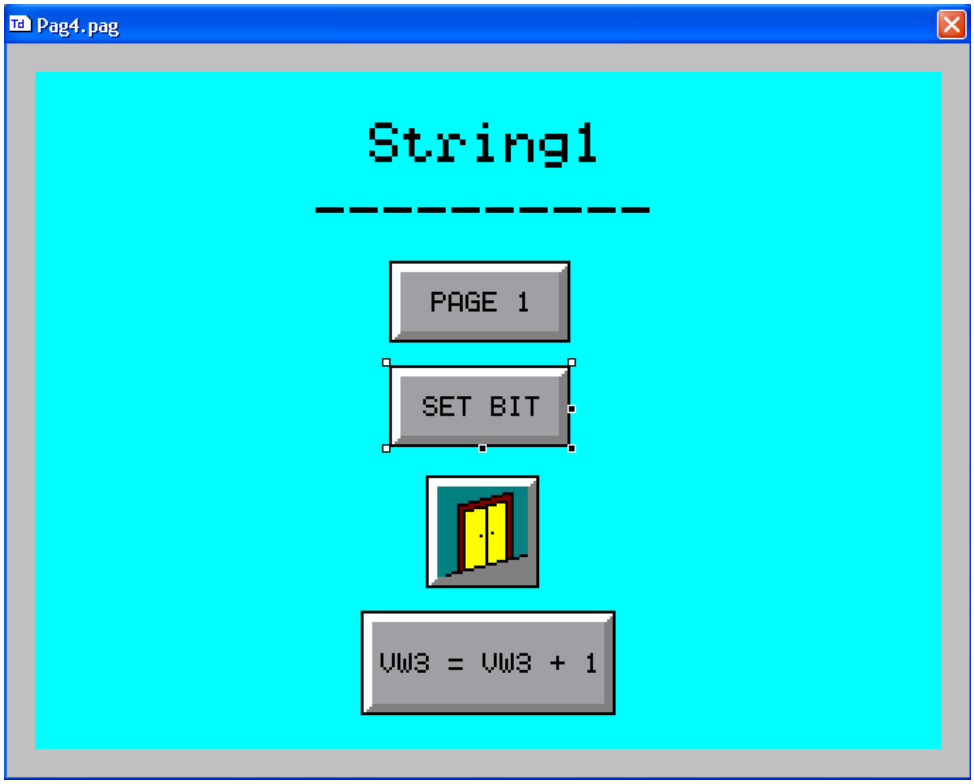


Figure 8.42: “Set Bit” button object of “SET BIT” on page Pag4

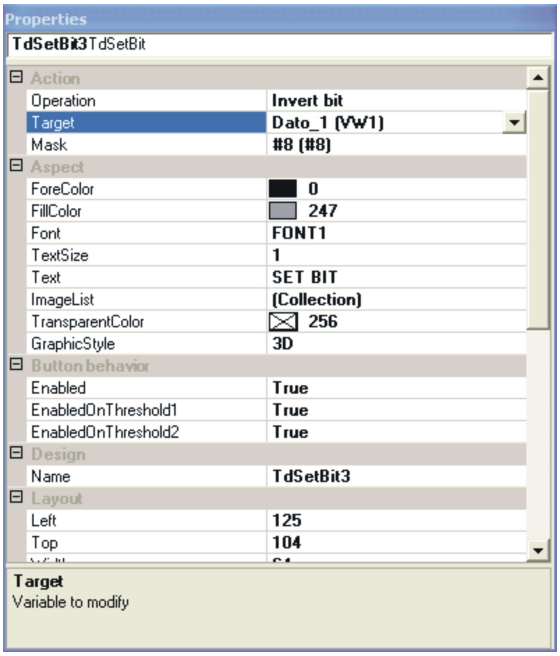


Figure 8.43: First part of Properties table of “Set Bit” button object "SET BIT" on page Pag4

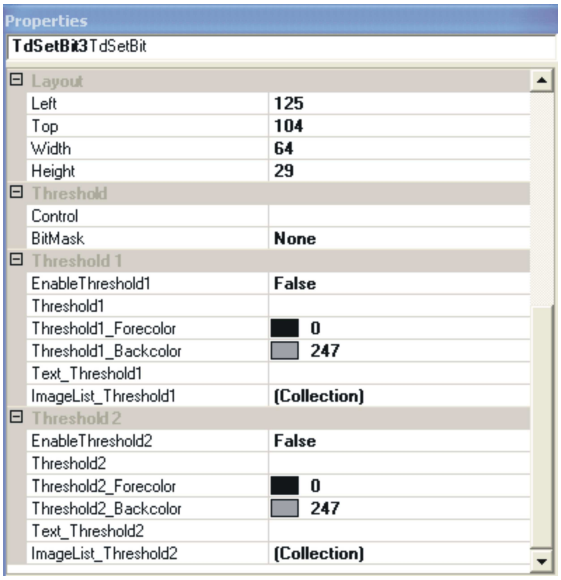


Figure 8.44: Second part of Properties table of “Set Bit” button object of "SET BIT" on page Pag4

The properties of this object are the exact same as those in the Go to page button (see par. 8.6.6.1), apart from the action, naturally:

Action: the **Operation** field has the following options:

- **Set bit**: Set to 1
- **Reset bit**: Set to 0
- **Invert bit**: Inverts bit

- **Set real-time**: Continues to set to 1 until the button is pressed
- **Reset real-time**: Continues to set to 0 until the button is pressed.

The **Target** field should be associated with the variable to modify. **Mask**, which is inserted in decimals, like constants or variables, should always be in the binary representation.

With reference to the Properties table in Figure 8.43 and Figure 8.44, the "**SET BIT**" object is a button which, each time it's pressed, inverts bit **3** of the **VW1** word (the mask is the decimal numeric constant #8 = 8_{10} = 0000000000001000_2). It is always enabled, there is no need for control variables and the thresholds are disabled.

8.6.6.3 "Assign" button

This type of button is used for two typical functions:

1. Assigning type X1 = X2
2. Mathematical operations type X3 = X3 + X4

Figure 8.45 shows an "Assign button".

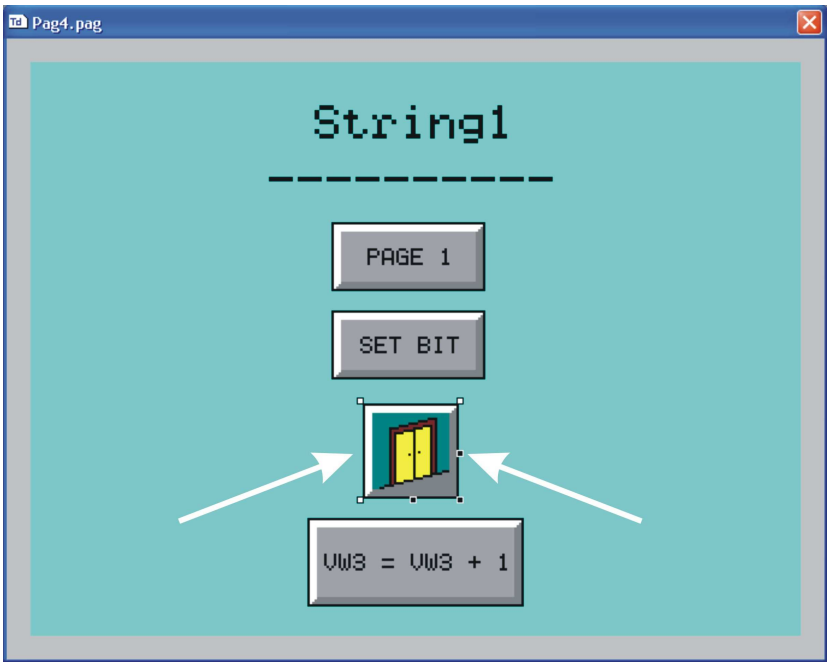


Figure 8.45: "Assign button" object on page Pag4.

Figure 8.46 and Figure 8.47 show the Properties table for the object.

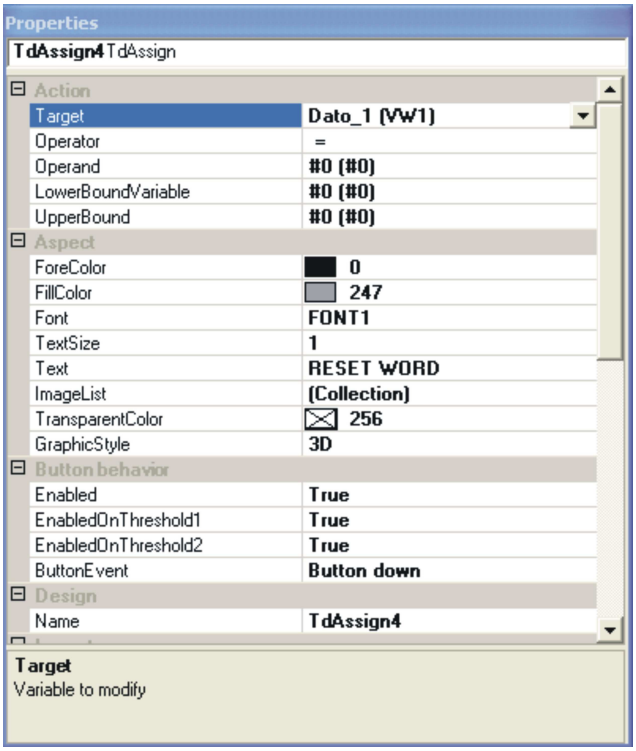


Figure 8.46: First part of Properties table of “Assign” button object on page Pag4

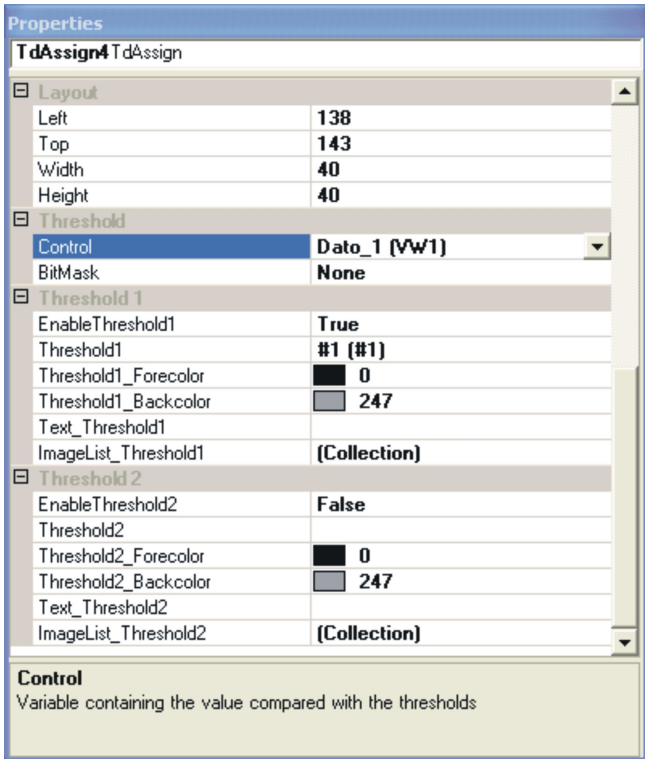


Figure 8.47: Second part of Properties table of “Assign” button object on page Pag4

The properties of this object are identical to those of the “Go to page” button (see par. 8.6.6.1) and the “Set Bit” button (see par. 8.6.6.2) apart from the action, naturally:

Action: the **Target** field contains the receiver operand of the operation, i.e. the variable to modify.
Operator: “**+**”, “**-**“ for maximise and minimise operations, “**=**” to assign.
Operand: the source operand of the operation, which can be a constant or a variable.
LowerBoundVariable and **UpperBound** are the limits within which the result of the assignment operation must be maintained.

For the formula **X1 = X1 + #1**
Destination : **X1**
Operator: **+**
Operating: **#1**

For the formula **X2 = #0**
Destination: **X2**
Operator: **=**
Operand: **#0**

Button behaviour: the **ButtonEvent** field defines the pressing type of the button on the terminal which determines its activation. There are 3 types:

- **Button down**: the operation is performed once on pressing the button.
- **Button up**: the operation is performed once on releasing the button.
- **Autorepeat**: the operation is performed multiple times with increasing frequency as the amount of times the button is pressed increases.

With reference to the Properties table in Figure 8.46 and Figure 8.47, the object indicated with the arrows in Figure 8.45 is an always enabled button which on pressing it on the terminal

performs the operation **VW1 = 0**, and therefore assigns the value **0** to the **VW1** variable. If the **VW1** variable is less (<) than **1**, the image above the button is the "closed door" (the **Door_1_Close.bmp** image is loaded from the **Library** folder in the **ImageList** field). If the **VW1** variable is greater or equal (≥) to **1**, the image above the button is an "open door" (the **Door_1_Open.bmp** image loaded from the Library folder in the **ImageList_Threshold1** field).

Figure 8.48 shows another "Assign" button, "**VW3 = VW3 + 1**" and Figure 8.49 and Figure 8.50 shows the Properties table.

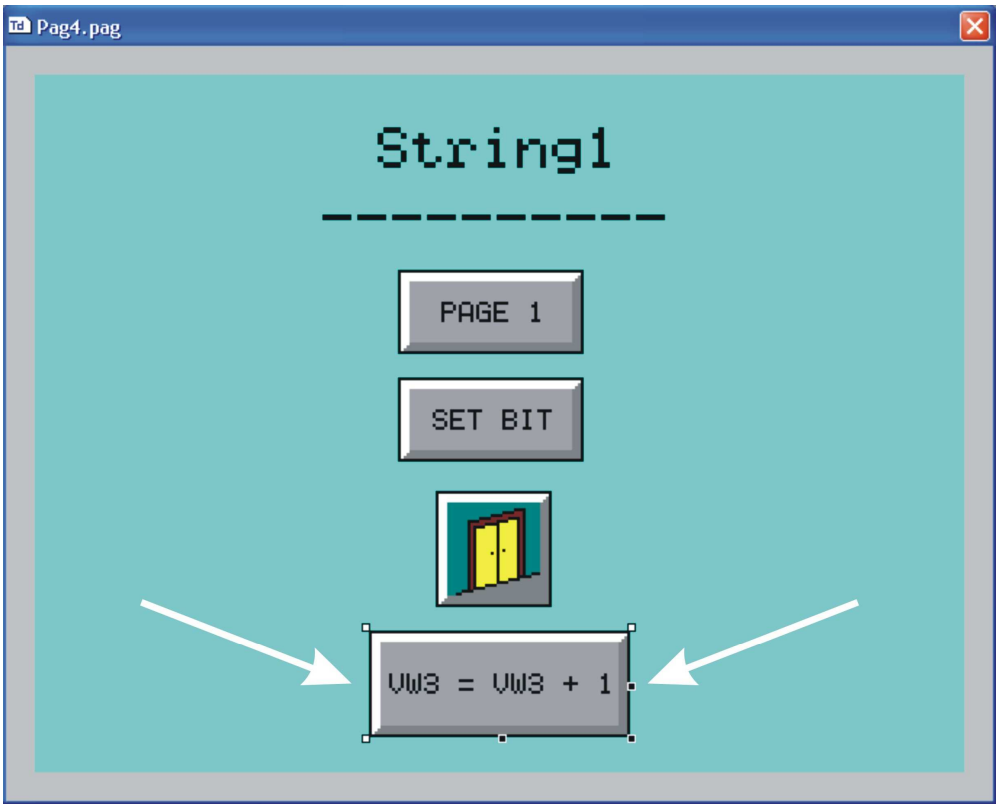


Figure 8.48: "Assign" button object "VW3 = VW3 + 1" on page Pag4.

The object indicated is a button that performs the operation corresponding to the text that reports, **VW3 = VW3 + 1**. It is always enabled, within the limits reported (the operation is performed if the **VW3** variable is between the limits reported, **0** and **5**), and the controls on the thresholds are disabled.

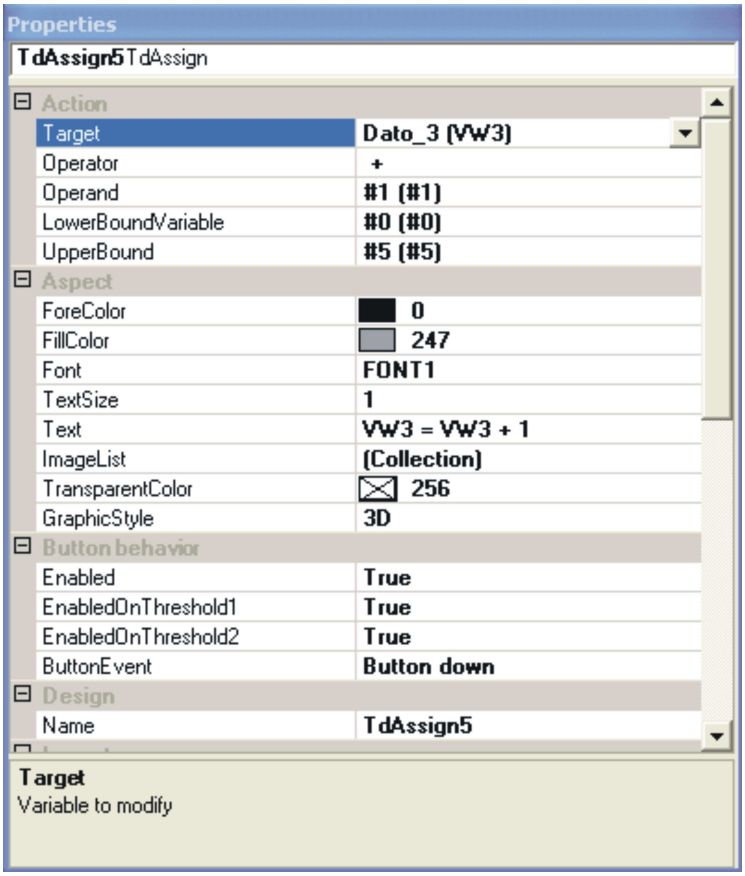


Figure 8.49: First part of Properties table of “Assign” button object “VW3 = VW3 + 1” on page Pag4

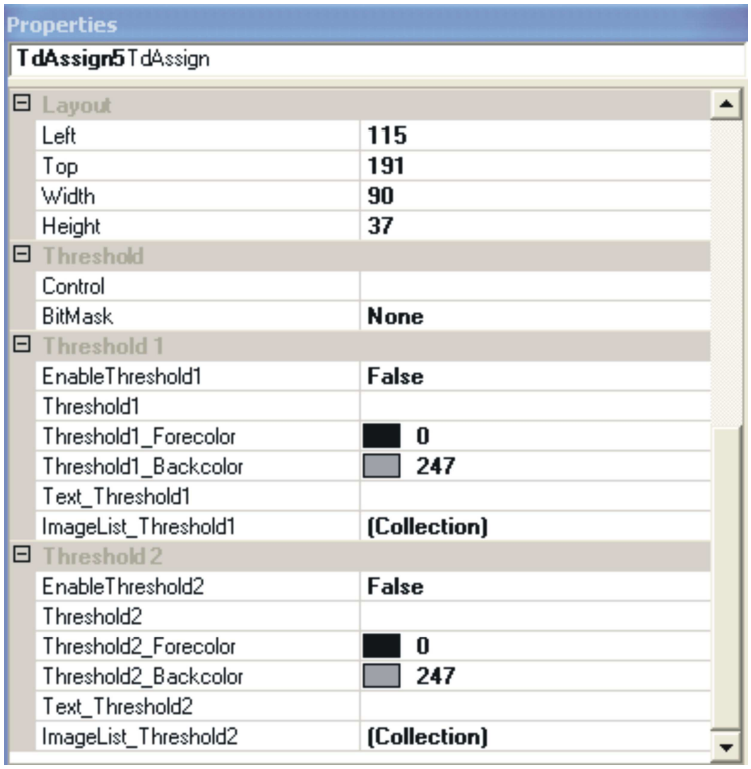


Figure 8.50: Second part of Properties table of “Assign” button object “VW3 = VW3 + 1” on page Pag4

8.6.7 Bitmap object

To display an image starting with a file in bitmap format, select the corresponding button in the Toolbar and click on the window that represents the page. An image will be displayed as reported in Figure 8.51

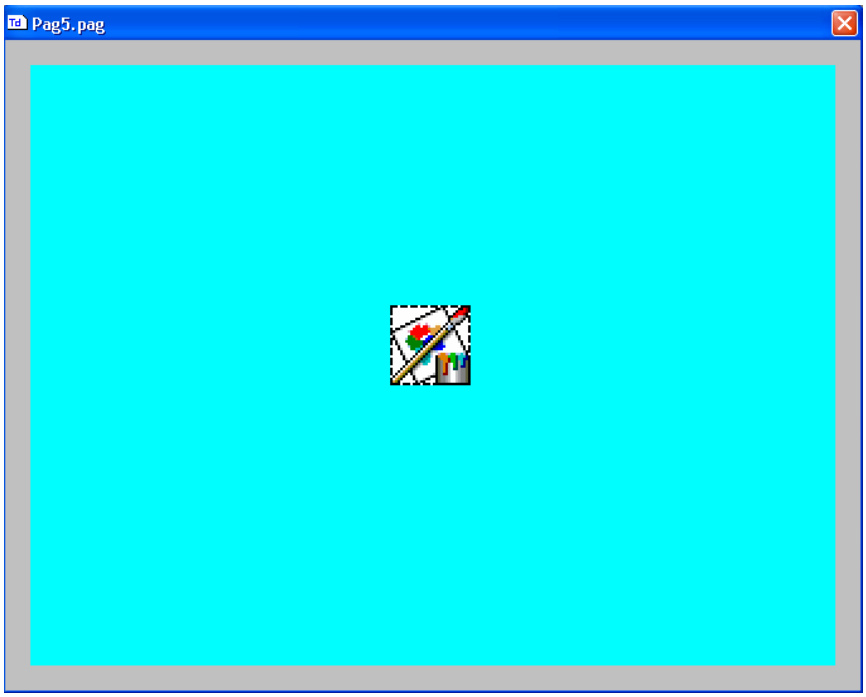


Figure 8.51: Bitmap object on page Pag5.

To replace this first image with one of your choice, you must open the specific edition of the set, as shown in Figure 8.52

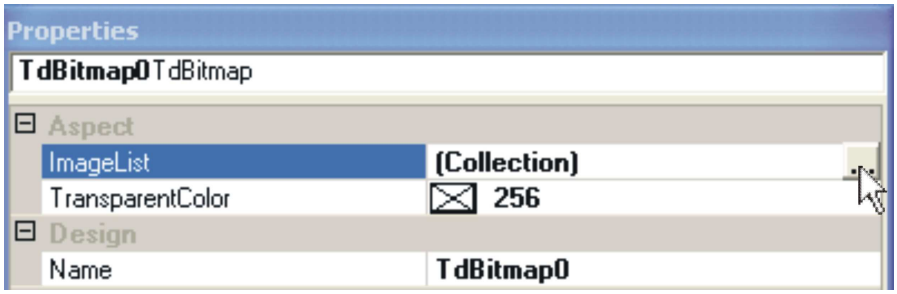


Figure 8.52: Opening of editor for ImageList of Bitmap object.

The editor is the same as that in Figure 8.41. The first image is called **Bitmap** and until it is removed (Remove button in the editor) or moved from the first position, the desired image cannot be displayed.
After selecting an image (the **Link_Login.bmp** image is loaded from the **Library** folder), the editor is as follows:

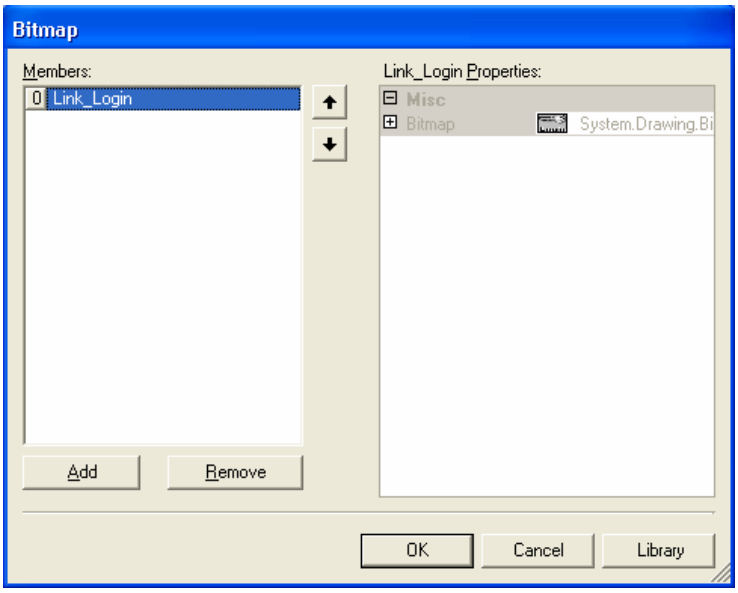


Figure 8.53: Editor for ImageList of Bitmap object.

The new image loaded is visible in Figure 8.54.

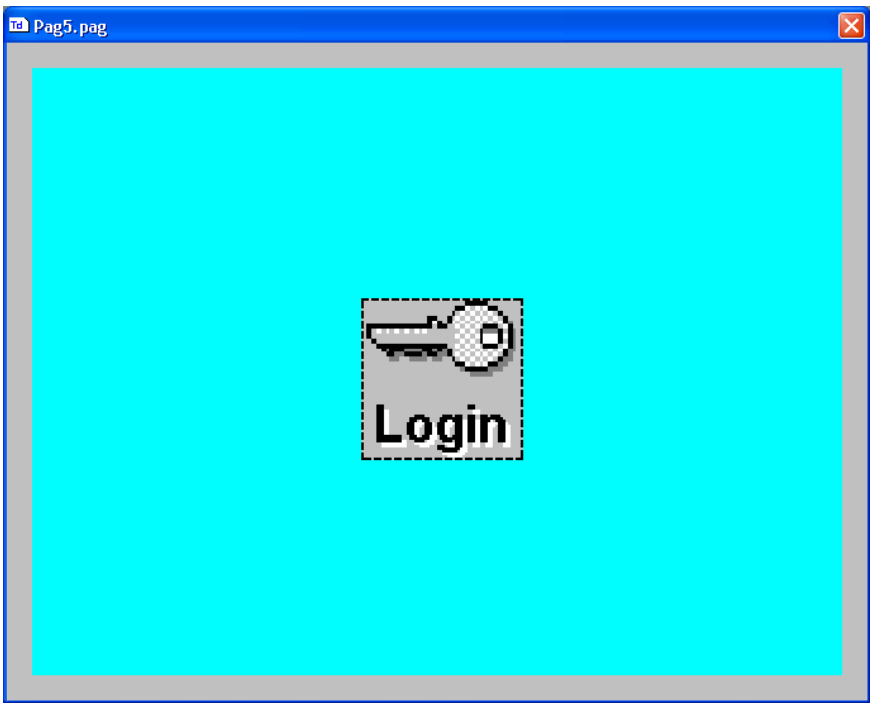


Figure 8.54: Bitmap object on page Pag5.

Figure 8.55 shows the Properties table of the object.

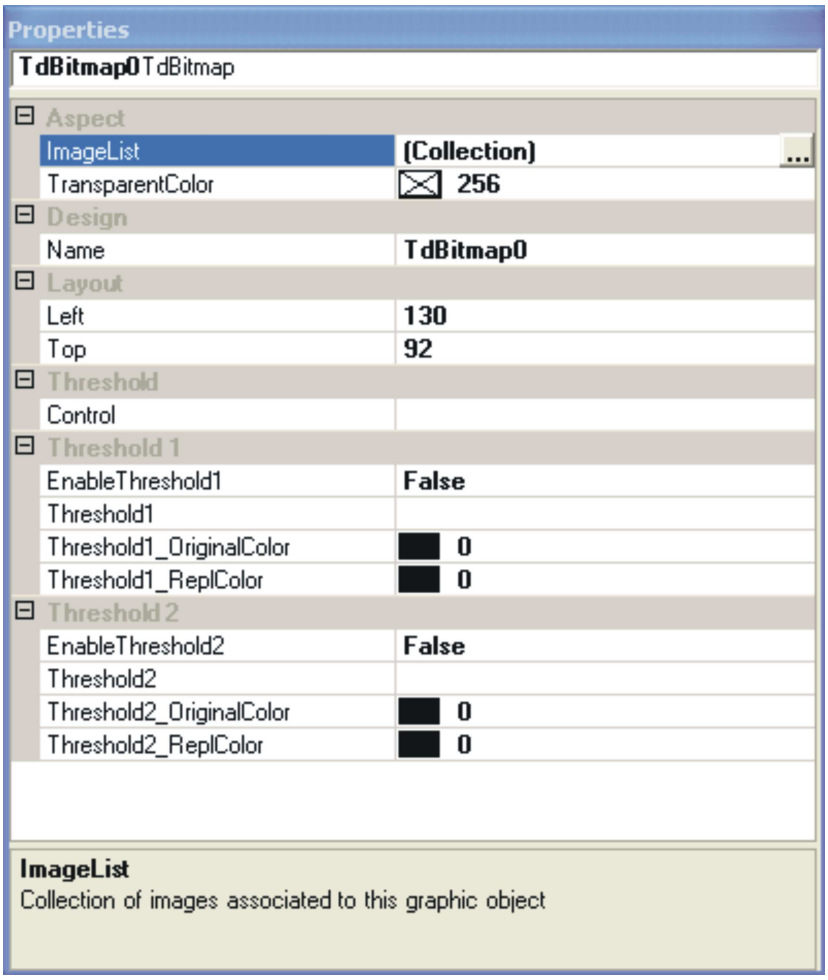


Figure 8.55: Properties table of Bitmap object on page Pag5.

Aspect: the **TransparentColor** field has the same significance as for the Button objects (see par. 8.6.6).

All the fields were already analysed for other objects. It is also possible for the Bitmap object to use a **Control** variable and two thresholds.
You can replace a colour (**Threshold_OriginalColor**) with another (**Threshold_ReplColor**) when the conditions set by the control variable are met, with the same methods described for the other objects.

Figure 8.54 shows a bitmap object without a transparent colour and without set threshold variables.

8.6.8 Symbolic Field object

Starting with a bitmap list, it displays an image indexed by the value of a variable. The insertion of images occurs as in the case of the Bitmap object.
If the **Input** field is set as **True** the operator can also choose the image to display.
To import this object on the page, you must re-size it by dragging the mouse or, subsequently, use the Properties window.
Figure 8.56 shows a Symbolic Field object.

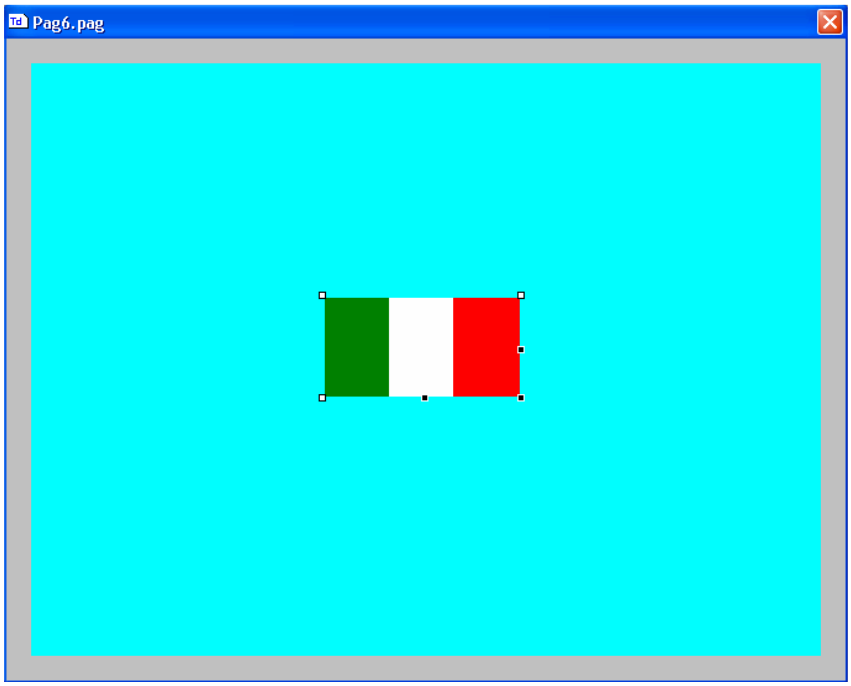


Figure 8.56: Symbolic Field object on page Pag6

The editor to insert images is shown in Figure 8.57; the Properties table is shown in Figure 8.58. In this case the object is a list of 4 images (4 flags of an equal amount of states), indexed by the **VW4** variable, which cannot be modified by the user.

Aspect: the **RipeatList** field refers back to the number of languages selected for the project. If set as **True** the list of images inserted will be automatically duplicated for each of the project

languages. For the case in question, there are 2 languages (see par. 8.5.1), so they will have the same 4 images for both.

Language1: Italian
VW4 = 0 FlagIT.bmp
VW4 = 1 FlagGB.bmp
VW4 = 2 FlagFR.bmp
VW4 = 3 FlagESP.bmp

Language2: English
VW4 = 0 FlagIT.bmp
VW4 = 1 FlagGB.bmp
VW4 = 2 FlagFR.bmp
VW4 = 3 FlagESP.bmp

If instead the **RipeatList** field is set as **False** a list of images should be inserted for each project language. In our example, only the **RipeatList** field changes giving 2 images for each language selected.

Language1: Italian
VW4 = 0 FlagIT.bmp
VW4 = 1 FlagGB.bmp

Language2: English
VW4 = 0 FlagFR.bmp
VW4 = 1 FlagESP.bmp

Aspect: the **ImageList** and **TransparentColor** fields are identical to those of the Bitmap field (see par. 8.6.7).

Input: if the **AcceptInput** field is **True**, the operator can choose which image to display within the limits set in **MinIndex** and **MaxIndex**.

Memory: in the **Index** field the variables that index the list of images should be connected.

The **BitMask** field is the mask that determines the scrolling of images. If selected as **None**, each image will have an Index variable value associated to it, as a decimal number (0, 1, 2,..., N-1, where N is the number of images on the list). This is the case in the example shown in Figure 8.56. If selected as **bitN**, selection is possible only between the two images: the image in the **0** position (see Figure 8.57) if bit **N** is equal to **0**, the image in position **1** if bit **N** is equal to **1**. Any other possible images on the list will be ignored.

The **Layout** and **Design** fields are identical to those in the Bitmap field (see par. 8.6.7).

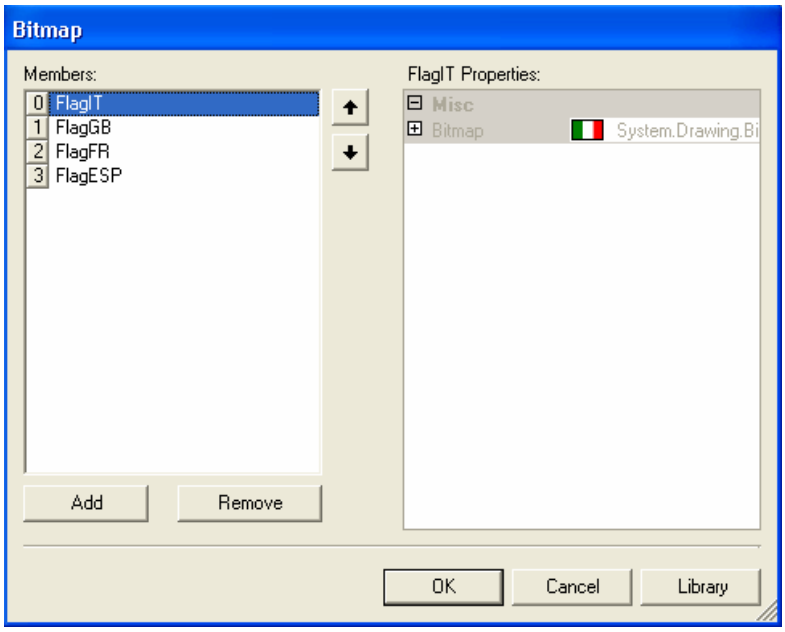


Figure 8.57: Editor for ImageList of Symbolic Field object.

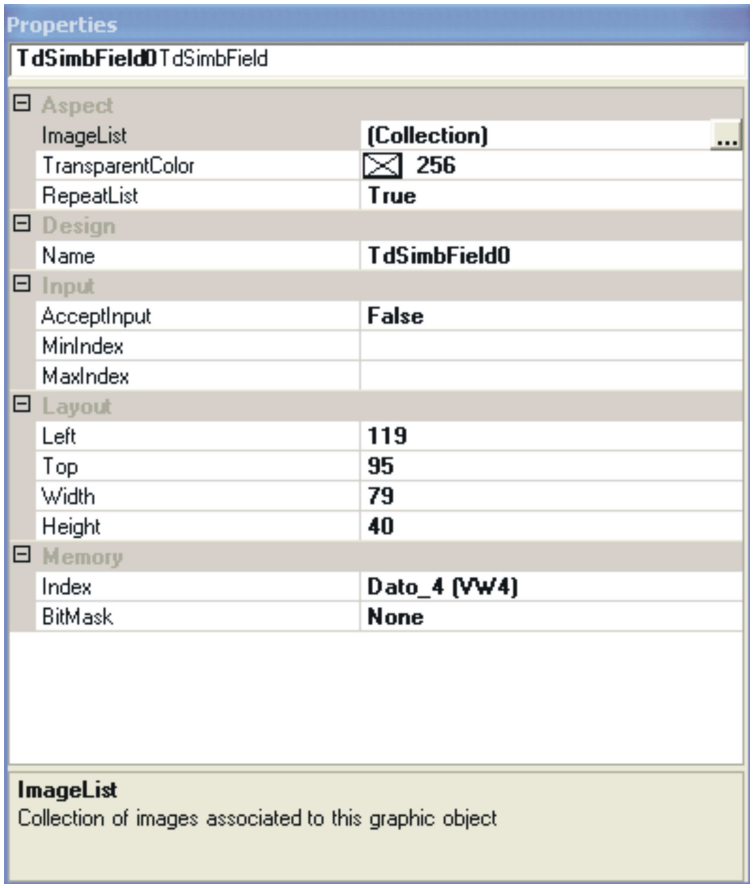


Figure 8.58: Properties table of Symbolic Field object of page Pag6

8.7 Program compilation and transfer

The last step for configuration is transferring the programme to the graphical terminal memory. As mentioned above, the development of an application requires using TdDesigner and PlProg simultaneously. In particular, **the compilation operation** (from “File/Compile”) **should be done with PlProg open** in order to allow TdDesigner to change the graphical settings of the .plp file. It is recommended to read the terminal manual for details on interaction between logic and graphics (chapter 4).

It should also be noted that, on the contrary to what happens for PlProg, TdDesigner automatically saves the compiled project. If the .plp file is also saved it will contain all the information necessary for later programming, including graphics.

Figure 8.59 and Figure 8.60 outline the procedures with the correct sequence of operations to create a new project and for using any changes to an already existing project.

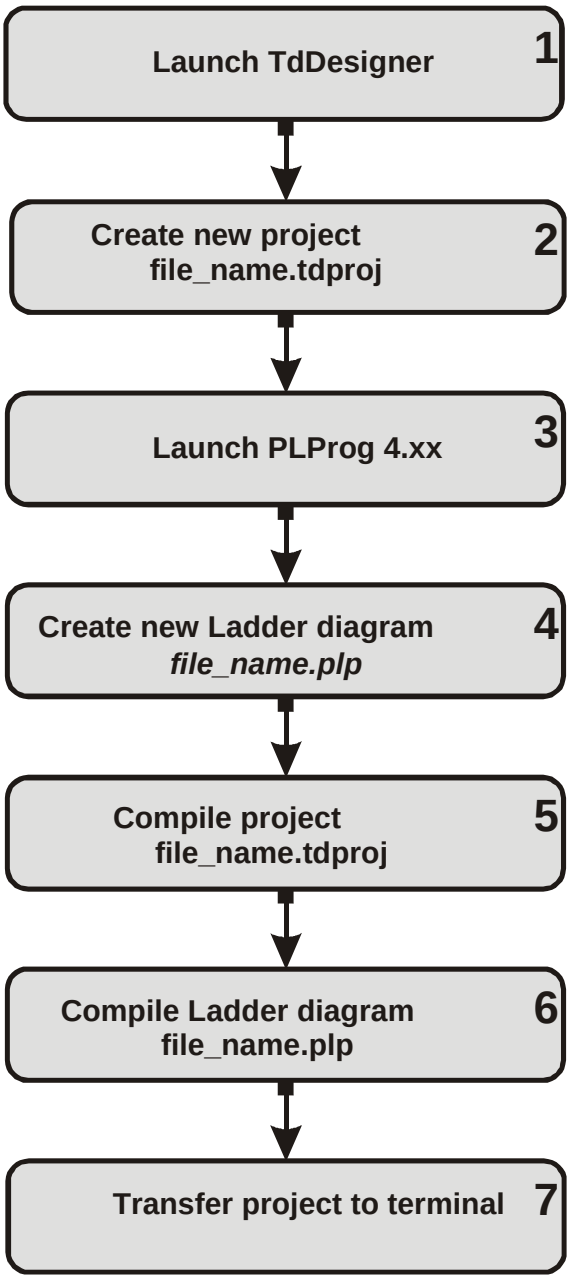


Figure 8.59: Create a new project.

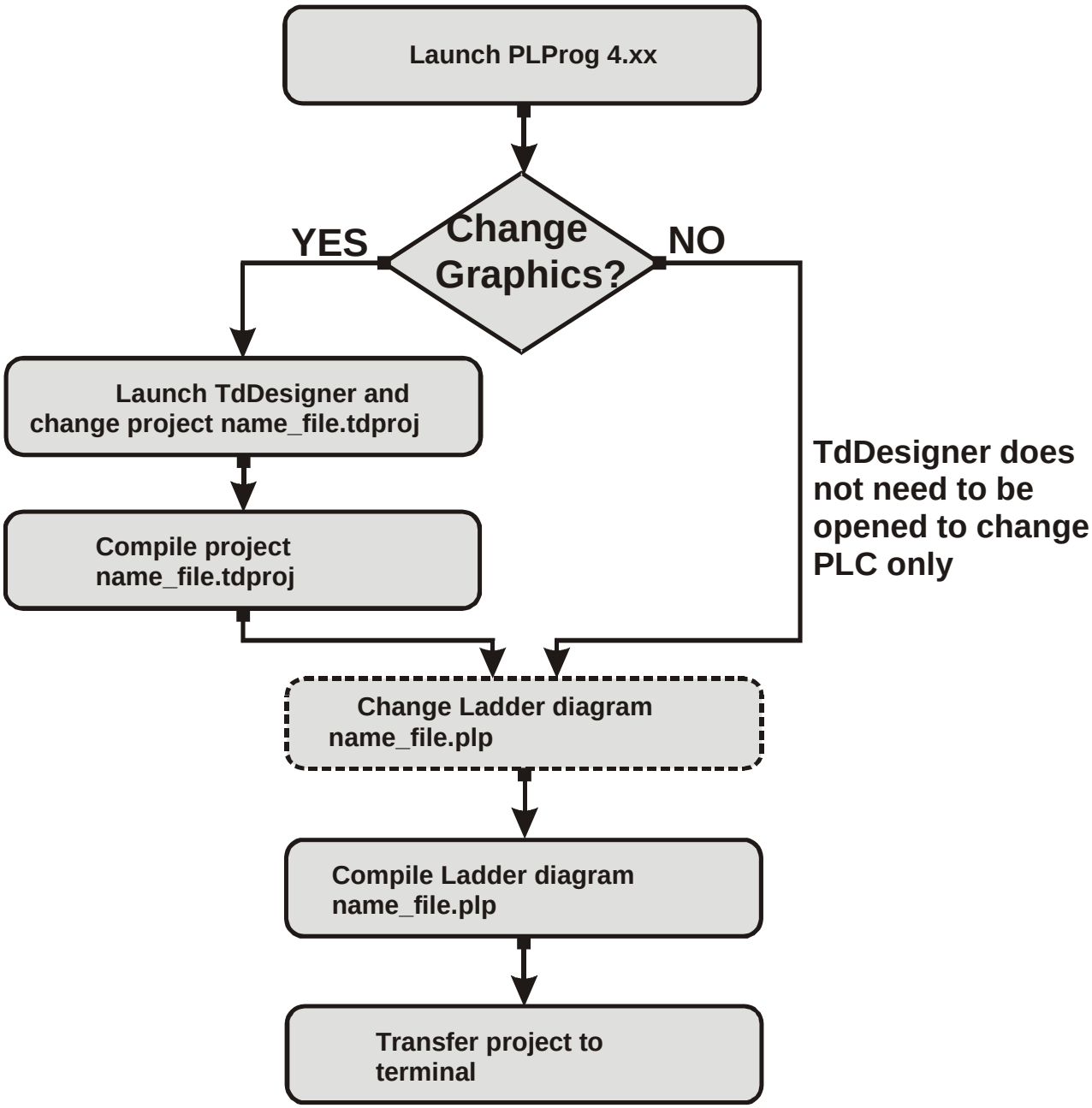


Figure 8.60: Use and if necessary change an already existing project.

9 Glossary

Boot: program residing in the PLC memory which manages the loading of firmware. Not accessible to end-users.
Compilation: in PLProg's case, this term indicates the process that translates the ladder diagram developed by the user into a format suitable to transmit to the PLC.
Debug: action to find errors in program logic. PLProg possesses a debug mode through which it is possible to view the state of contacts and coils directly on the ladder diagram while the PLC is running. It is not a simulation, the program has to be running on the PLC.
Editor: program to draft and save documents. The PLProg editor provides tools to create a ladder diagram.
Firmware: program that manages PLC functions.
Ladder: short for “Ladder Diagram”. It is a contact programming language derived from diagrams of control systems made with electromechanical relays.
It is based on the concept of contact and coil. At first it was possible to manage only binary logic functions, later it was extended to deal with integer and/or real numbers.
Memory card: small electronic card with EEPROM memory where you can save PLC programs.
Modbus: communication protocol typically used by supervision systems.
Special Marker: memory area containing all the data needed by the ladder program to interact with PLC hardware. For instance, there are values for analogue inputs and trimmers, encoder sums and sets, and settings for serial port communication.
Variable: The program running on the PLC can transfer quickly data from and to certain locations on the RAM memory. These locations are accessible for the ladder diagram and are commonly called variables. The user must choose the type of variable to use according to the data being transferred. For instance, if you want to save an integer over 32767 in area V it is necessary to use a VD memory (V Double Word).

10 ITALIANO

11 Introduzione

In molti casi, con gli sviluppi tecnologici degli ultimi anni, il PLC non è più una scelta ma una necessità. La complessità delle applicazioni che il mercato dell’automazione richiede rende infatti sempre più necessario questo tipo di dispositivi.
Pixsys ha realizzato una variegata serie di prodotti, semplici da installare e configurabili attraverso un ambiente di sviluppo che, installato sul vostro PC, mette a disposizione tutti i tools di programmazione e le librerie che consentono di realizzare rapidamente le più comuni applicazioni dell'automazione industriale.
L’obiettivo di questo manuale è far conoscere i concetti che stanno alla base del funzionamento del software PLprog senza insistere su dettagli di prodotto o su fondamenti di programmazione Ladder che vengono trattati in altra documentazione o dati per acquisiti da parte dell’utente.

12 Composizione KIT

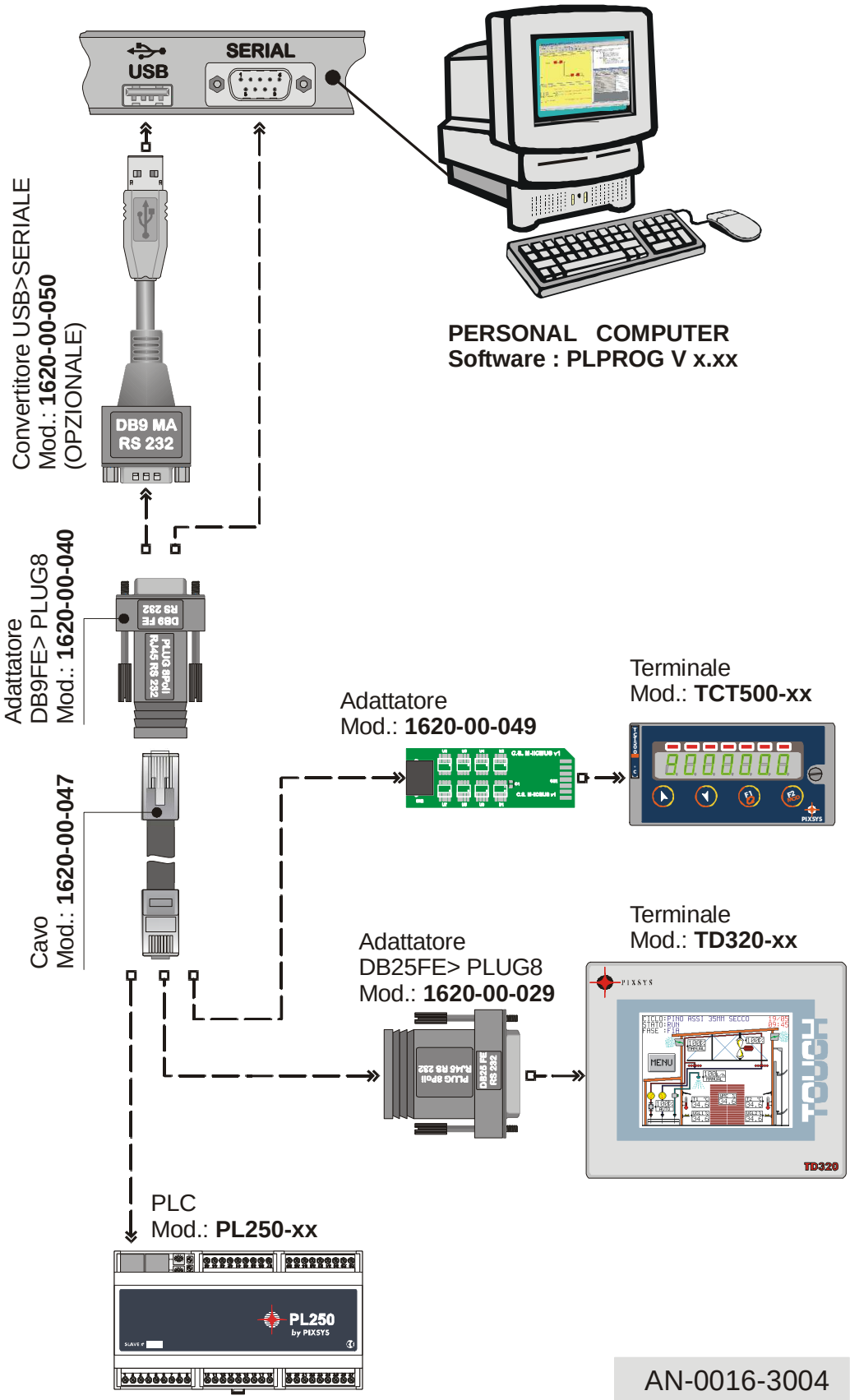


Figura 12.1: connessioni tra dispositivi Pixsys e PC tramite i cavi forniti con il Kit.

13 Installazione di PLprog

Inserire il Cd nel lettore del PC, il programma partirà automaticamente e guiderà l'utente nel processo di installazione attraverso una serie di finestre di dialogo. Alcune di queste sono riportate di seguito per evidenziare i passaggi più significativi.

Figura 13.1: la prima finestra permetterà di slezionare la lingua da usare durante l'installazione.

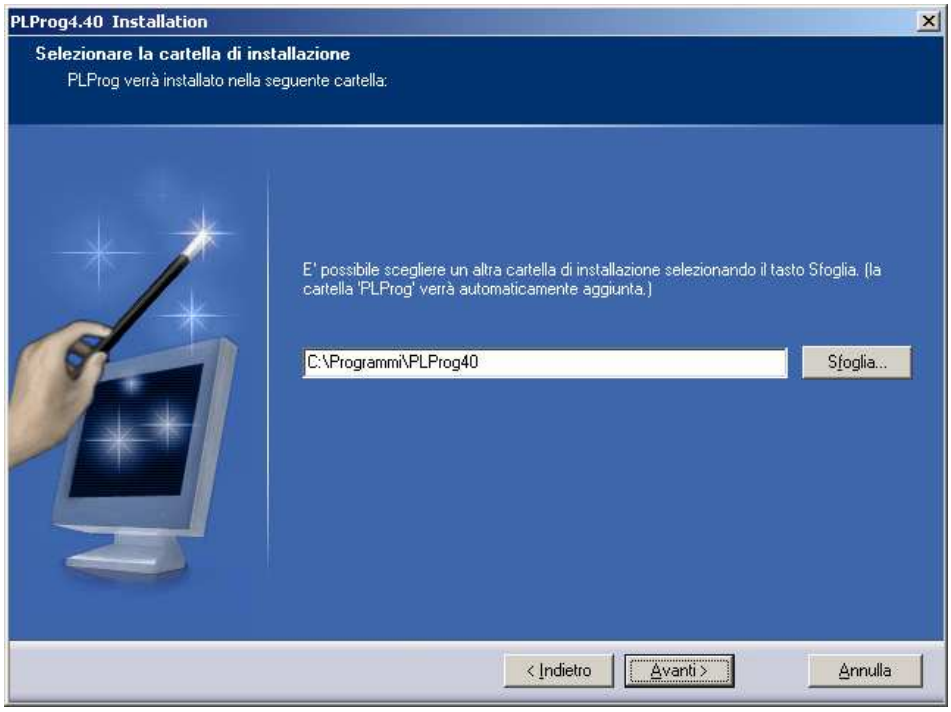


Figura 13.2: scelta della cartella di installazione.

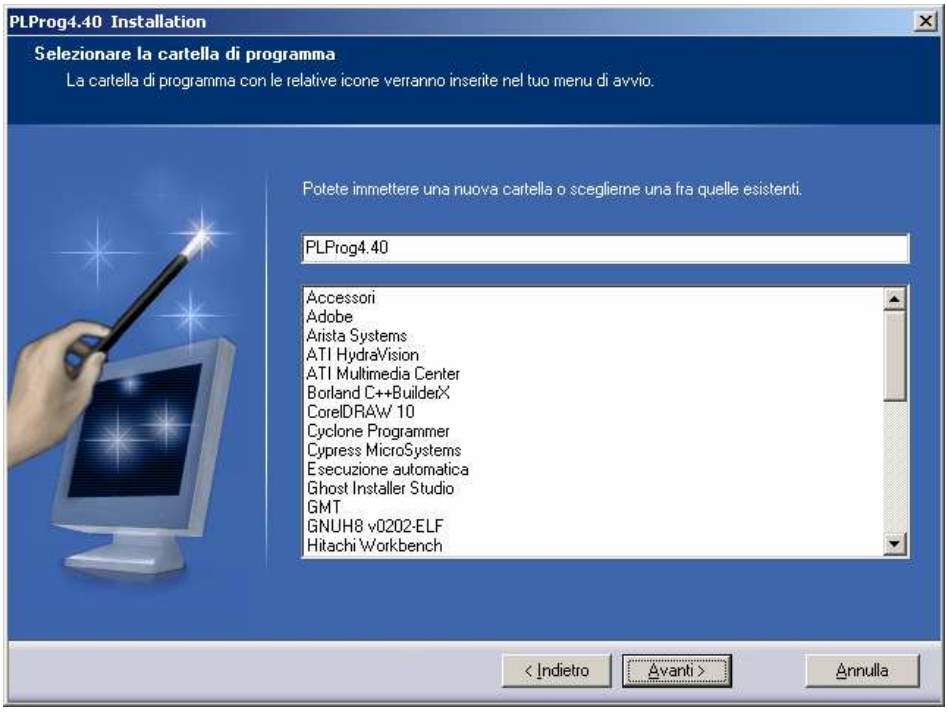


Figura 13.3: inserimento nel menu di avvio.

PLProg è ora installato e funzionante, può essere avviato dal menu Start\Programmi o dal collegamento sul Desktop.

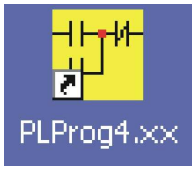


Figura 13.4: il programma di installazione crea un'icona di collegamento sul desktop.

La lingua usata dal programma può essere scelta tra Italiano e Inglese dal menu principale, si veda a tal proposito il paragrafo 15.1.4.

13.1 Prova hardware

E' possibile verificare immediatamente e in pochi passi il corretto funzionamento del materiale ricevuto.
Innanzitutto occorre ricreare i collegamenti tra PC e dispositivo illustrati in Figura 12.1, avviare PLProg e impostare il numero di porta seriale.

Figura 13.5: Dal menu 'Porta comunicazione' si accede al sottomenu 'Com' che apre la finestra di selezione.

E' possibile anche selezionare che supporto seriale utilizzare per la connessione e/o programmazione dei PLC (dalla versione software di PlProg 4.63). E' possibile la connessione tramite RS232 ed RS485 (per tutti i PLC), la programmazione tramite RS232 (per tutti i PLC) e RS485 (per il solo PL260-11AD). In quest'ultimo caso, selezionare il **Tempo di attesa risposta del PLC** (default 2msec) ed il **Ritardo di ritrasmissione del PC** (defalut 2msec)

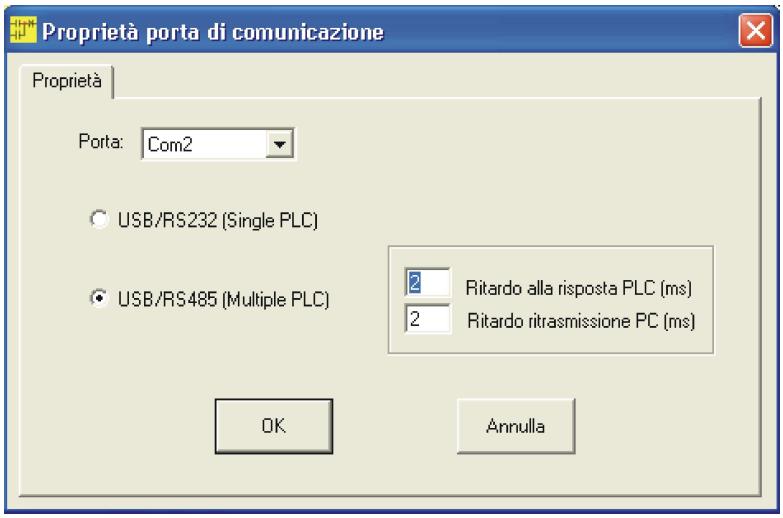


Figura 13.6: Selezione connessione/programmazione tramite RS485 ed impostazione parametri Modbus master per il PC

Nel caso in cui si sia selezionata la casella USB/RS485 (Multiple PLC), all'uscita si dovrà selezionare quale PLC della linea

eventualmente presente dovrà essere interrogato dal PC (fino ad un massimo di 254 dispositivi).

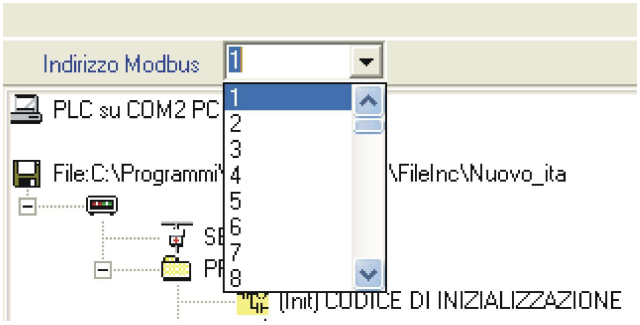


Figura 13.7: Selezione numero PLC da interrogare in modalità Multiple PLC USB/RS485

Con il PLC alimentato correttamente, verificare di aver scelto la porta giusta con un click sulla prima riga della finestra di stato (cfr.15.4):

Figura 13.8: con un click sull'area evidenziata, PLProg proverà a comunicare con il PLC attraverso COM1.

Se si ottiene un messaggio di errore significa che la porta selezionata non è quella giusta, altrimenti il programma visualizzerà alcune informazioni relative al PLC collegato.

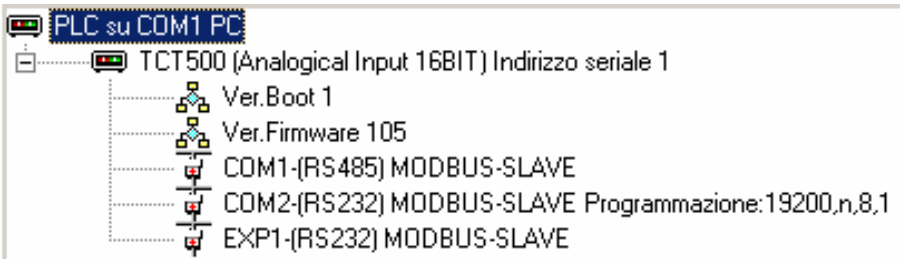


Figura 13.9: il collegamento e le impostazioni sono stati eseguiti correttamente.

Fatto questo, premendo il tasto  si può accedere alle risorse del computer. Nella cartella
c:\PLProg40\Examples\TestHardware
si trovano dei file il cui nome è
TestHardware_*NomePLC*:
aprire quello corrispondente al dispositivo collegato.
Il programma così caricato deve essere compilato e inviato al PLC premendo nell'ordine i tasti  e  della barra degli strumenti: se tutto funziona correttamente i relé di uscita cominceranno a muoversi e, nel caso del TCT500, sul display comparirà una scritta.

14 Come usare PLProg

Questo capitolo introdurrà all'utilizzo di PLProg attraverso la descrizione delle operazioni più comuni e dei passaggi che conducono alla programmazione di un PLC.
Come si è visto nei capitoli precedenti e in particolare nella Figura 12.1, PLProg è in grado di interfacciarsi con diversi strumenti; d'ora in avanti, per semplicità, chiameremo genericamente PLC il dispositivo collegato al PC tramite il Kit.
Per una descrizione esaustiva del significato di tutti i pulsanti e delle voci di menu si rimanda al prossimo capitolo.

14.1 Setup

Per realizzare una nuova applicazione, la prima cosa da fare è il Setup del dispositivo che si intende usare. Se si è già superata la prova hardware (par. 13.1) significa che i collegamenti e la porta seriale sono impostati correttamente, altrimenti ripetere i passaggi illustrati in Figura 13.5 e Figura 13.8.
Con un click sull'area dell'editor ancora vuoto o sul tasto  della barra degli strumenti si aprirà la finestra di Figura 14.1.

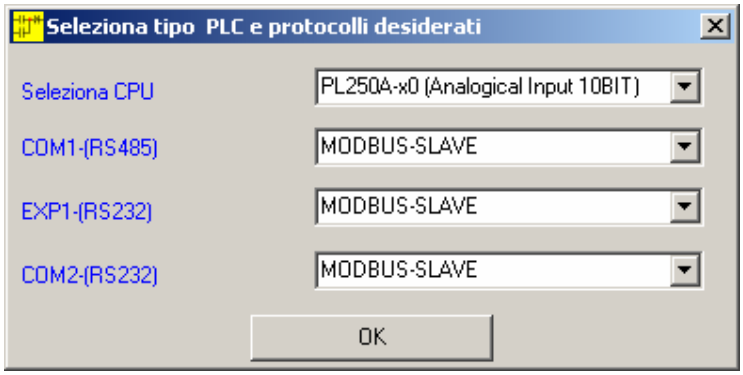


Figura 14.1: finestra di setup. Compare ogni volta che si apre un nuovo file e può essere in seguito richiamata dall'utente con un click nell'area 'File' della finestra di stato (cfr. paragrafo 15.4).

Il campo "Seleziona CPU" consente di scegliere il tipo di PLC mentre i tre successivi selezionano il protocollo di comunicazione delle porte seriali disponibili. Il protocollo assegnato a COM2 è sempre "MODBUS-SLAVE" perché questa è la porta di comunicazione solitamente usata per programmare il PLC con la RS232.

Il setup può essere fatto anche se nessun PLC è fisicamente connesso al PC, inoltre, in fase di stesura del diagramma ladder, PLProg non fa nessun controllo sulla corrispondenza tra il dispositivo collegato e quello selezionato. Un'eventuale incompatibilità verrà rilevata, e segnalata con un messaggio di errore, nel momento in cui si tenterà di trasferire il programma nella memoria del PLC (cfr. par. 14.4).
Per avere una panoramica completa della situazione si può far riferimento alla finestra di stato (par. 15.4) che riporta tutte le informazioni relative al dispositivo collegato al PC e alle impostazioni dell'utente.

14.2 Creare e modificare un diagramma ladder

La finestra dell'editor di PLProg è ricca di funzionalità, per essere sfruttata appieno richiederà quindi una descrizione approfondita e un po' di pratica da parte dell'utente. Può essere vista come un foglio su cui disporre i classici oggetti che compongono un programma ladder: contatti, bobine, collegamenti e righe di commento.

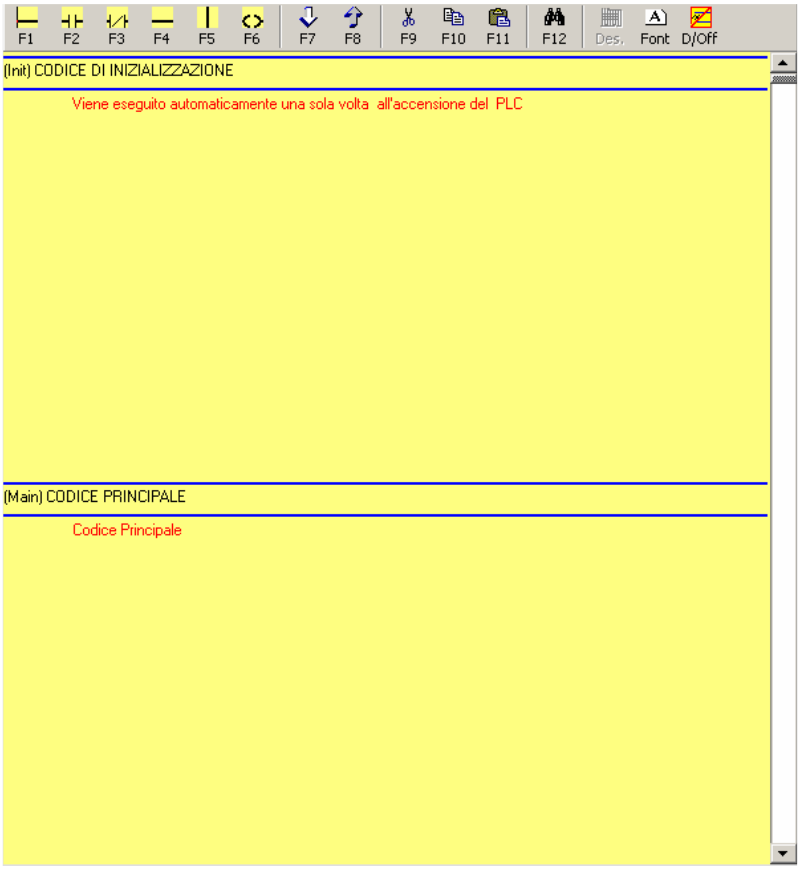


Figura 14.2: finestra dell'editor di PLProg.

Questi oggetti si inseriranno tramite i tasti della toolbar (Figura 15.19) o direttamente utilizzando il mouse.
Si può subito osservare che l'area dell'editor è sensibile alle azioni del mouse: come illustrato in Figura 14.3 e Figura 14.4, il cursore cambia aspetto a seconda della zona su cui è posizionato ad indicare che gli oggetti devono essere disposti alternando i contatti e le bobine con almeno un tratto di collegamento.

Figura 14.3: se il cursore ha questo aspetto, la casella su cui si trova può ospitare un collegamento.

Figura 14.4: in questo caso il cursore si trova su una casella che può ospitare un contatto o una bobina.

Per tracciare un collegamento usare i pulsanti F1, F4 o F5 della toolbar o della tastiera, oppure trascinare il puntatore del mouse tenendo premuto il tasto sinistro.
Per inserire un contatto usare i pulsanti F2 o F3 oppure fare doppio click su un tratto di collegamento sul quale il puntatore abbia l'aspetto di Figura 14.4.
Per inserire una bobina usare il pulsante F6.
In ogni caso, se si usano i pulsanti da F1 a F6, l'oggetto corrispondente sarà posizionato nella casella evidenziata con un riquadro (vedi Figura 14.5). Questa selezione si può fare con un click del mouse sul foglio o con i tasti freccia della tastiera.

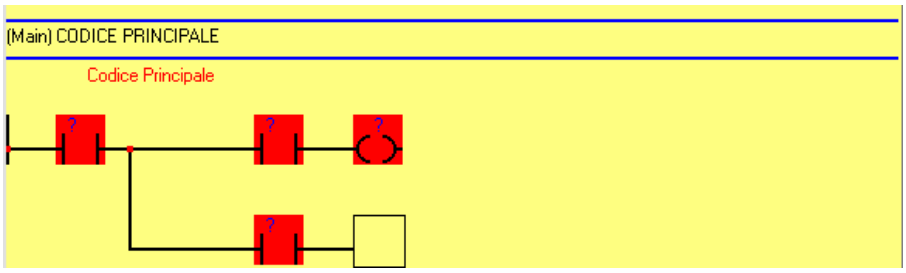


Figura 14.5: inizio di stesura di un diagramma ladder. Sono visibili i nuovi contatti e la casella selezionata per l'inserimento di un nuovo oggetto.

Questo primo esempio di diagramma ladder non ha alcun significato finché ai contatti e alle bobine non vengono associati delle funzioni o dei parametri del PLC. Un doppio click sul contatto apre la finestra "Inserisci/Modifica contatto".

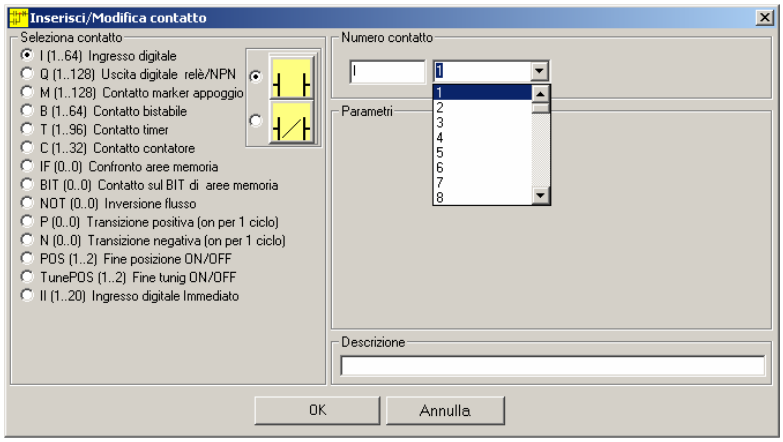


Figura 14.6: In questa finestra si possono scegliere il tipo di contatto (Normalmente Aperto o Normalmente Chiuso), il flag associato (I, Q, M...) ed eventualmente associare una descrizione

Un doppio click su una bobina richiamerà invece la finestra “Seleziona/Modifica bobina”.

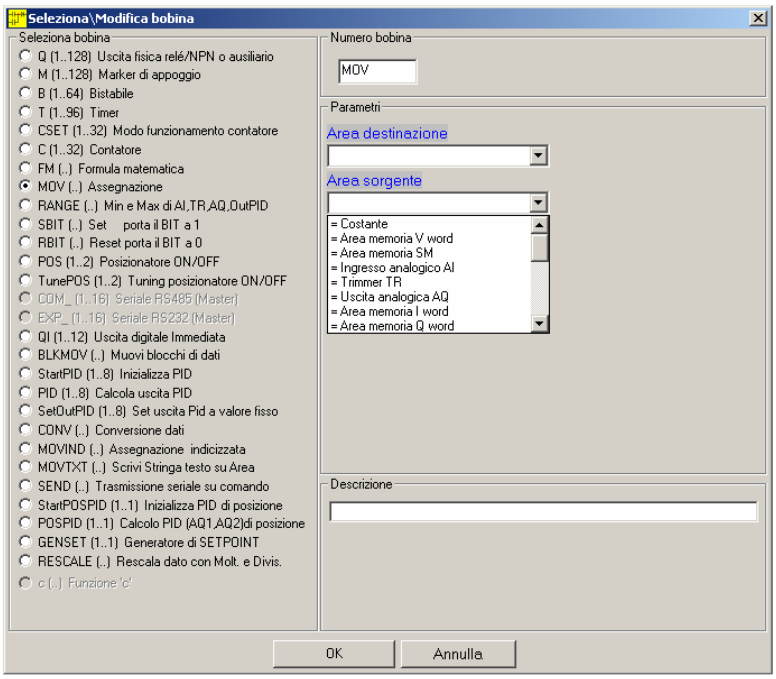


Figura 14.7: in questa finestra si possono associare alla bobina una funzione con i parametri necessari ed una descrizione.

Nell’esempio di Figura 14.8 sono stati associati ai contatti tre ingressi del PLC e alle bobine un’uscita e una funzione di assegnamento.

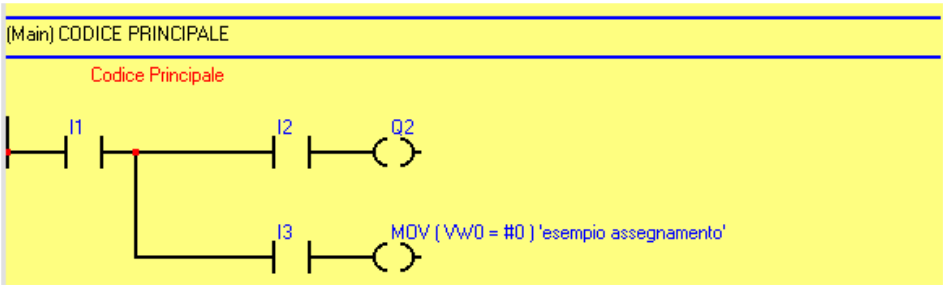


Figura 14.8: schema completato con l’inserimento delle funzioni e dei parametri.

Il diagramma così ottenuto può essere esteso aggiungendo nuove righe di codice con le tecniche appena descritte e può anche essere modificato. Con un doppio click su bobine e contatti si riaprono le finestre di Figura 14.6 e Figura 14.7 per modificare funzioni e parametri mentre con il tasto destro del mouse si può cancellare.

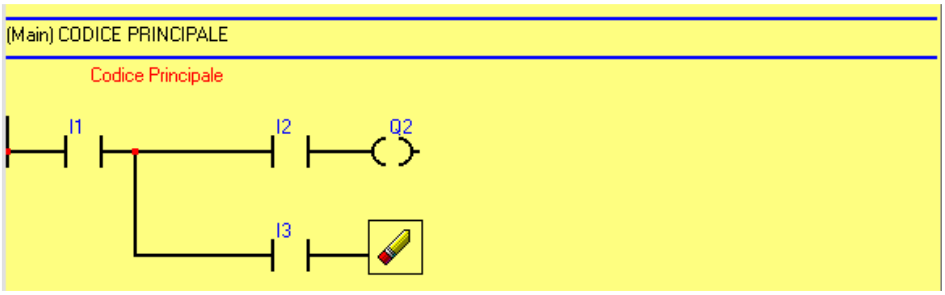


Figura 14.9: cancellazione di un oggetto. Il puntatore del mouse cambia aspetto.

Infine, un contatto o una bobina possono essere trascinati per linee orizzontali con il classico meccanismo “drag-and-drop”.

E’ possibile contrassegnare un’area del diagramma con un titolo per ottenere un programma più leggibile e indicizzato.
Con un doppio click su un’area vuota dell’editor si apre la finestra di Figura 14.10.

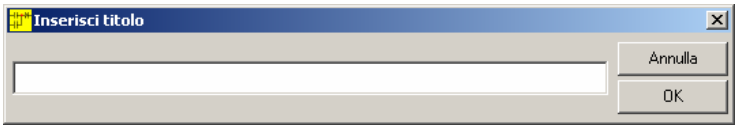


Figura 14.10: inserire il titolo nell’area testo e dare OK.

Se posizionati all’estremità sinistra del foglio, i titoli assumono un significato particolare diventando dei veri e propri “segnalibro”, particolarmente utili nei programmi molto lunghi.
Infatti in tal caso PLProg inserisce nella finestra di stato un collegamento (cfr. par. 15.4) dal quale si potrà in ogni momento “saltare” all’area di codice che segue il titolo.

14.2.1 Selezione e copia di un blocco

PLProg consente di copiare (tagliare) un blocco di righe per poterlo incollare in un’altra parte del programma o per salvarlo in un file (si vedano a tal proposito i par. 15.1.1.5 e 15.1.1.9).
Per la sua particolare importanza, questo argomento è organizzato in un altro paragrafo ed anticipa la descrizione della toolbar dell’editor che verrà fatta nel capitolo 15.3.
La procedura completa prevede quattro passi:

1. **Click su prima riga** – prima colonna del blocco che si vuole copiare (tagliare), nelle colonne diverse dalla prima si copia solo una casella;
2. **F10** (**F9**) da tastiera o con mouse su toolbar dell'editor (Figura 15.19), il puntatore del mouse cambia aspetto e diventa come in Figura 15.6;
3. **Click sull'ultima riga del blocco che si vuole copiare (tagliare) o salvare**, la colonna è ininfluente. Il puntatore torna all'aspetto tradizionale;
4. **click sulla riga a partire dalla quale si vuole posizionare il blocco acquisito**.
5. **Per incollare premere F11**.

Se non ci sono righe vuote sufficienti a ospitare il blocco, verrà eseguita una copia parziale senza sovrascrivere quelle preesistenti.

14.3 Aprire e salvare un diagramma ladder

Finora si è vista la procedura per la creazione di un diagramma ladder; se si desidera ripartire dall'inizio premere il pulsante della barra degli strumenti  corrispondente alla voce di menu 'File\Nuovo file'.

Se invece si vogliono salvare diagramma e impostazioni premere  o 'File\Salva file' da menu. Quando si salva per la prima volta si aprirà una finestra per assegnare un nome al file, l'estensione è sempre 'plp'.

Infine,  ('File\Carica file *.plp') apre una finestra da cui selezionare e aprire un file salvato in precedenza.

14.4 Compilazione

Per tradurre in pratica la logica rappresentata dal diagramma ladder occorre eseguire una compilazione e caricare il file così ottenuto nella memoria del PLC. Il tasto  della barra degli strumenti lancia il processo di compilazione; il risultato sarà mostrato nella barra di stato del programma, sotto la finestra dell'editor (cfr. Figura 15.8).

Dopo la compilazione è possibile trasferire il programma al PLC: controllare il collegamento fisico e la porta seriale (voce “Porta comunicazione” del menu), assicurarsi che il dispositivo connesso sia alimentato e corrisponda a quello selezionato per il programma (nella finestra di stato, Figura 14.12), infine premere il tasto  .

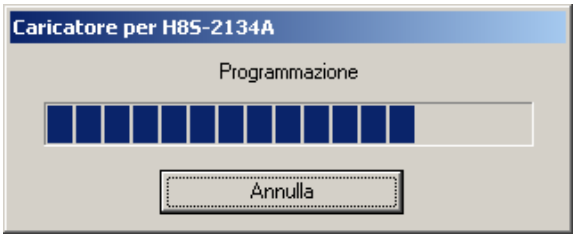


Figura 14.11: programmazione del PLC, l’intera operazione richiederà pochi secondi.

14.5 Verifica del funzionamento hardware e software

Prima di mettere in funzione l’impianto, è buona norma provare il programma del PLC attraverso una simulazione. PLProg dispone in particolare di tre strumenti attraverso i quali analizzare il comportamento di un dispositivo e del programma in esso caricato: debug, monitor delle aree di memoria e test PLC.

14.5.1 Debug

La modalità di funzionamento “Debug” consente di testare il programma caricato nel PLC visualizzando nel diagramma ladder lo stato dei contatti e delle bobine.

Per eseguire il debug è necessario che al PC sia collegato un dispositivo dello stesso tipo specificato nel programma e che la comunicazione sia attiva. Le informazioni relative a queste condizioni si trovano nella finestra di stato (cfr. par. 15.4).

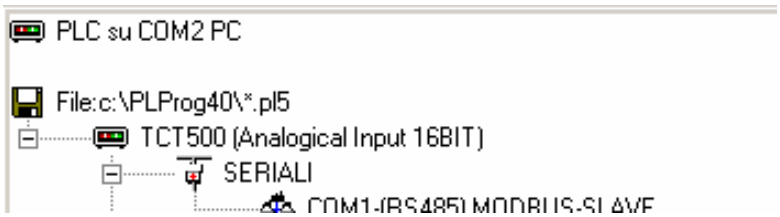


Figura 14.12: l’utente ha selezionato il PLC TCT500 ma non è ancora stata stabilita una connessione.

Figura 14.13: con un click sull'area evidenziata il programma rileva il PLC. In questa situazione è possibile procedere al debug.

Ovviamente, affinché la simulazione sia significativa, occorre che il diagramma ladder visualizzato corrisponda al programma caricato nella memoria del PLC.

Premendo  si entra in modalità debug; questo tasto apparirà disabilitato se si stanno monitorando le aree di memoria del PLC (cfr. Figura 14.17) perché non è possibile usare contemporaneamente queste due funzionalità.

Da questo momento, i contatti e le bobine attive saranno evidenziate con un riquadro blu:

Figura 14.14: bobina attiva.

inoltre, i contatti con condizione IF esporranno i valori delle espressioni confrontate.

Figura 14.15: lo stesso contatto visto in modalità normale e in debug quando la condizione è verificata.

La modalità debug blocca gran parte delle funzionalità di PLProg; per uscire premere nuovamente il tasto  .

14.5.2 Monitor delle aree di memoria

In alcuni casi può essere utile visualizzare dinamicamente il contenuto della memoria del PLC: la finestra rappresentata in Figura 14.16 e Figura 14.17 raccoglie e organizza in tabelle le

diverse aree. Un doppio click sulla griglia inizia e sospende la lettura.
La tabella “USER” è inizialmente vuota, in essa l’utente può inserire un riferimento alle locazioni di memoria che gli interessa monitorare contemporaneamente.

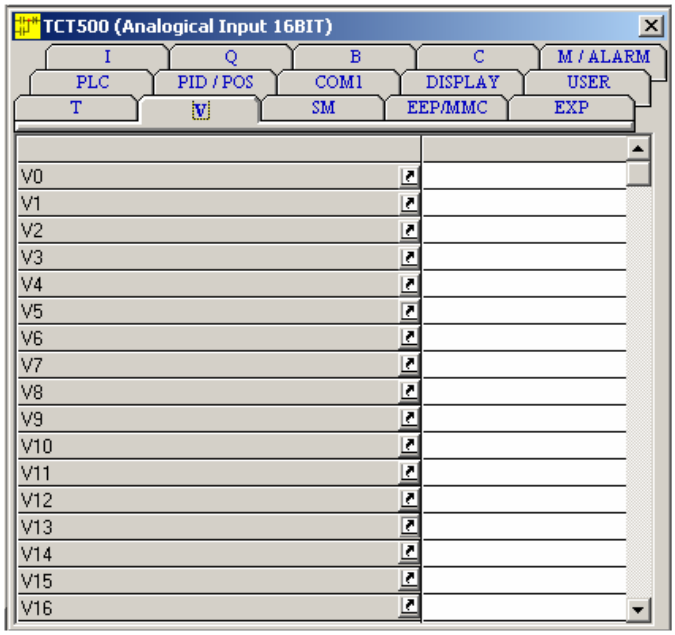


Figura 14.16: con un doppio click sulla griglia si inizia la lettura.

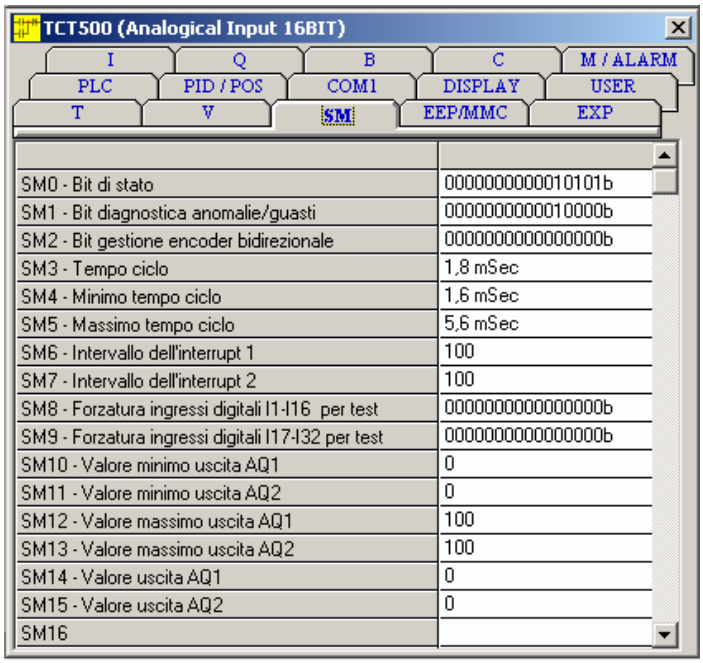


Figura 14.17: comunicazione Modbus in corso, con un doppio click sulla griglia si interrompe la lettura.

Quando la comunicazione con il PLC non è attiva (situazione di Figura 14.16), l'utente può inserire una variabile nella tabella "USER" o forzarne il valore: con un click sul tasto  si apre la finestra di Figura 14.18 tramite la quale è possibile compiere queste operazioni.

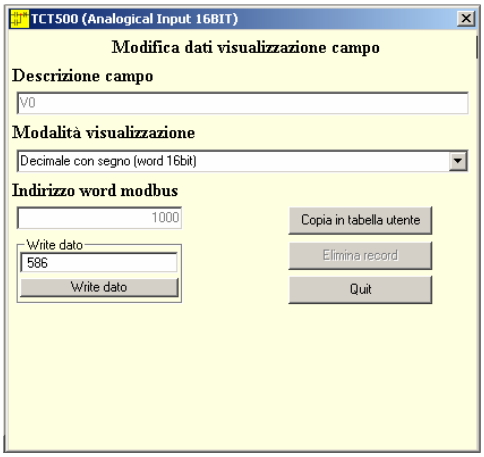


Figura 14.18: "Descrizione campo" ed "Elimina record" sono modificabili solo nella tabella "USER", "Write dato" non è disponibile per le aree della tabella "PLC".

14.5.3 Test PLC

Questo strumento è in grado di testare le funzionalità del PLC indipendentemente dal particolare programma caricato.

Premendo il tasto  della barra degli strumenti, si apre la finestra di Figura 14.19 (per tutti i PLC esclusi i terminali TD320 e TD240) grazie alla quale è possibile verificare per via grafica il corretto funzionamento della comunicazione, forzare ingressi e uscite e monitorare le variabili digitali e analogiche.

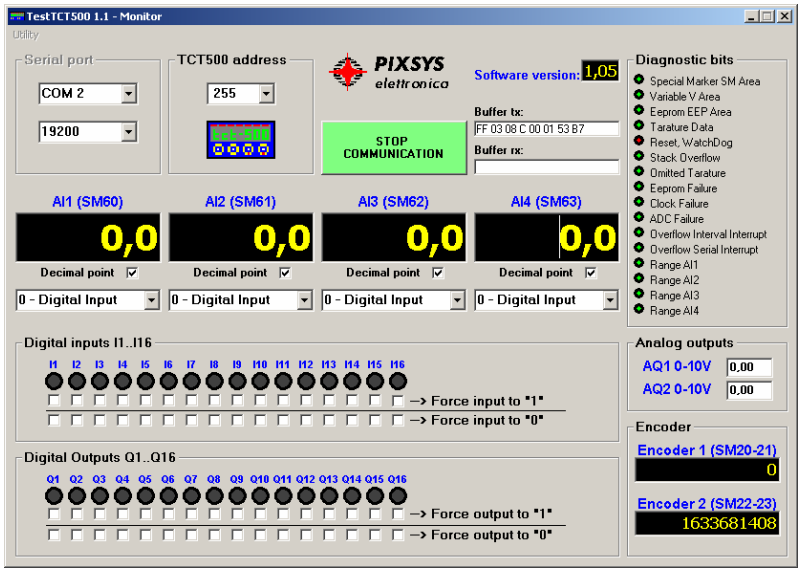


Figura 14.19: programma di test per il PLC, in questo caso TCT500.

14.6 Stampa

Dato il grande numero di informazioni legate a un programma ladder, la procedura (lanciata dal tasto ) è sempre preceduta dalla selezione di ciò che si vuole stampare.

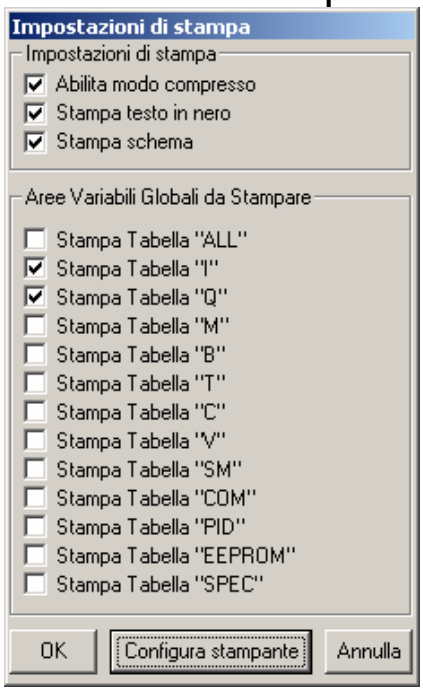


Figura 14.20: impostazioni di stampa.

“Abilita modo compresso”: stampa il diagramma ladder eliminando tutte le righe vuote.

“Stampa testo in nero”: anche le stringhe non nere del diagramma ladder (commenti, formule…) saranno stampate in nero.

“Stampa schema”: è possibile non stampare il diagramma ladder disattivando questa opzione.

“Area variabili globali da stampare” permette, attraverso la selezione dei suoi campi, di stampare una tabella con l’elenco di tutte le aree di memoria utilizzate nel programma, nello stile della finestra “Tabella globale variabili” (cfr. para. 15.5).

15 Guida al programma

L'obiettivo dei capitoli precedenti è introdurre alle principali funzionalità di PLProg, per questo finora sono state escluse dalla trattazione le funzioni avanzate o accessorie.
Per completare la descrizione del software, in questo capitolo saranno presi in esame tutte le finestre, le voci di menu e i pulsanti del programma.

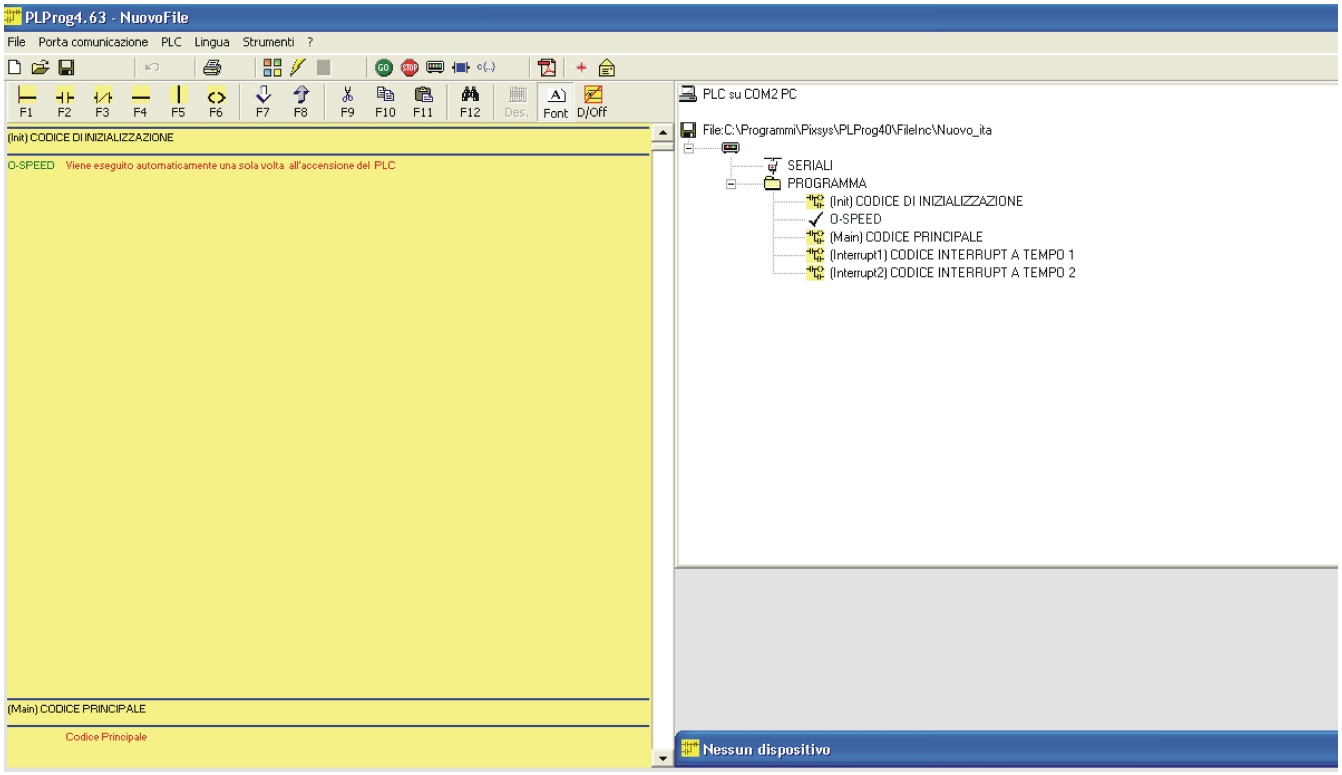


Figura 15.1: la schermata iniziale di PLProg

15.1 Menu principale

Raccoglie gli strumenti necessari al funzionamento del programma.

Figura 15.2: particolare del menù principale.

Figura 15.3: la barra degli strumenti, alcuni pulsanti replicano una funzione del menu principale.

15.1.1 File

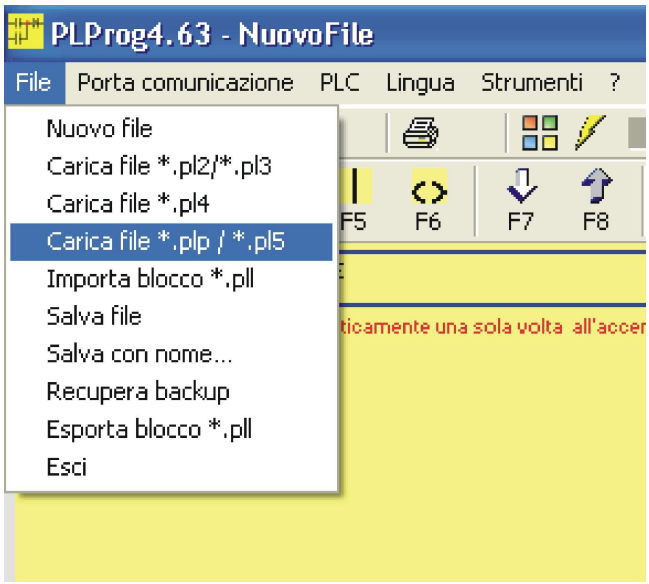


Figura 15.4: Voce “File” del menù principale

La voce “File” presenta tutte le possibili operazioni di apertura e salvataggio di interi programmi o di parte di essi. I pulsanti della barra degli strumenti:

Figura 15.5: Pulsanti della barra degli strumenti

fanno riferimento rispettivamente alle voci “Nuovo file”, “Carica file *.plp” e “Salva file”.

15.1.1.1 Nuovo file

Prepara l’ambiente per la stesura di un nuovo diagramma ladder, in particolare apre una finestra che permette di impostare il PLC nel quale si intende caricare il programma.

15.1.1.2 Carica file *.pl2/pl3

Permette di accedere a programmi creati con le precedenti versioni di PLProg.

15.1.1.3 Carica file *.pl4

Come punto precedente.

15.1.1.4 Carica file *.plp / *.pl5

I file con estensione ‘plp’ sono quelli creati con l’attuale versione di PLProg. Rispetto ai precedenti, questi formati memorizzano non solo il diagramma ladder ma anche altre utili informazioni come gli eventuali nomi associati dall’utente alle aree di memoria (cfr. par. 15.5), le impostazioni del PLC (Figura 14.1) e la tabella “USER” (cfr. par. 14.5.2).

15.1.1.5 Importa blocco *.pll

E’ possibile salvare alcune righe (un “blocco”) di un diagramma ladder in un file con estensione ‘pll’ rendendole così velocemente accessibili e riutilizzabili da altri progetti.
Per importare un blocco precedentemente memorizzato (cfr.15.1.1.9), la sequenza di operazioni è la seguente:

- comando dal menù “Importa blocco *.pll”;
- selezionare dall’elenco il file ‘pll’ desiderato, a questo punto il puntatore del mouse cambia aspetto:

Figura 15.6

- click sull’area del diagramma in cui si vuole inserire il blocco; se non ci fosse lo spazio sufficiente questa operazione non sovrascriverà righe di programma ma si limiterà a riempire quello disponibile.

15.1.1.6 Salva file

Salva il programma in un file ‘.plp’; i file aperti come *.pl2, *.pl3, *.pl4 e *.pl5 saranno automaticamente convertiti a questo formato. Oltre al diagramma ladder, sono memorizzate le impostazioni del

PLC e la tabella ‘USER’ della finestra monitor delle aree di memoria.

15.1.1.7 Salva con nome…

Come punto precedente con la possibilità di assegnare o cambiare nome al file.

15.1.1.8 Recupera backup

PLProg crea automaticamente delle copie di backup del file su cui si sta lavorando. La frequenza di questa operazione può essere impostata nel menu “Opzioni salvataggio automatico” (cfr.15.1.5.3).
Il comando “Recupera backup” consente di caricare questi file altrimenti nascosti.

15.1.1.9 Esporta blocco *.pll

Salva una parte di diagramma ladder in un file, ha effetto solo se in precedenza si è selezionato un blocco (cfr. procedura per la copia di un blocco in 14.2.1).

15.1.1.10 Esci

Chiude l’applicazione.

15.1.2 Porta comunicazione

15.1.2.1 Com

Apre una finestra da cui selezionare, coerentemente con i collegamenti fisici, la porta di comunicazione tra PC e PLC.
N.B.: il Baudrate della comunicazione tra PC e PLC è impostato automaticamente.

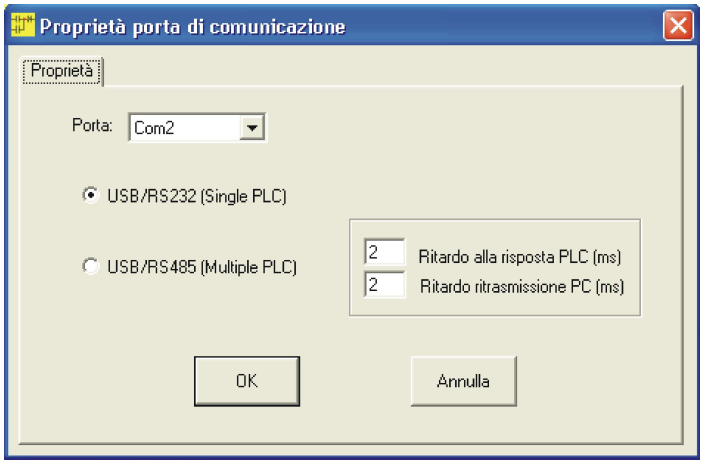


Figura 15.7: Dal menu 'Porta comunicazione' si accede alle Proprietà

15.1.3 PLC

Operazioni di gestione del PLC connesso al PC.
Le voci di menù sono replicate sui pulsanti:

per "RUN" e "STOP", sui pulsanti:

per "Compila tutto" e "Invia al PLC".

15.1.3.1 RUN

Riattiva l'esecuzione del programma da parte del PLC se questo era bloccato.

15.1.3.2 STOP

Blocca l'esecuzione del programma da parte del PLC.

15.1.3.3 Compila tutto

A partire dal diagramma ladder, crea il file binario da scaricare nella memoria del PLC. L'esito della compilazione è riassunto nella barra di stato, in basso a sinistra.

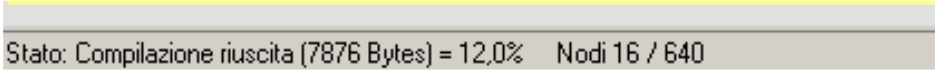


Figura 15.8: barra di stato di PLProg, si trova sotto la finestra dell'editor.

15.1.3.4 Invia al PLC

Attraverso il collegamento seriale programma il PLC. Questa operazione è consentita solo in seguito a una compilazione.

15.1.3.5 TipoPLC

Apre la finestra:

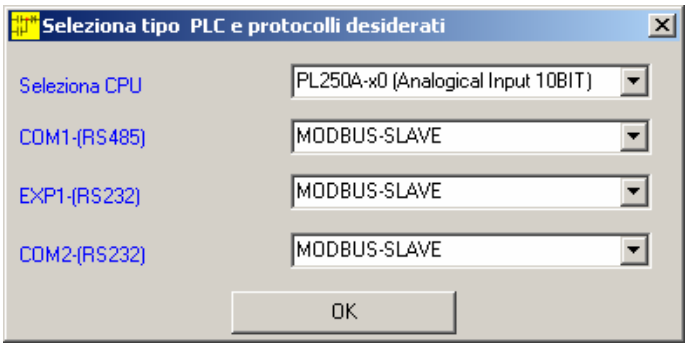


Figura 15.9: configura il tipo di dispositivo e i protocolli delle seriali. COM2 è la porta di programmazione del PLC ed è quindi fissa a MODBUS-SLAVE.

15.1.3.6 Aggiorna orologio PLC

Sincronizza l'orologio del PLC con quello del PC.

15.1.3.7 Crea memory card

Oltre che attraverso PLProg, i programmi possono essere caricati nella memoria dei PLC tramite una memory card. Questo comando ha un funzionamento simile a quello di "Invia al PLC" (15.1.3.4); prima di iniziare la procedura PLProg visualizzerà una foto che illustra il corretto posizionamento della memory card.
Per programmare un PLC, togliere l'alimentazione, inserire la card e riaccendere.

15.1.3.8 Manutenzione PLC

Insieme di operazioni avanzate per la gestione del PLC.
Le prime due, "Aggiorna versione firmware" e "Aggiorna Versione BOOT", non sono in genere di competenza dell'utente finale anche perché PLProg provvederà automaticamente, se fosse necessario, all'aggiornamento del firmware.
"DataSave (Read/Write–VW-EEPROM)" apre la finestra:

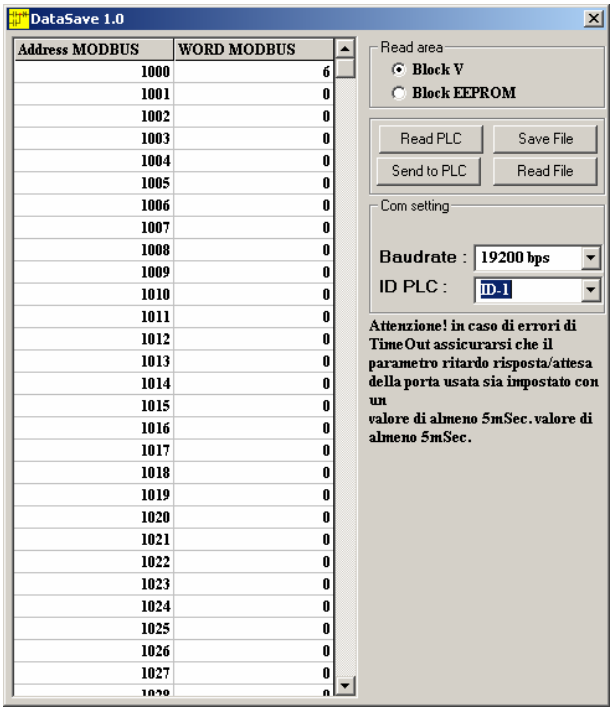


Figura 15.10: finestra per la lettura e scrittura delle aree VW e EEPROM.

Attraverso questo strumento è possibile salvare l'intero contenuto delle aree di memoria VW ed EEPROM del PLC in un file o, viceversa, copiarne il contenuto da un file. Un esempio di applicazione è la clonazione di dispositivi.

15.1.4 Lingua

Le due opzioni sono Italiano e Inglese.

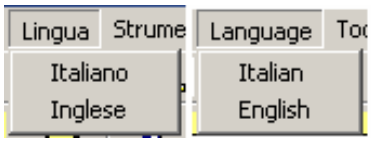


Figura 15.11: il menu Lingua nei due casi.

15.1.5 Strumenti

Attraverso i comandi del menu Strumenti, l'utente può modificare il funzionamento e configurare alcune caratteristiche dell'ambiente di sviluppo.

15.1.5.1 Opzioni Editor

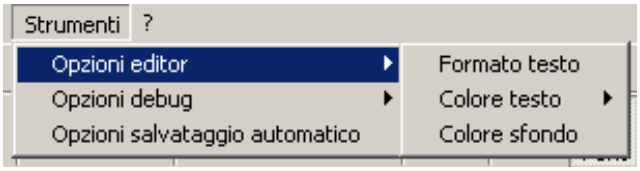


Figura 15.12: Opzioni editor

E' possibile impostare secondo le proprie preferenze

- **Formato testo** (tipo e dimensione del carattere);
- **Colore testo**, per tutti i diversi tipi di scritte dell'editor (Funzioni, Descrizioni, Titolo area codice, segnalibro);
- **Colore sfondo** (questa opzione non è attiva per i sistemi operativi Win98).

15.1.5.2 Opzioni debug

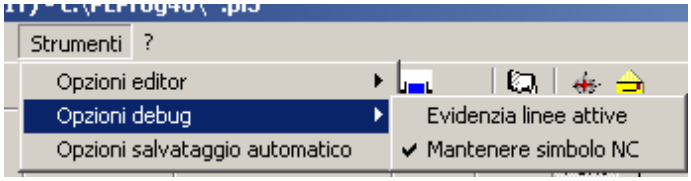


Figura 15.13: Opzioni debug

- Evidenzia linee attive
  - Se si sceglie questa opzione saranno evidenziati anche i collegamenti che seguono contatti e bobine attivi; le figure (Figura 15.14, Figura 15.15) mostrano la stessa porzione di schema nei due casi.

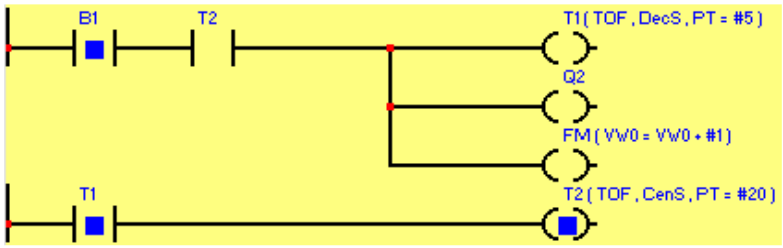


Figura 15.14: opzione "Evidenzia linee attive" non selezionata.

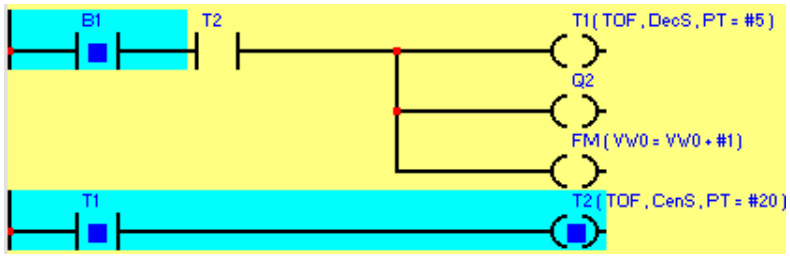


Figura 15.15: opzione “Evidenzia linee attive” selezionata.

- Mantenere simbolo NC
  - La sigla NC indica contatti logici considerati attivi quando il corrispondente contatto fisico è aperto. In fase di debug si può scegliere se visualizzarli con lo stesso simbolo dei contatti NA o, viceversa, mantenere il simbolo utilizzato in fase di progettazione.

Figura 15.16: contatto NC attivo; con opzione “Mantenere simbolo NC” non selezionata a sinistra, selezionata a destra.

15.1.5.3 Opzioni salvataggio automatico

Impostazione della frequenza con cui sono salvati automaticamente i file di backup.

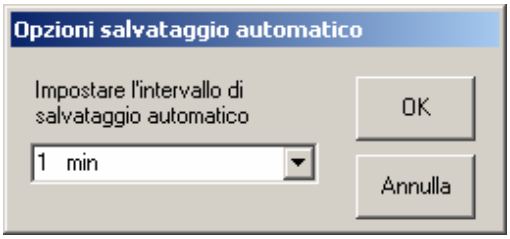


Figura 15.17: finestra per l’impostazione del salvataggio automatico.

15.1.6 ? (Guida)

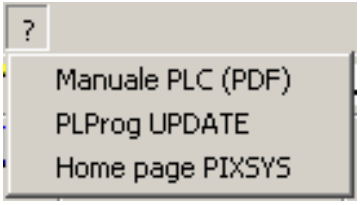


Figura 15.18: Menu “?”

15.1.6.1 Manuale PLC (PDF)

Apre il manuale del PLC o del dispositivo; se non ne è ancora stato selezionato uno non ha alcun effetto. Qualora mancasse un programma adatto a visualizzare documenti in formato pdf, PLProg provvederà automaticamente ad installarlo.
Questo comando è equivalente al pulsante della barra degli strumenti:  .

15.1.6.2 PLProg UPDATE

Se si dispone di una connessione internet permette di scaricare il programma di installazione di PLProg più recente.

15.1.6.3 Home page PIXSYS

Connette al sito http://www.pixsys.net.

15.2 Barra degli strumenti

In questo capitolo sono presentati i pulsanti della barra degli strumenti, in particolare sarà chiarito il funzionamento di quelli che non corrispondono a voci del menu principale.

 Nuovo file (cfr.15.1.1.1)

 Apri file (cfr.15.1.1.4)

 Salva file (cfr.15.1.1.6)

 E’ la classica funzione “Undo”: annulla le ultime modifiche apportate al diagramma ladder.

 Apre la finestra di stampa di Figura 14.20.

 Compila il programma(cfr.15.1.3.3).

 Invia il programma al PLC (cfr.15.1.3.4).

 Apre la tabella delle variabili (cfr.15.5)

 Start PLC (cfr.15.1.3.1).

 Stop PLC (cfr.15.1.3.2).

 Avvia programma di test per il PLC collegato.

 Entra in modalità debug.

 Apre il manuale relativo al PLC selezionato (cfr.15.1.6.1).

 Connette alla home page Pixsys (cfr.15.1.6.3).

 Per suggerimenti e chiarimenti via e-mail sul funzionamento di PLProg.

15.3 Barra degli strumenti dell'editor

Sono qui descritte brevemente le funzioni associate ai tasti della toolbar (Figura 15.19).

Figura 15.19:la toolbar dell'editor

F1 : inserisce nello schema un nuovo simbolo di inizio riga.
F2 : nuovo contatto normalmente aperto.
F3 : nuovo contatto normalmente chiuso.
F4 : collegamento orizzontale, se la casella selezionata è un collegamento verticale, traccia una curva.

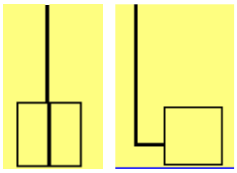


Figura 15.20: effetto della funzione F4 su un collegamento verticale

F5 : collegamento verticale, se la casella selezionata è un collegamento orizzontale, traccia un nodo.

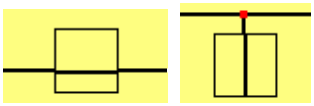


Figura 15.21: effetto della funzione F5 su un collegamento orizzontale. Questo vale solo dove il puntatore del mouse ha l'aspetto di Figura 14.3.

F6 : nuova bobina.
F7 : a partire dalla riga della casella selezionata, sposta in basso di una riga tutto il diagramma.
F8 : a partire dalla riga della casella selezionata, sposta in alto di una riga tutto il diagramma.
F9 : taglia una casella o un blocco di righe e li mantiene in memoria per poterli incollare in un'altra parte del programma.
F10: copia una casella o un blocco di righe (14.2.1).
F11: incolla l'ultimo blocco di righe copiato.
F12: apre la finestra di ricerca

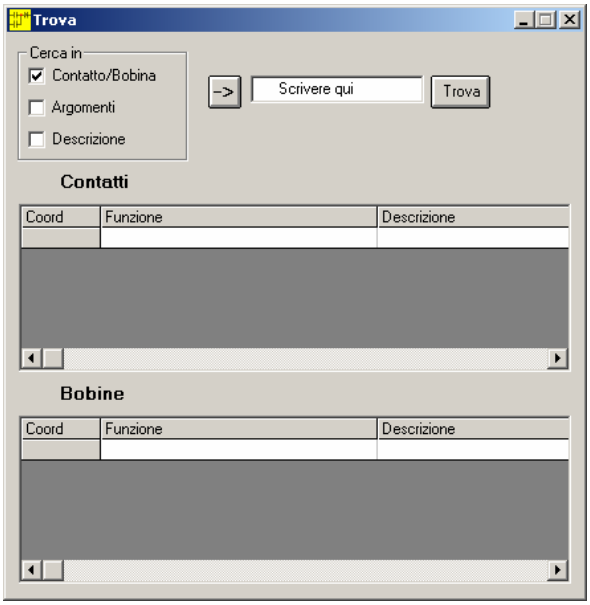


Figura 15.22: scrivere la stringa da ricercare nel campo testo. La prima colonna della griglia conterrà sempre le coordinate riga-colonna del risultato.

L'utente può scegliere attraverso le tre opzioni del campo "Cerca in" se cercare una stringa tra i nomi e le funzioni di contatti e bobine, tra gli argomenti di funzione o tra titoli e descrizioni. Queste opzioni non sono mutuamente esclusive.
La ricerca non distingue tra caratteri maiuscoli e minuscoli.
Se si seleziona sull'editor un contatto o una bobina del diagramma ladder (click del mouse), si può immettere automaticamente nel campo testo il nome della funzione associata usando il tasto freccia  .
La ricerca con l'opzione Contatti/Bobine funziona solo con nomi completi; ad esempio, per trovare il contatto I11, non basta inserire la stringa "i" o "i1" ma occorre scrivere "i11". Le opzioni Argomenti e Descrizione, invece, accettano qualsiasi tipo di stringa; ad esempio, cercando la stringa "sm", si potrebbero trovare **SMW0.1** e **SMW1.2** tra gli argomenti e "Tra**sm**issione" tra le descrizioni.

Des.: attiva/disattiva la modalità di visualizzazione descrittiva (Figura 15.27 e Figura 15.28).

Font: se attivo, seleziona il font personalizzato (cfr.15.1.5.1), altrimenti il font di default.

D/Off: nasconde le descrizioni associate a contatti e bobine.

15.4 Finestra di stato

Raggruppa tutte le informazioni relative al dispositivo collegato al PC e al file su cui si sta lavorando. La finestra è organizzata in modo che i dati possano essere mostrati per esteso o nascosti rispettivamente con un click sui segni '+' e '-' a fianco del titolo.

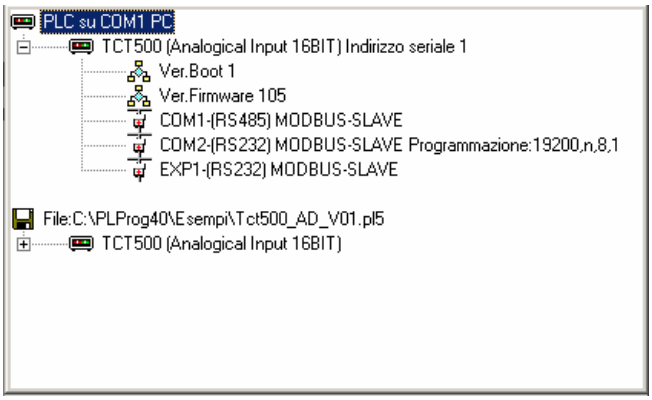


Figura 15.23: finestra di stato. Le impostazioni evidenziate sono relative al PLC collegato

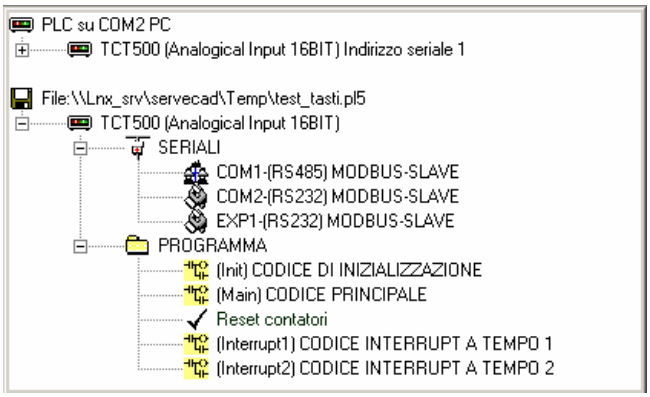


Figura 15.24: la finestra fa riferimento alla stessa situazione della figura precedente. In questo caso sono state messe in evidenza le impostazioni del programma.

A partire dall'alto si incontrano i seguenti campi.

- Le impostazioni del PLC fisicamente connesso al PC (Figura 15.23).
- Il PLC scelto dall'utente e le impostazioni per le seriali (Figura 15.24). In generale lo stesso diagramma ladder non può essere compilato e trasferito su due PLC diversi. Per cambiare PLC o l'impostazione delle seriali richiamare la

finestra di Figura 14.1 con un click su una riga di questo campo.

- L'indice del diagramma ladder: con un click su una riga di questo campo si visualizza la parte di programma che segue il titolo corrispondente. Come spiegato in seguito alla Figura 14.10, si possono inserire nuove voci, contrassegnate con il simbolo ✓ per distinguerle da quelle create automaticamente da PLProg (simbolo).

15.5 Tabella delle variabili

L'utente può decidere di assegnare ai flag e alle aree di memoria del programma un nome significativo per l'applicazione che sta sviluppando. Il tasto apre la finestra

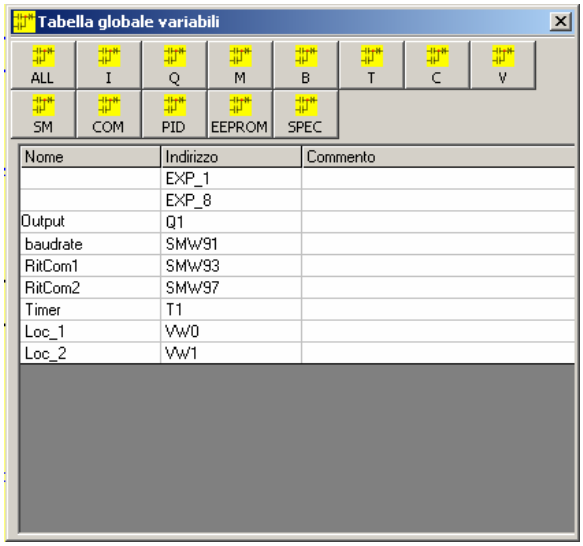


Figura 15.25: i campi Nome e Commento sono editabili, la griglia è sensibile al click del mouse, ai tasti freccia, Tab, Invio, Space e BackSpace.

I nomi assegnati alle "variabili" possono essere visualizzati anche sul diagramma ladder. Il tasto

Figura 15.26: tasto per l'attivazione/disattivazione della modalità descrittiva

permette di passare da un modo di visualizzazione all'altro. Le due figure che seguono fanno riferimento alla stessa porzione di diagramma; i nomi sono quelli assegnati in Figura 15.25.

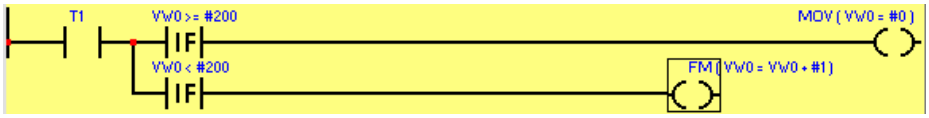


Figura 15.27: modalità di visualizzazione normale

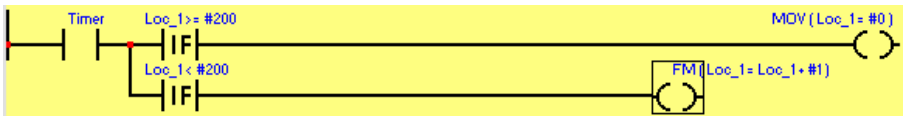


Figura 15.28: modalità di visualizzazione descrittiva

16 Esempi di programmazione Ladder

16.1 Autoritenuta

L'autoritenuta è uno degli schemi logici più usati nella programmazione ladder; si tratta di un meccanismo in grado di mantenere attiva un'uscita anche quando è cessato il segnale di attivazione.
In Figura 16.1 e Figura 16.2 è presentato un esempio di realizzazione.

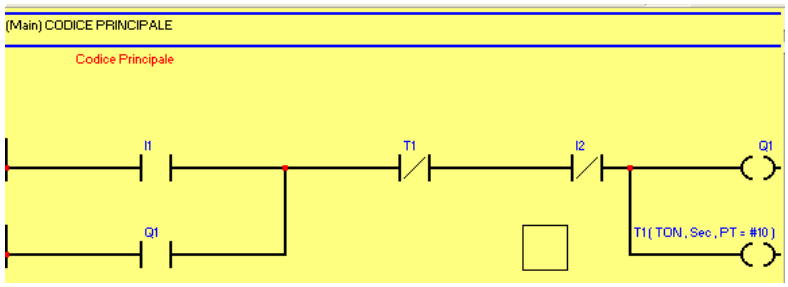


Figura 16.1: il contatto I1 è lo start, quando viene chiuso attiva Q1. Per diasattivare l'uscita occorre aprire il contatto normalmente chiuso I2 o attendere 10 secondi per l'apertura del contatto a timer.

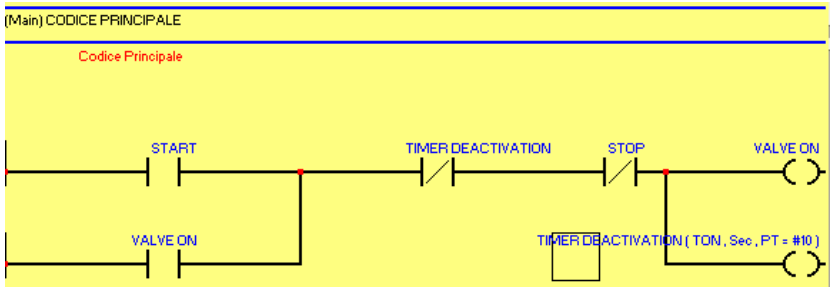


Figura 16.2: diagramma in modalità descrittiva (visualizzazione dei nomi assegnati dall'utente).

16.2 Regolazione di temperatura con uscita analogica

Nell'esempio che segue si realizzerà una regolazione di temperatura usando una PT100 come sensore e un dispositivo comandato in tensione come attuatore.
Il primo passo è l'impostazione dell'hardware, ovviamente a dispositivo spento, in accordo con le specifiche presenti nel manuale del PLC.

Successivamente, nel codice di inizializzazione, selezionare il particolare tipo di ingresso analogico mediante gli opportuni campi dell'area Special Marker: nel caso di una PT100 connessa fisicamente agli ingressi AI3 e AI4 occorre scrivere la costante #13 nelle word SMW42 e SMW43, per altre selezioni consultare il manuale del PLC.
Il SetPoint per il processo è preso da uno dei due Trimmer del PLC normalizzato tra 0 e 30℃.
L'altro Trimmer è un offset per l'ingresso, variabile tra –3 e +3℃.
Si supponga di voler regolare un'uscita continua 0-10V: in conformità con l'uscita della bobina che realizza la regolazione P.I.D. si normalizza l'uscita AQ1 tra 0 e 10000.
Infine, deve essere inizializzata l'azione del P.I.D, in questo caso con la sola componente proporzionale. La Figura 16.3 riporta tutte le configurazioni iniziali finora discusse.

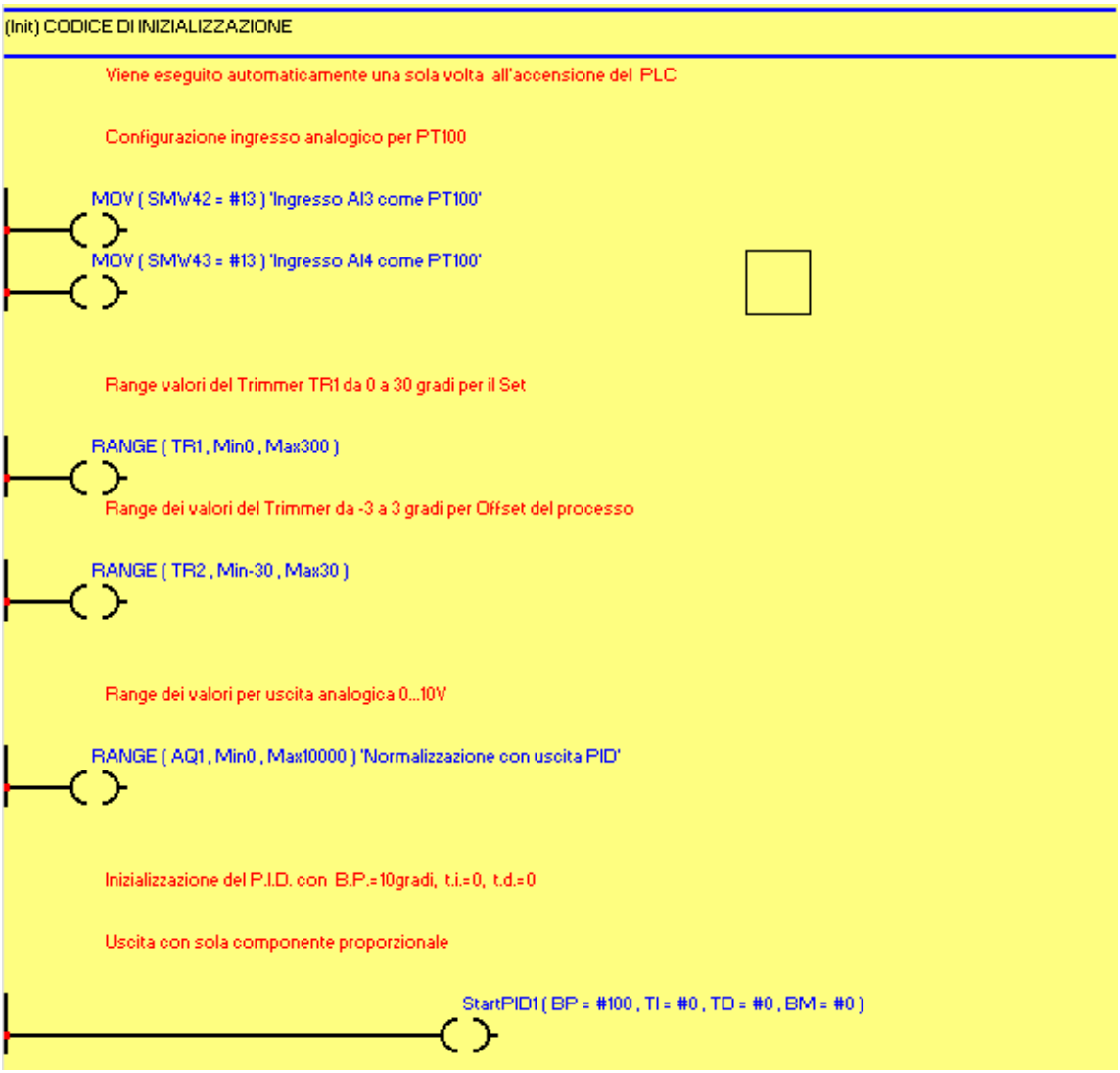


Figura 16.3: codice di inizializzazione per l'applicazione. Nota: i valori passati alla funzione RANGE applicata ai trimmer sono moltiplicati per 10 rispetto al valore in gradi che si vuole impostare. Questo è dovuto al fatto che, quando un ingresso analogico legge una temperatura, la restituisce moltiplicata per 10 per ottenere una precisione del decimo di grado.

Per questa applicazione sarà sufficiente calcolare l'azione PID una volta al secondo. Il valore così ottenuto è passato all'uscita analogica AQ1, disponibile tra i morsetti specificati nel manuale del PLC.
La Figura 16.4 riporta la parte Main del diagramma.

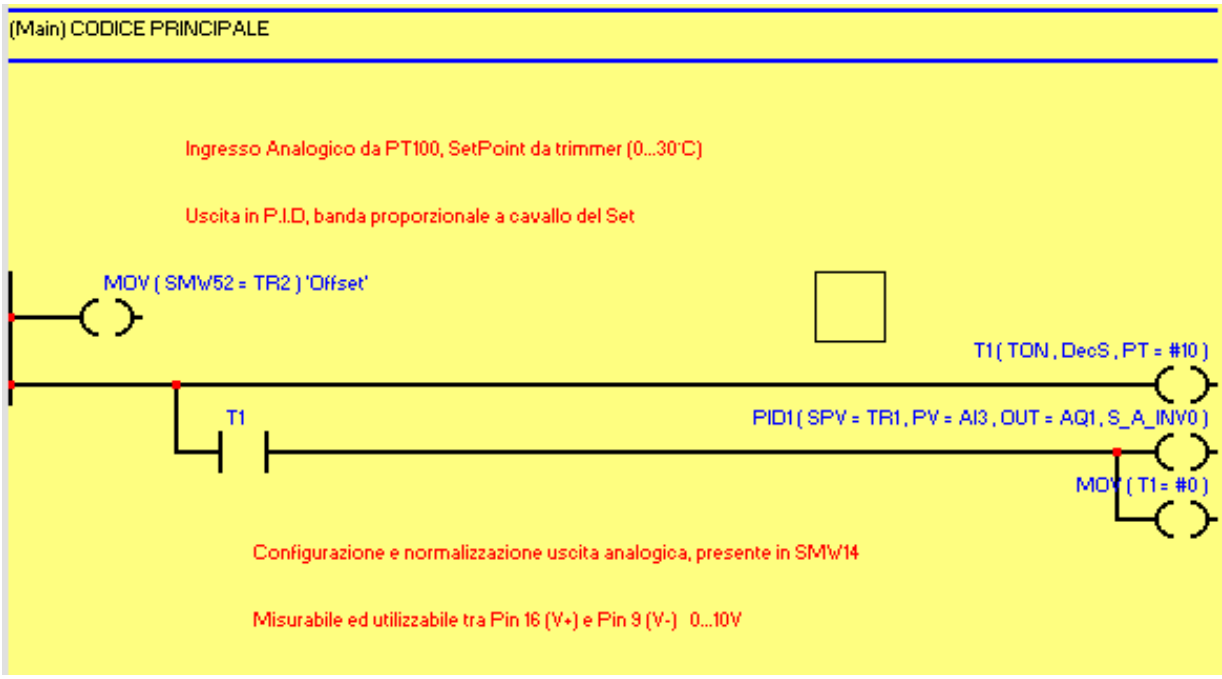


Figura 16.4: ad intervalli di un secondo il contatto T1 si attiva per un istante durante il quale la funzione PID1 imposta il valore dell'uscita analogica AQ1.

16.3 Regolazione di temperatura con uscita relè

L'applicazione dell'esempio precedente può essere leggermente modificata per funzionare anche nel caso in cui l'attuatore sia di tipo ON/OFF. Volendo ancora usare la funzione PID occorre modularne l'uscita come rapporto tra tempo di chiusura e di apertura di un relè in un ciclo fissato.

La parte di inizializzazione è la stessa di Figura 16.3 con la differenza che ora non è più necessario impostare il RANGE dell'uscita analogica.

Nel codice principale, il valore calcolato dalla funzione PID non va più passato direttamente all'uscita continua AQ1 ma deve essere salvato in una locazione di memoria per poter calcolare il rapporto tra tempo di accensione del relè e tempo di ciclo.

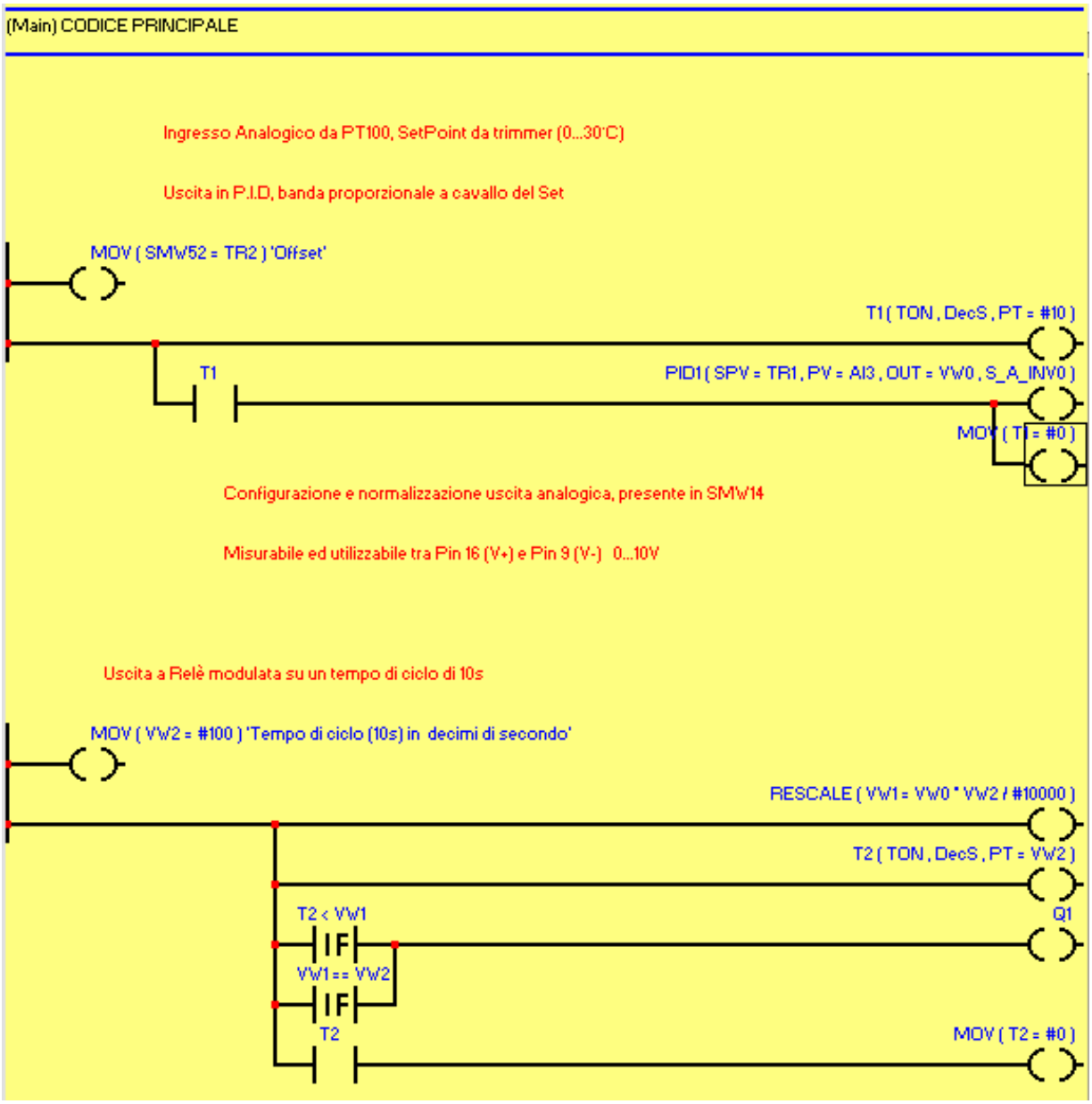


Figura 16.5: l'uscita del PID viene copiata nell'area di memoria VW0. VW1 rappresenta la percentuale rispetto al tempo di ciclo VW2 in cui il relè Q1 dovrà essere chiuso.

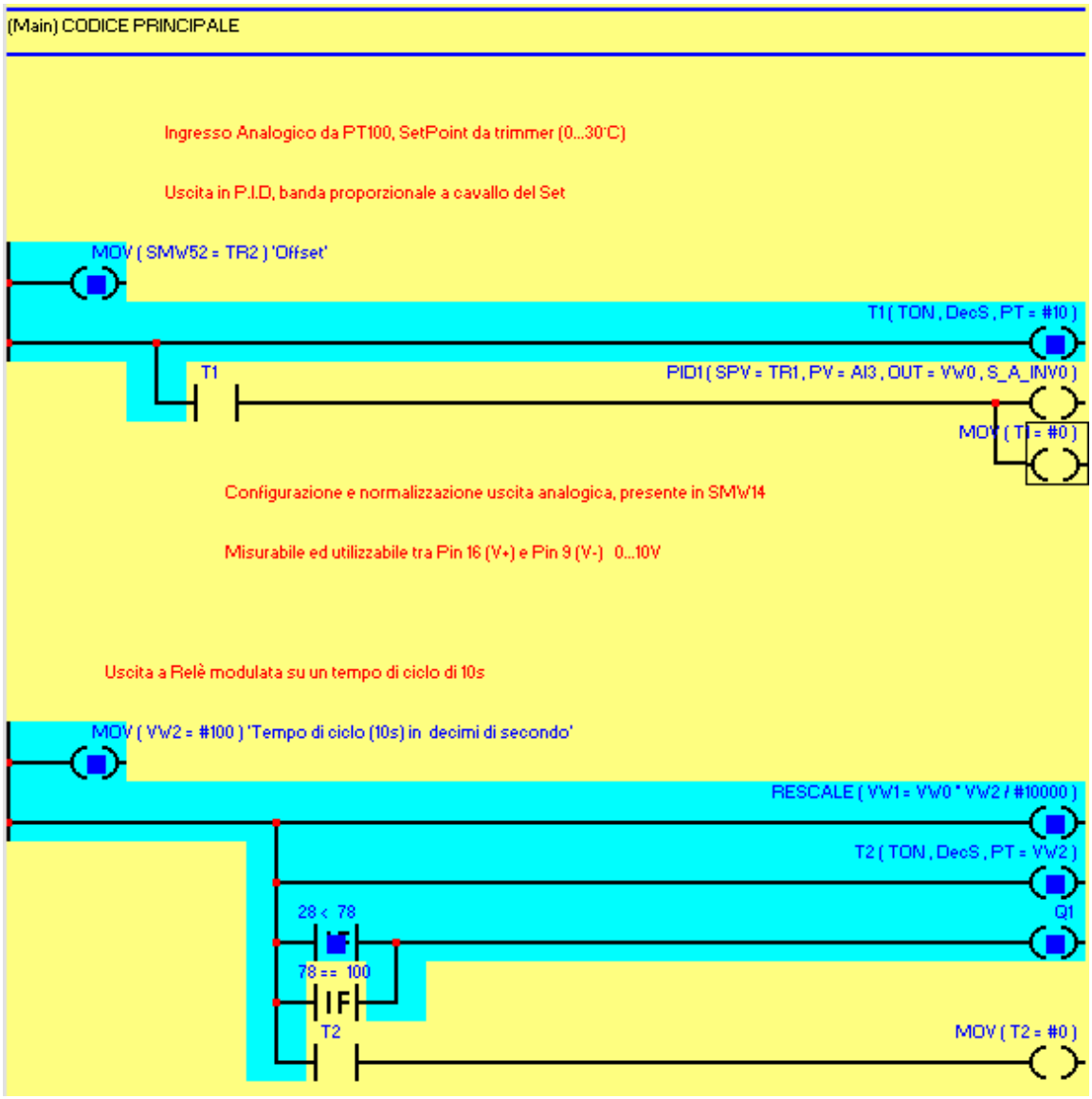


Figura 16.6: funzionamento in Run-time. L'uscita calcolata dal PID è 7800, quindi, in un tempo di ciclo di 10 secondi, Q1 risulta chiuso per 7,8 secondi e aperto per 2,2. In questa immagine, T2 vale 2.8s, quindi Q1 è chiuso.

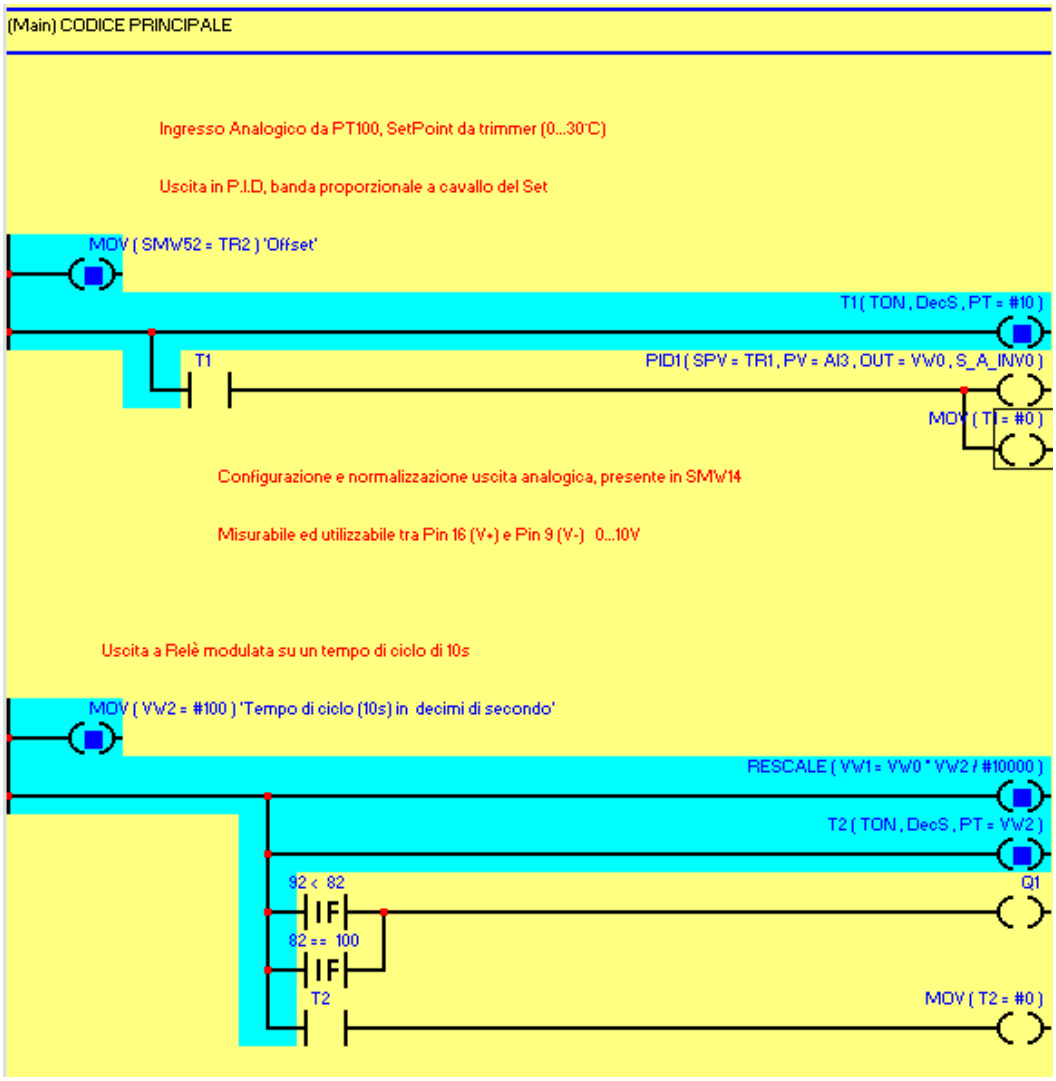


Figura 16.7: funzionamento in Run-time. in questo caso T2 vale 9.2s, quindi Q1 è aperto.

17 TdDesigner

17.1 Introduzione

TdDesigner è l'ambiente di sviluppo per la configurazione della grafica dei terminali Pixsys TD320 e TD240. Questi dispositivi racchiudono la doppia funzionalità di PLC e HMI (Human Machine Interface, Interfaccia Uomo Macchina) quindi, in generale, lo sviluppo di un'applicazione richiede l'utilizzo contemporaneo di PlProg e TdDesigner.

L'approccio WYSIWYG (What You See Is What You Get, "ottieni quello che vedi") di TdDesigner rende particolarmente semplice la creazione e la gestione di progetti anche complessi: una volta trasmessi al terminale, gli oggetti grafici e le schermate appariranno esattamente come sullo schermo del PC.

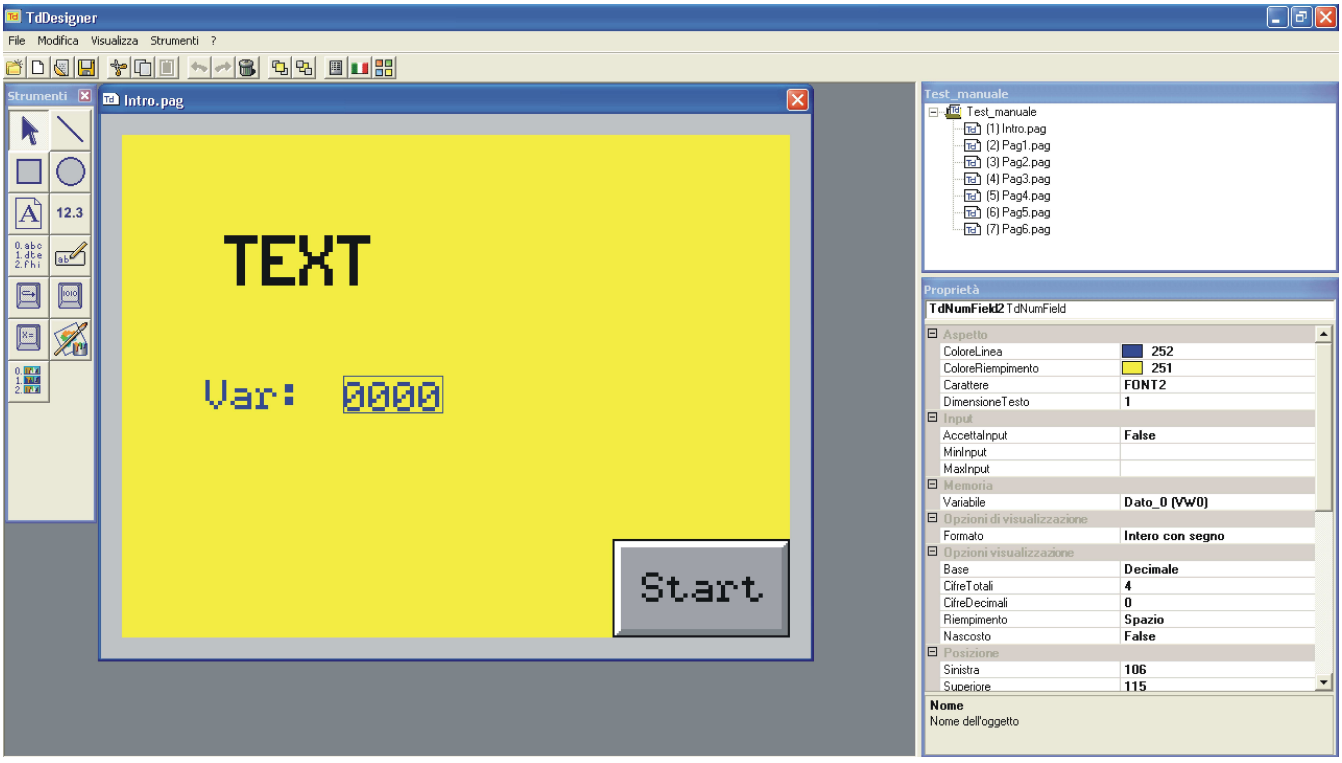


Figura 17.1: una tipica schermata di TdDesigner. Sono visibilli la Casella Strumenti (a sinistra), una pagina (al centro), la Finestra di Gestione Progetto (in alto a destra) e la Finestra Proprietà (in basso a destra).

17.2 Definizioni

In seguito si indicherà con il nome “utente” l’utilizzatore dell’ambiente di sviluppo. “Operatore” sarà invece chi si troverà di fronte al terminale grafico configurato con TdDesigner.
Le voci contrassegnate con (*) sono quelle non ancora completamente supportate al momento della pubblicazione di questo manuale.

17.2.1.1 Oggetto grafico

Unità grafica contenuta in una schermata del terminale, configurabile ed eventualmente in grado di reagire all’azione dell’operatore. Tipici oggetti grafici sono i pulsanti, le forme geometriche e i campi variabili.

17.2.1.2 Variabile

Area di memoria del terminale accessibile in fase di progettazione tramite un nome assegnato dall’utente.

17.2.1.3 Pagina

L’intero contenuto di una schermata del terminale grafico, comprende in generale un’immagine o un colore di sfondo e una serie di oggetti grafici.

17.2.1.4 Progetto

Insieme di pagine e di variabili che descrivono il funzionamento del terminale. Ogni progetto sarà salvato in un insieme di file, uno con estensione “pag” per ciascuna pagina e uno con estensione “tdproj” che conterrà i riferimenti alle pagine e alle variabili.

17.2.1.5 Compilazione

Operazione automatica attraverso la quale TdDesigner traduce un progetto in un file di configurazione da inviare al terminale tramite PlProg.

17.3 Menu

Il menu di TdDesigner è simile a quello della maggior parte dei software commerciali.

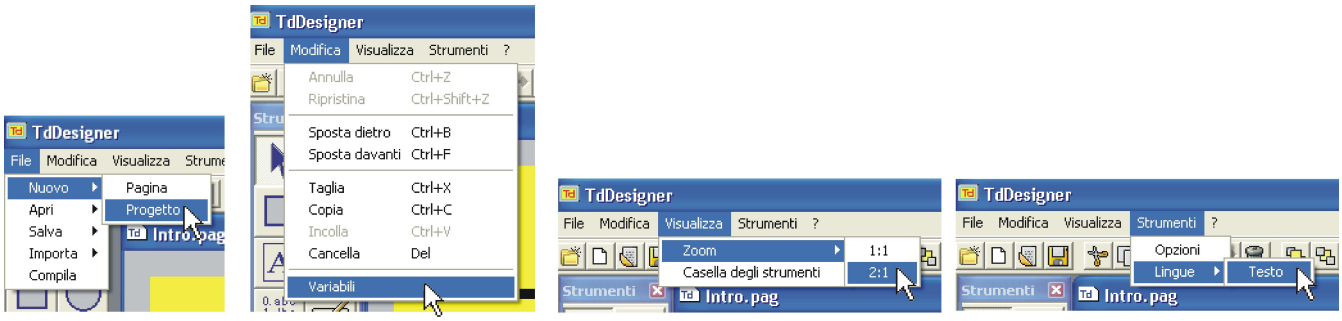


Figura 17.2: voci di menu di TdDesigner.

Alcune funzioni saranno riprese e approfondite nei prossimi paragrafi, per ora si osservi che sono disponibili le consuete funzioni di

- creazione, apertura e salvataggio file;
- modifica di un file aperto: “Annulla” e “Ripristina” modifiche, “Taglia-Copia-Incolla” di oggetti singoli, gruppi di oggetti e intere pagine, ognuna con la relativa sequenza rapida da tastiera.
- Possibilità di importare pagine da altri progetti TdDesigner.
- impostazione “Opzioni” di ambiente: dimensione e visualizzazione griglia, e gestione lingue.
- Compilazione del progetto

Le più importanti voci del menu sono inoltre disponibili nella toolbar:

 **Nuovo Progetto:** Crea un nuovo progetto TdDesigner.

 **Nuova Pagina:** Crea una nuova pagina nel progetto.

 **Apri Progetto:** Apre un progetto già esistente.

 **Salva:** Salva il progetto aperto con lo stesso nome.

 **Taglia:** Taglia l'oggetto selezionato / gli oggetti selezionati. Attiva solo se è stato selezionato almeno un oggetto (Ctrl+X).

 **Copia:** Copia l'oggetto selezionato / gli oggetti selezionati. Attiva solo se è stato selezionato almeno un oggetto (Ctrl+C).

 **Incolla:** Incolla l'oggetto selezionato / gli oggetti selezionati. Attiva solo se è stato copiato almeno un oggetto (Ctrl+V).

 **Annulla:** Annulla l'ultima operazione effettuata (Ctrl+Z).

 **Ripristina:** Ripristina l'ultima operazione annullata (Ctrl+Shift+Z).

 **Cancella:** Cancella l'oggetto selezionato / gli oggetti selezionati. Attiva solo se è stato selezionato almeno un oggetto.

 **Sposta davanti:** Modifica la posizione reciproca di oggetti sovrapposti. Sposta davanti l'oggetto selezionato (Ctrl+F).

 **Sposta dietro:** Modifica la posizione reciproca di oggetti sovrapposti. Sposta dietro l'oggetto selezionato (Ctrl+B).

 **Modifica variabili:** Apre l'editor di impostazione e modifica variabili.

 **Traduzioni:** Apre l'editor **TdLinguist** di selezione ed impostazione lingue.

 **Compila:** Compila il progetto aperto (vedi par. 17.7)

17.4 Finestra Proprietà

Tutte le proprietà delle pagine e degli oggetti grafici di un progetto sono visibili e modificabili nella Finestra Proprietà (Figura 17.1 e Figura 17.3). In particolare l'ultimo oggetto selezionato (con un click) dall'utente esporrà le sue proprietà in questa finestra. Appena sotto la griglia delle proprietà un breve messaggio presenta una sintetica spiegazione della voce selezionata.

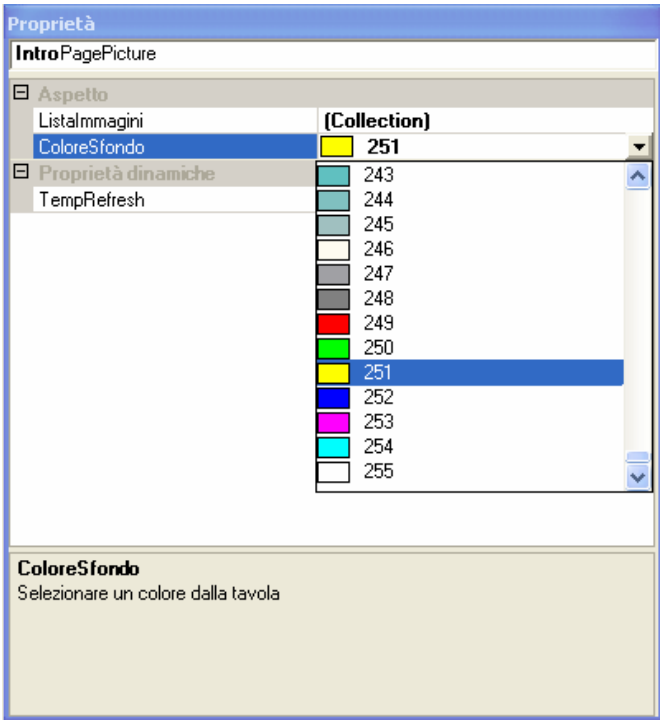


Figura 17.3: Finestra Proprietà. L'oggetto e la proprietà selezionati sono rispettivamente una pagina e il suo colore di sfondo.

Nella Figura 17.3 il nome dell'oggetto **pagina** è **Intro**. PagePicture indica il tipo **pagina**.

TdDesigner assegna un'etichetta progressiva di default agli oggetti. In questo esempio è **TdText0**, mentre TdText indica il tipo di dato.

Figura 17.4: Finestra proprietà. L'oggetto selezionato è il testo "Text" della pagina Intro

Nel campo **Progettazione** l'utente può assegnare un nome diverso da quello di default, come riportato in Figura 17.5. Ora il campo testo "Text" si chiama **Testo Libero**.

Figura 17.5: Finestra di Proprietà. Cambio del nome dell'oggetto.

17.5 Creazione di un nuovo progetto

Selezionare la voce di menu "**File\Nuovo\Progetto**", come mostrato in Figura 17.2 a sinistra. Selezionare il terminale che si intende programmare.

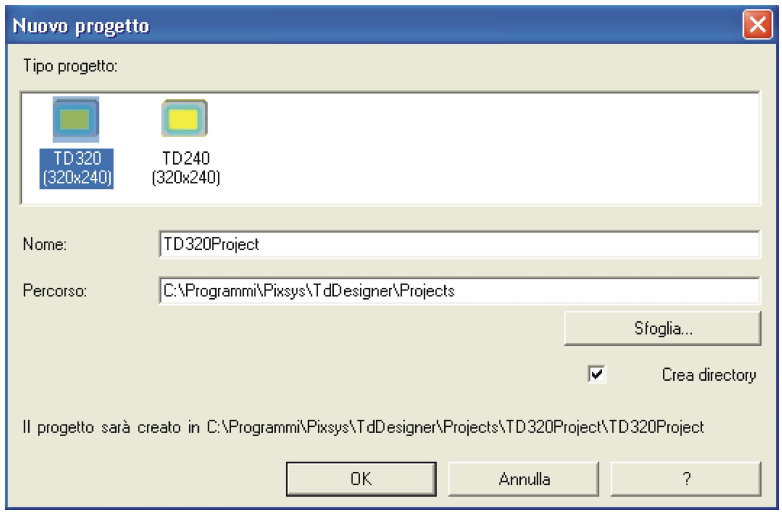


Figura 17.6: Selezione tipo terminale e nome del progetto

L'ambiente di sviluppo TdDesigner può inserire il nuovo progetto in una directory creata automaticamente oppure in un percorso scelto dall'utente.

17.5.1 Gestione lingue

Il numero di lingue associato all'applicazione può essere modificato in qualsiasi momento. Facendo click una volta il pulsante sinistro del mouse nella Finestra di Gestione, sopra il nome del progetto, la Finestra di Proprietà apparirà come in Figura 17.7

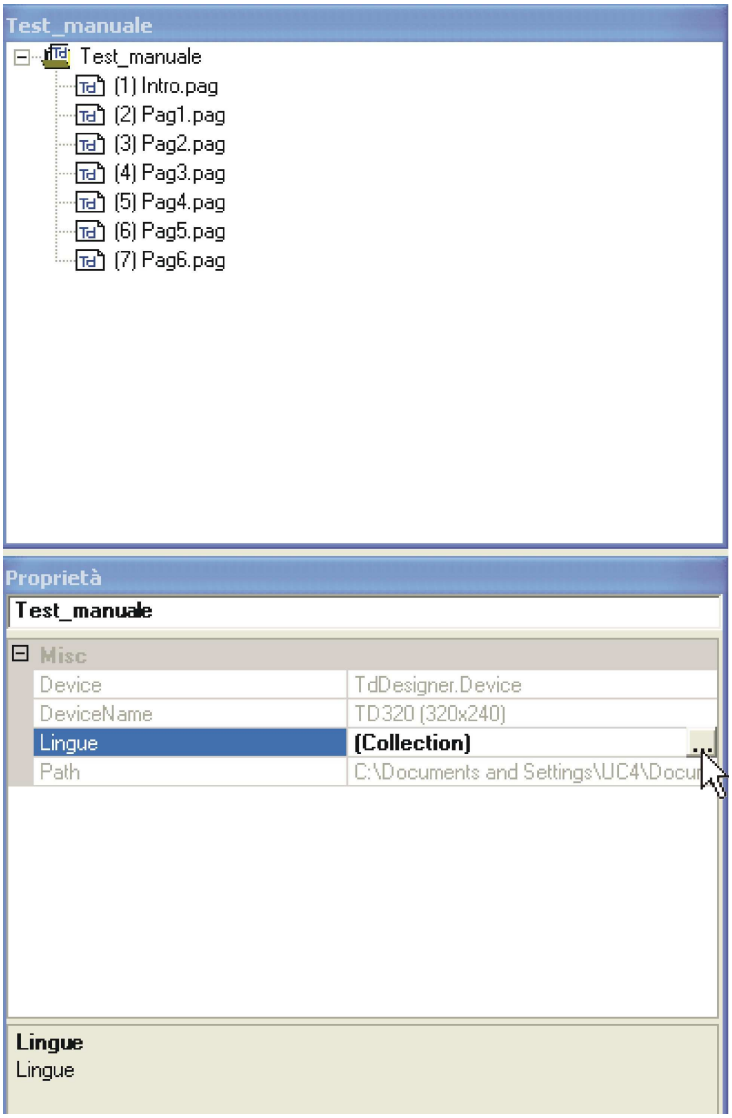


Figura 17.7: Modifica del numero di lingue.

Sempre in riferimento alla figura, facendo click su **Collection,** si accederà all'editor di immissione lingue. Quella di default è la prima dell'elenco.

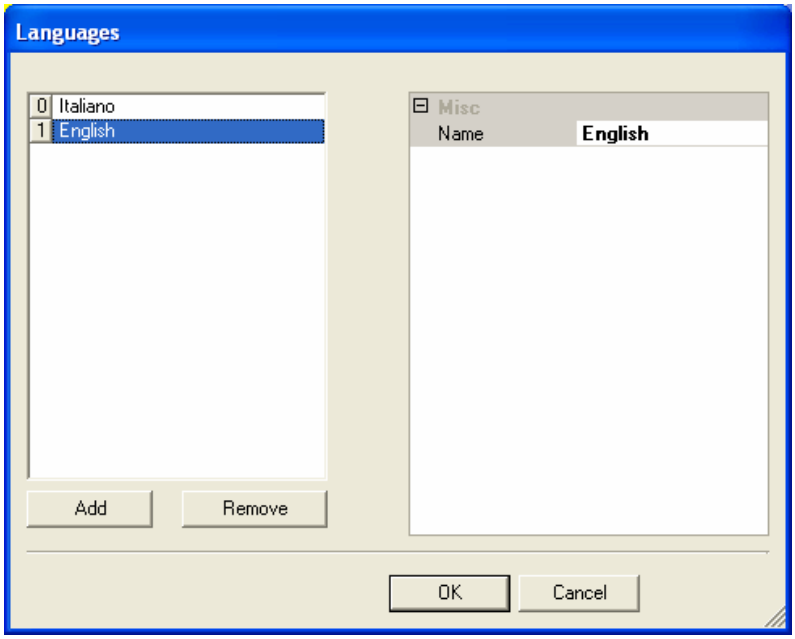


Figura 17.8: Editor di immissione Lingue. Per aggiungere una nuova lingua, premere il pulsante Add. Il nome va digitato alla destra di Name. Per rimuovere una lingua premere Remove.

Per inserire le traduzioni da associare agli oggetti contenenti testo, fare click nell'area **Strumenti / Lingue / Testo**, oppure fare click sull'icona di Figura 17.9 per accedere alla finestra **TdLinguist**.

Figura 17.9: Icona "Traduzioni".

Selezionare la lingua per cui si vogliono effettuare le traduzioni, come riportato in Figura 17.10.

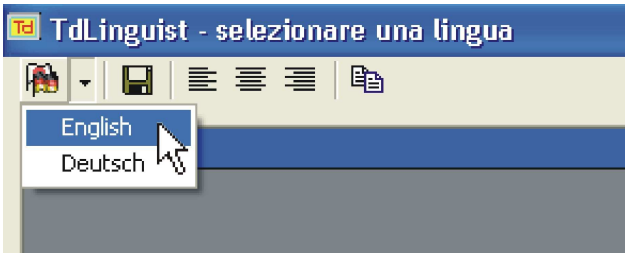


Figura 17.10: Finestra TdLinguist per le traduzioni.

Facendo un click su una delle lingue disponibili (devono esserne state impostate almeno 2 nella finestra **Languages** di Figura 17.8), compare il messaggio:

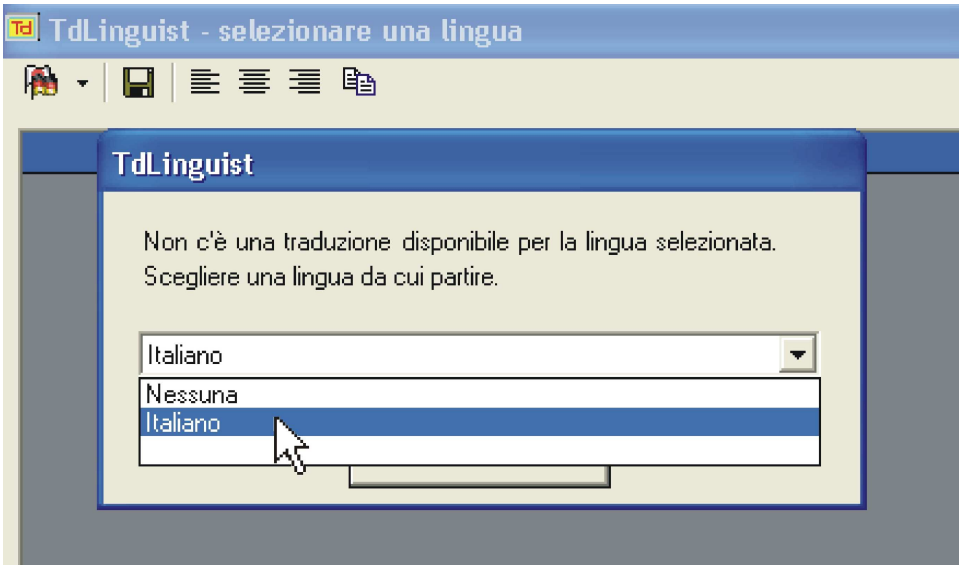
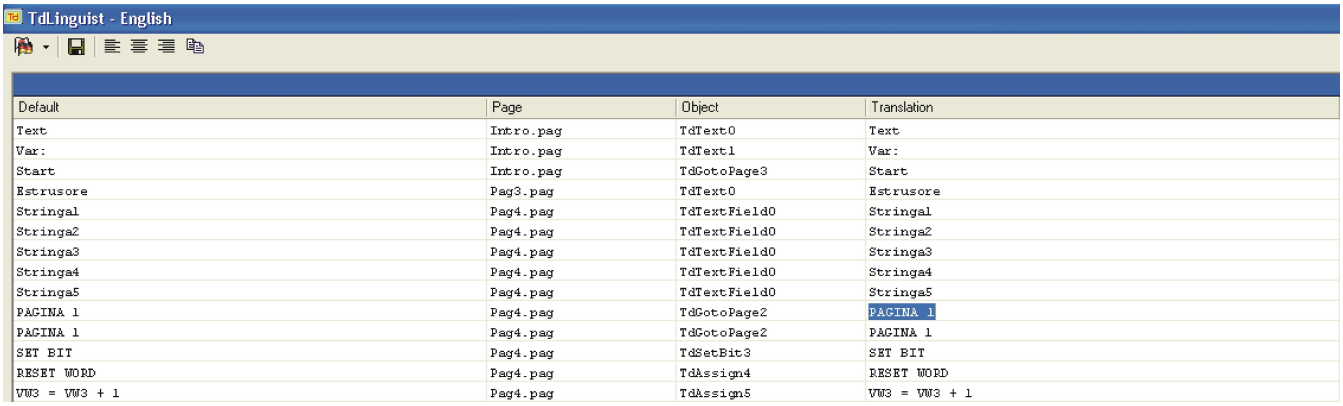


Figura 17.11: Finestra TdLinguist, messaggio d'avviso.

In base all'opzione selezionata, verrà creata una tabella in cui a ciascun testo sarà assegnata:

- una stringa vuota (**""**) se si sceglie **Nessuna**
- il testo associato alla lingua selezionata (**Italiano** nell'esempio riportato)

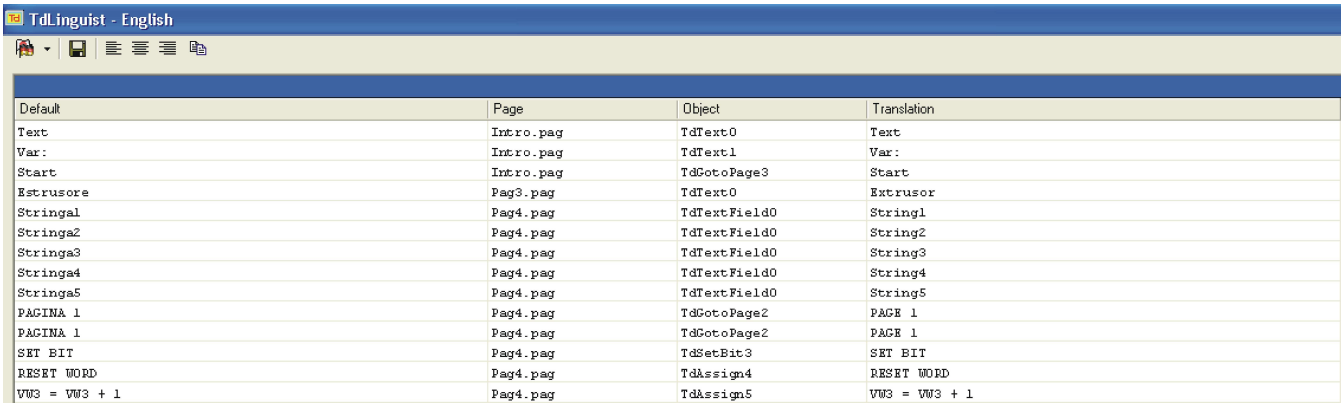


Default	Page	Object	Translation
Text	Intro.pag	TdText0	Text
Var:	Intro.pag	TdText1	Var:
Start	Intro.pag	TdGotoPage3	Start
Estrusore	Pag3.pag	TdText0	Estrusore
Stringa1	Pag4.pag	TdTextField0	Stringa1
Stringa2	Pag4.pag	TdTextField0	Stringa2
Stringa3	Pag4.pag	TdTextField0	Stringa3
Stringa4	Pag4.pag	TdTextField0	Stringa4
Stringa5	Pag4.pag	TdTextField0	Stringa5
PAGINA 1	Pag4.pag	TdGotoPage2	PAGINA 1
PAGINA 1	Pag4.pag	TdGotoPage2	PAGINA 1
SET BIT	Pag4.pag	TdSetBit3	SET BIT
RESET WORD	Pag4.pag	TdAssign4	RESET WORD
VW3 = VW3 + 1	Pag4.pag	TdAssign5	VW3 = VW3 + 1

Figura 17.12: Tabella delle traduzioni TdLinguist – English

Nella colonna **Default** sono riportate tutte le stringhe di tutti gli oggetti contenuti nel progetto, così come sono state inserite al momento della definizione.
Nella colonna **Page** viene riportata la pagina in cui è collocato l'oggetto proprietario di ciascuna stringa.
Nella colonna **Object** si riportano i nomi degli oggetti a cui le stringhe appartengono.

La colonna **Translation** riporterà le traduzioni che si scelgono per le stringhe.
Con riferimento Figura 17.13, l'utente può personalizzare le traduzioni, utilizzando eventualmente gli strumenti di allineamento testo presenti nella toolbar.
Molto spesso le applicazioni contengono oggetti con stringhe uguali, quindi riscriverne le traduzioni può essere laborioso. Il pulsante  permette di associare automaticamente a stringhe uguali la stessa traduzione.



Default	Page	Object	Translation
Text	Intro.pag	TdText0	Text
Var:	Intro.pag	TdText1	Var:
Start	Intro.pag	TdGotoPage3	Start
Estrusore	Pag3.pag	TdText0	Extrusor
String1	Pag4.pag	TdTextField0	String1
String2	Pag4.pag	TdTextField0	String2
String3	Pag4.pag	TdTextField0	String3
String4	Pag4.pag	TdTextField0	String4
String5	Pag4.pag	TdTextField0	String5
PAGINA 1	Pag4.pag	TdGotoPage2	PAGE 1
PAGINA 1	Pag4.pag	TdGotoPage2	PAGE 1
SET BIT	Pag4.pag	TdSetBit3	SET BIT
RESET WORD	Pag4.pag	TdAssign4	RESET WORD
VW3 = VW3 + 1	Pag4.pag	TdAssign5	VW3 = VW3 + 1

Figura 17.13: Tabella delle traduzioni TdLinguist – English

L'area di memoria che indicizza la lingua da utilizzare è **SMW13**
SMW8 = 0 Lingua1 Italiano
SMW8 = 1 Lingua2 Inglese
...

Proseguendo l'esempio, quando la **SMW13** assumerà il valore **1**, tutti gli oggetti con dei testi visualizzeranno le proprie stringhe caricandone il contenuto dalla colonna **Translation** di Figura 17.13.
Per inserire un'altra lingua, seguire la stessa procedura spiegata in questo paragrafo. Si dovrà scegliere un'altra lingua di partenza ed associare delle nuove traduzioni per le stringhe degli oggetti. La nuova lingua sarà associata al valore **2** della **SMW13**.

17.5.2 Gestione variabili

Anche l'insieme delle variabili può essere modificato in qualsiasi momento tramite l'editor, rappresentato in Figura 17.15, richiamato dalla voce di menu "**Modifica \ Variabili**" (Figura 17.2 al centro) oppure dal pulsante di Figura 17.14.

Figura 17.14: Icona "Modifica variabili"

Il campo **Filtro** seleziona se visualizzare tutte le variabili di progetto (prima opzione *****) o le sole variabili contenute in una particolare pagina. Nel campo **Aggiungi** va selezionato il tipo di area di memoria della variabile che si vuole inserire (# indica costanti numeriche). Il pulsante → inserisce la sola variabile selezionata, **[...]**→ inserisce al massimo un gruppo di 100 variabili contigue.

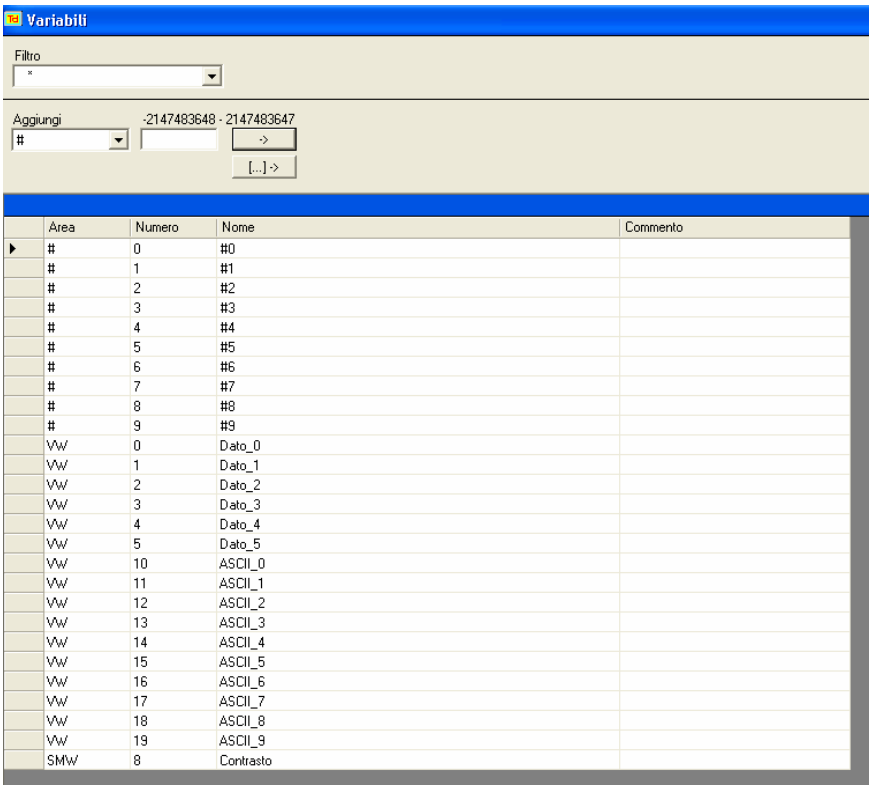


Figura 17.15: editor dell'insieme Variabili. L'elenco può essere visualizzato in ordine alfabetico in base al nome assegnato dall'utente con un click sulla casella "Nome" o in ordine progressivo per area di memoria (default) con un click sulle caselle "Area" o "Numero".

Le variabili devono sempre essere dichiarate nella tabella prima di essere utilizzate negli oggetti grafici.

17.5.3 Gestione pagine

L'insieme delle pagine di progetto con il relativo numero progressivo è schematizzato nella Finestra di Gestione Progetto (Figura 17.1). Per aprire una pagina creata in precedenza è sufficiente un doppio click sul nome. Per aggiungere una nuova pagina, richiamare la voce di menu "**File \ Nuovo \ Pagina**", o il corrispondente pulsante (vedi par. 17.3).
L'ordine delle pagine è significativo perché all'accensione del terminale sarà sempre visualizzata la prima. Ciascuna pagina potrà in ogni momento essere impostata come prima da un apposito menu come illustrato in Figura 17.16

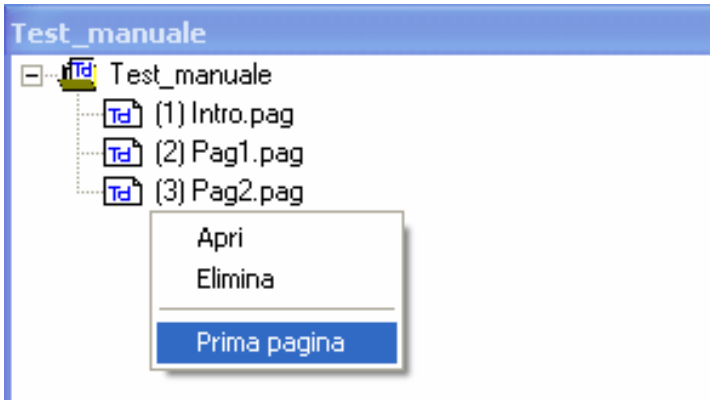


Figura 17.16: menu di contesto, si apre con un click del tasto destro su un nome di pagina.

Una volta creata una pagina, occorre impostare le sue proprietà: il Tempo di Refresh che può variare da 100ms a 5s e lo sfondo che può essere di colore uniforme (vedi Figura 17.3), oppure un immagine bitmap[4] di dimensioni pari alla risoluzione del display (320x240). In particolare per quest'ultima operazione si legga quanto riportato nel par. 17.6.7 a proposito dell'oggetto bitmap.

[4] Il terminale TD320 è in grado di visualizzare immagini a 256 colori con palette fissa.

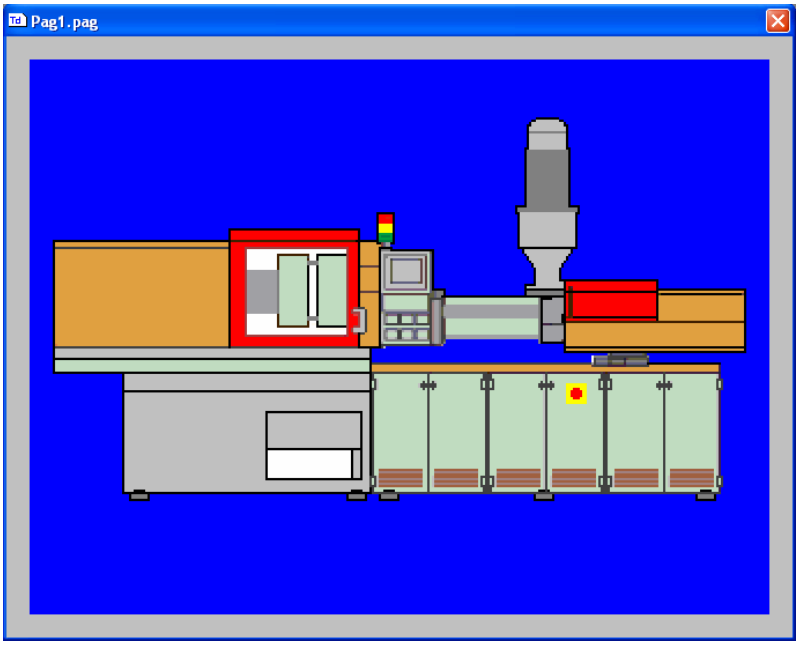


Figura 17.17: una pagina con sfondo caricato da un'immagine bitmap.

17.6 Oggetti grafici

La configurazione di una pagina passa attraverso l'impostazione delle sue proprietà e l'inserimento di oggetti grafici. Per introdurre un nuovo oggetto è sufficiente selezionarlo dalla Casella Strumenti e posizionarlo nella finestra che rappresenta la pagina. Il terminale è sensibile all'ordine di inserimento: se due oggetti sono sovrapposti verrà visualizzato per intero solo l'ultimo. Per questo nel menu "**Modifica**" (Figura 17.2 al centro) sono presenti le voci "**Sposta dietro**" (anche con i pulsanti della tastiera **Ctrl+B**) e "**Sposta davanti**" (anche con **Ctrl+F**), che consentono di modificare la posizione reciproca degli oggetti. La stessa funzione è disponibile con i pulsanti della toolbar

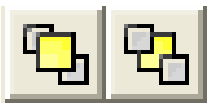


Figura 17.18: "Sposta davanti" a sinistra, "Sposta dietro" a destra

Figura 17.19: Casella Strumenti. A parte lo strumento puntatore, contiene tutti gli oggetti che si possono disporre su una pagina.

17.6.1 Forme geometriche

17.6.1.1 Oggetto Linea

Una volta selezionato dalla Casella Strumenti, per disegnare una linea nella pagina, fare click con il mouse tenendo premuto il tasto sinistro e trascinare il puntatore fino alla lunghezza desiderata.

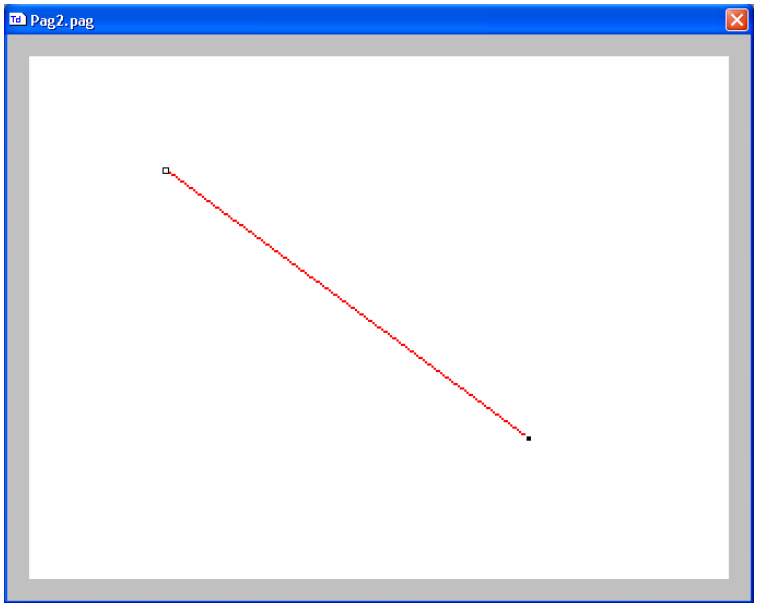


Figura 17.20: Oggetto Linea in pagina Pag2.

L'oggetto potrà essere spostato e ridimensionato anche successivamente all'inserimento sia con il mouse che dalla finestra Proprietà, che ora si presenta come in Figura 17.21
Sono modificabili, oltre al **ColoreLinea**, le coordinate di partenza (**X1**, **Y1**), di arrivo (**X2**, **Y2**) e il nome dell'oggetto (**TdLine0**, assegnato di default).

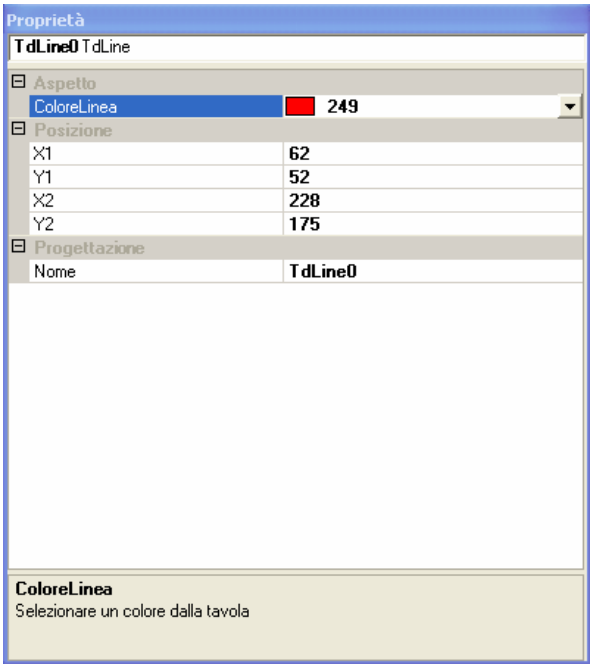
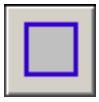


Figura 17.21: Tabella Proprietà dell'oggetto Linea di pagina Pag2.

17.6.1.2 Oggetto Rettangolo



Una volta selezionato dalla Casella Strumenti, per disegnare un rettangolo nella pagina, fare click con il mouse tenendo premuto il tasto sinistro e trascinare il puntatore in modo da definire le dimensioni.

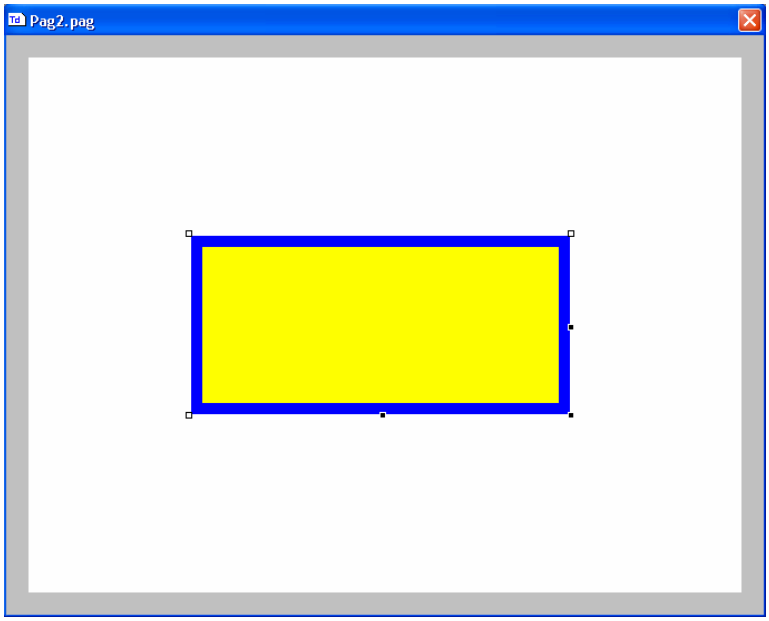


Figura 17.22: Oggetto Rettangolo in pagina Pag2.

L'oggetto potrà essere spostato e ridimensionato anche successivamente all'inserimento sia con il mouse che dalla Finestra Proprietà, che ora si presenta come in Figura 17.23.
Sono modificabili, oltre al **ColoreLinea** (nell'esempio è il blu, il perimetro del rettangolo), il **ColoreRiempimento** (nell'esempio è giallo, l'interno del rettangolo), il campo **Pieno** (nell'esempio è **True**, rettangolo pieno, per avere il solo perimetro con interno vuoto selezionare **False**), le coordinate del vertice superiore sinistro (**Sinistra** asse X, **Superiore** asse Y), le dimensioni (**Larghezza**, **Altezza**) ed il nome dell'oggetto (**TdRectangle0**, assegnato di default).

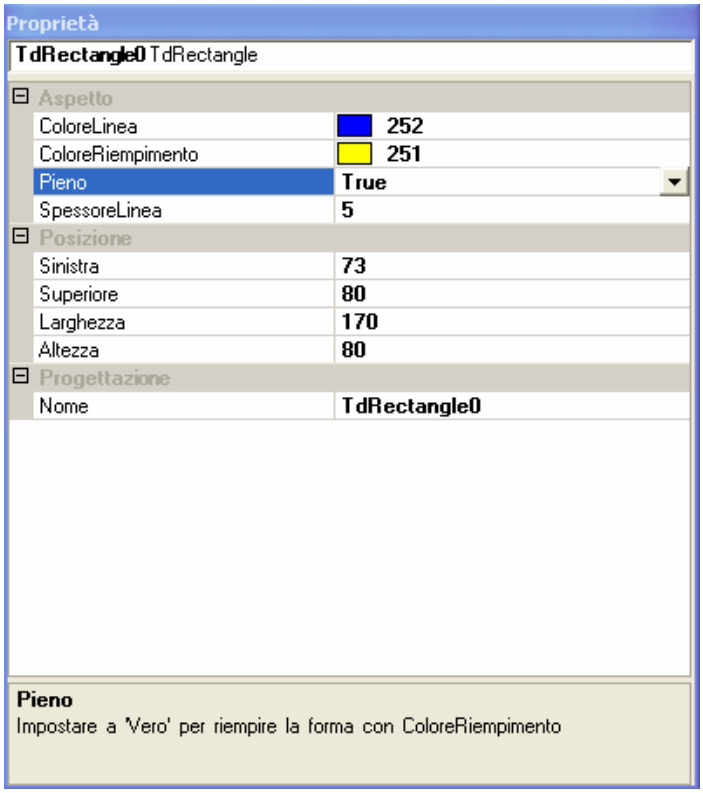
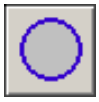


Figura 17.23: Finestra Proprietà dell'oggetto Rettangolo di pagina Pag2.

17.6.1.3 Oggetto Ellisse



Si disegna allo stesso modo ed ha le stesse proprietà dell'oggetto Rettangolo. Per ottenere una circonferenza basterà impostare i campi **Larghezza** ed **Altezza** uguali.

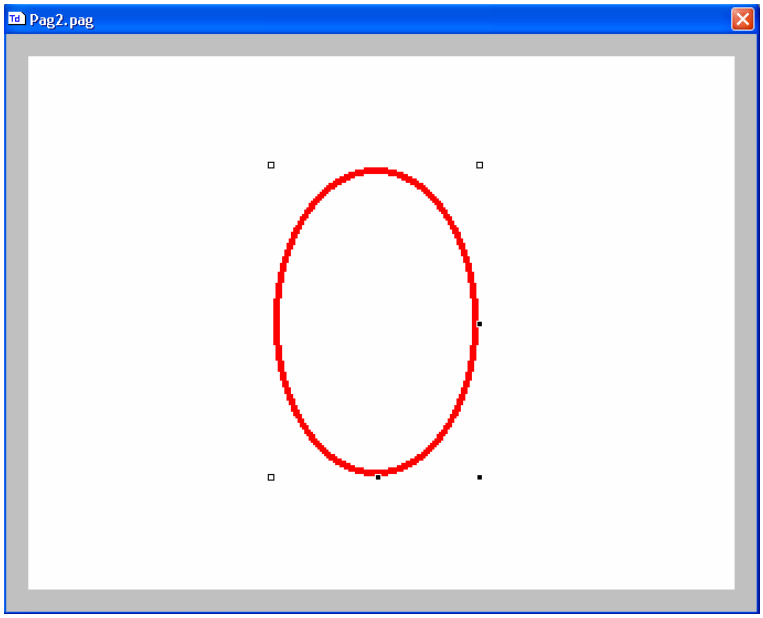


Figura 17.24: Oggetto Ellisse in pagina Pag2.

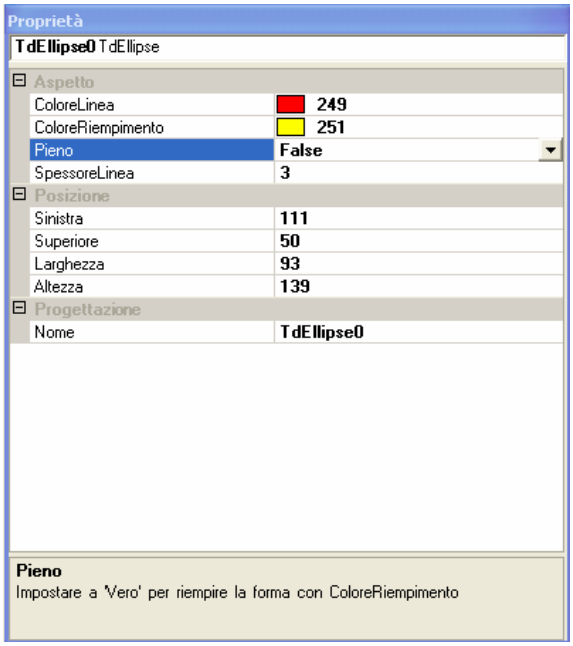


Figura 17.25: Finestra Proprietà dell'oggetto Ellisse di pagina Pag2.

17.6.2 Oggetto Etichetta

Una volta selezionato l'oggetto dalla Casella Strumenti, posizionarlo nella pagina con un semplice click. Il testo può essere immesso direttamente da tastiera se l'etichetta è stata appena

inserita o selezionata, oppure dalla Finestra Proprietà nel campo "**Testo**". Con riferimento alla scritta **TEXT**, visualizzata in Figura 17.1, si riporta la corrispondente Tabella Proprietà.

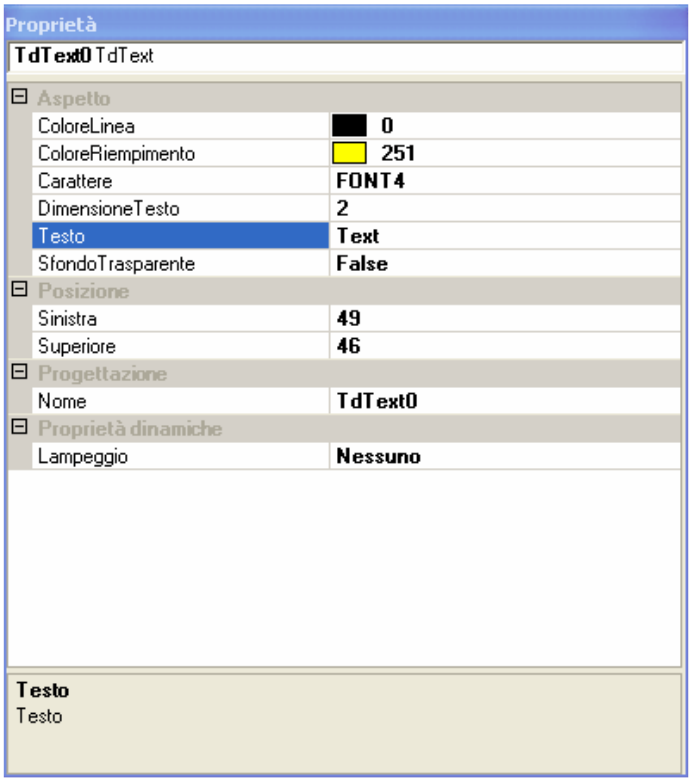


Figura 17.26: Finestra Proprietà dell'oggetto etichetta TEXT di Figura 17.1

Sono modificabili il **ColoreLinea** (colore del testo nella pagina), il **ColoreRiempimento** (colore dello sfondo rettangolare del testo), il **Carattere** (sono disponibili 5 font), la **Dimensione** del testo (moltiplicatore per 1, 2 o 4 delle dimensioni originali del font), le coordinate dell'angolo superiore sinistro (**Sinistra** asse X, **Superiore** asse Y) ed il nome dell'oggetto (**TdText0**, assegnato di default).
Il campo **SfondoTrasparente** è stato creato per poter sovrapporre un testo ad un'immagine (si pensi ad un sinottico), visualizzando solo la scritta senza il suo contorno rettangolare, come nella Figura 17.27

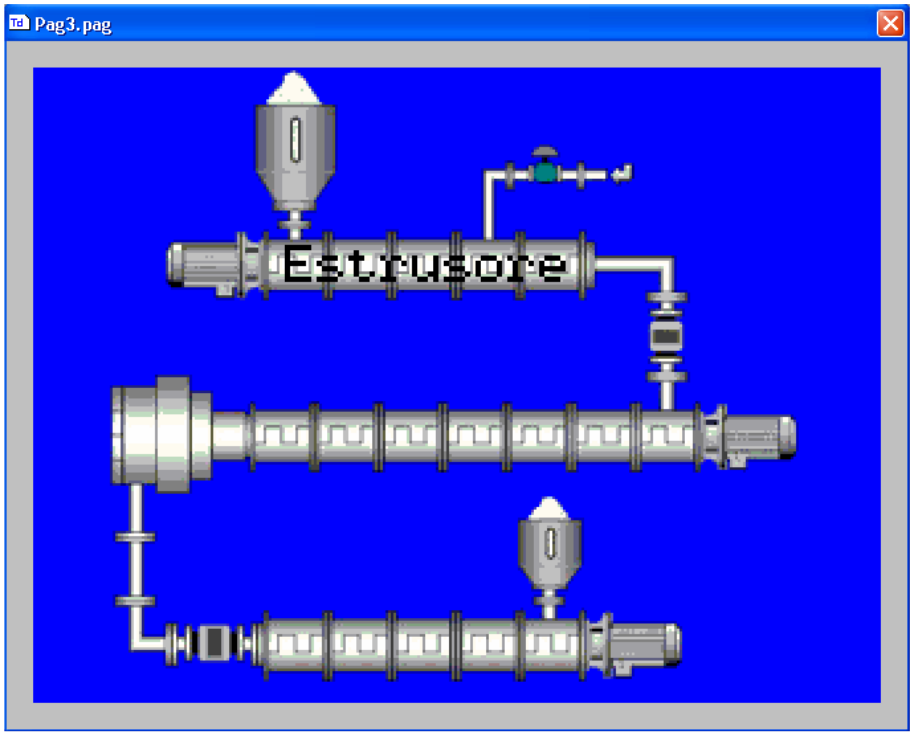


Figura 17.27: L'etichetta "Estrusore" è sovrapposta all'immagine di sfondo pagina con SfondoTrasparente True.

La Tabella Proprietà relativa all’oggetto descritto è la seguente:

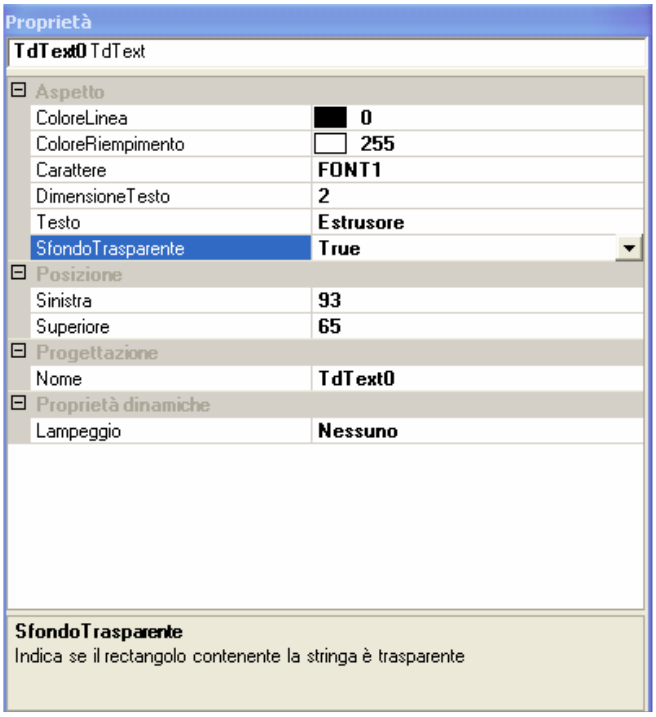


Figura 17.28: Tabella Proprietà dell'oggetto Etichetta "Estrusore" di pagina Pag3.

Lampeggio: Il testo dell'oggetto Etichetta può lampeggiare (periodo fisso 1 secondo). Esistono 4 tipi di lampeggio:

5. **Nessuno**: nessun lampeggio
6. **Testo**: lampeggia solo il testo e lo sfondo rettangolare resta fisso
7. **Sfondo**: il testo resta fisso e lampeggia lo sfondo rettangolare
8. **Tutto**: lampeggiano entrambi

17.6.3 Oggetto Campo Numerico

Consente di visualizzare e modificare il valore di una variabile. Una volta selezionato l'oggetto dalla Casella Strumenti, posizionarlo nella pagina con un semplice click. Per poter collegare un campo numerico a una varabile occorre che questa sia già presente nell'elenco associato al progetto (vedi par. 17.5.2, in questo esempio è l'area di memoria **VW0**, nominata **Dato_0**). Con riferimento al campo numerico **0000**, visualizzato in Figura 17.1, si riporta la corrispondente Tabella Proprietà.

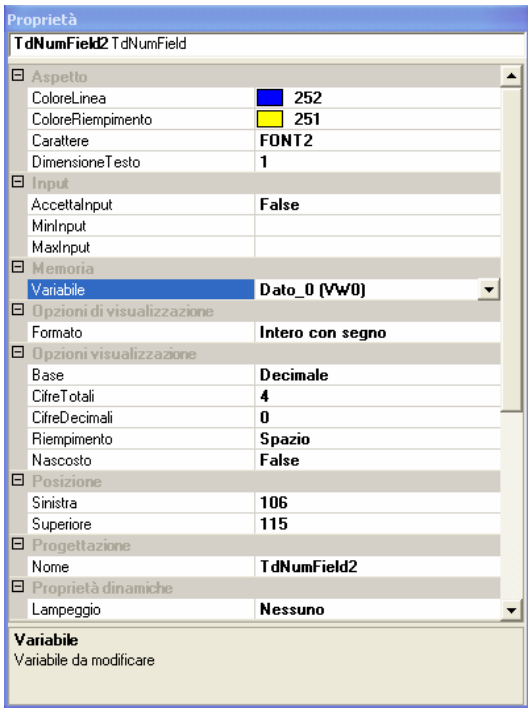


Figura 17.29: Prima parte della Tabella Proprietà dell'oggetto Campo Numerico di pagina Intro.

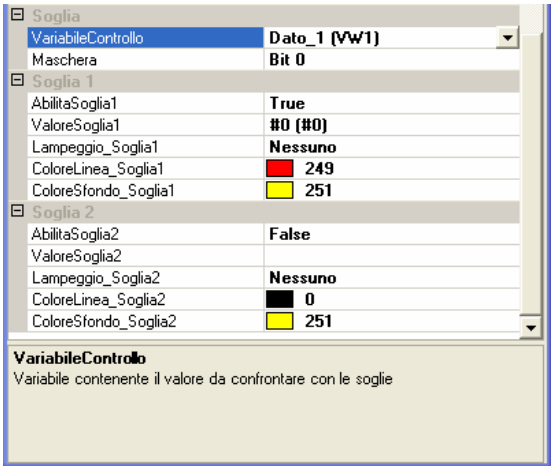


Figura 17.30: Seconda parte della Tabella Proprietà dell'oggetto Campo Numerico di pagina Intro. Soglia1 abilitata, Soglia2 non abilitata

I campi **ColoreLinea**, **ColoreRiempimento**, **Carattere**, **DimensioneTesto**, **Sinistra**, **Superiore, Nome** e **Lampeggio** sono analoghi a quelli dell'oggetto Etichetta (vedi par. 17.6.2).

Memoria: al campo **Variabile** deve essere collegata un'area di memoria precedentemente impostata (vedi par. 17.5.2).

Input: questo campo definisce se la variabile deve essere editabile oppure no. Se il campo **AccettaInput** è **False**, come nell'esempio riportato, la variabile non è modificabile dall'operatore. Se è **True**, l'operatore avrà la possibilità di modificare il valore della variabile entro i limiti **MinInput** - **MaxInput** (costanti o variabili), tramite un'apposita pagina caricata con una pressione sull'oggetto.
Opzioni di visualizzazione: il campo **Formato** identifica il tipo di dato e quindi la sua visualizzazione sul terminale. **Intero con segno** e **Intero senza segno** devono essere utilizzati per variabili word (16bit), **Long con segno** per le double word (32bit).
Il campo **Base** indica la base matematica del dato (**Decimale**, **BCD**, **Esadecimale**, **Binario**).
Il numero di **CifreTotali** dovrà tenere conto anche degli eventuali punto decimale e segno. Ad esempio, con 4 **CifreTotali** e 1 **CifreDecimali**, se il valore della variabile VW0 fosse 8375, nel terminale si vedrebbe 37.5, non visualizzando la cifra più

significativa (per una corretta visualizzazione il numero di **CifreTotali** dovrebbe essere 5).
Il campo **Riempimento** dà la possibilità di riempire le cifre più significative vuote con **Spazio** o **Zero**.
Il campo **Nascosto** è generalmente **False** per variabili che devono essere visualizzate e modificate, **True** per variabili il cui valore non deve essere conosciuto (utile per immissione di password, ad esempio). In quest'ultimo caso le cifre in visualizzazione verranno sostituite dal carattere *****

Soglia, Soglia1, Soglia2: il campo **VariabileControllo** dovrà essere associato alla variabile responsabile dell'eventuale variazione di alcune proprietà di visualizzazione.
A seconda del valore di **Maschera**, la **VariabileControllo** sarà valutata per intero (**Nessuna**) oppure in uno dei suoi bit (**bit0**, **bit1**, ... , **bit31**).
Per ciascuna delle due Soglie (Soglia1 e Soglia2), il controllo può essere abilitato (**AbilitaSoglia True**) oppure disabilitato (**AbilitaSoglia False**).
Il campo **ValoreSoglia** è il valore di confronto, da intendersi come ≥ (maggiore o uguale).

- Nel caso di nessuna maschera (**Maschera = Nessuna**), il confronto ha esito positivo se **VariabileControllo≥ValoreSoglia**
- Nel caso di maschera a bit (**Maschera = bit N**), il confronto ha esito positivo se **bit N** di **VariabileControllo = 1** (**ValoreSoglia** non ha alcun significato)

Nel caso di esito positivo del confronto, l'oggetto cambierà le sue caratteristiche di visualizzazione per quanto riguarda il lampeggio (**Lampeggio_Soglia**), colore della scritta (**ColoreLinea_Soglia**) e colore dello sfondo (**ColoreRiempimento_Soglia**).

In base alle Tabelle Proprietà di Figura 17.29 e Figura 17.30, finchè il **bit 0** della variabile **VW1** è uguale a **0**, la variabile **VW0** è visualizzata in blu su sfondo giallo. Quando il bit **0** della variabile **VW1** sarà uguale a **1**, la variabile **VW0** sarà visualizzata in rosso su sfondo giallo.

17.6.4 Oggetto Campo Testo

A partire da un elenco di stringhe, visualizza un testo indicizzato dal valore di una variabile. La stringa nell'ambiente di sviluppo è sempre quella associata alla prima dell'elenco, mentre la stringa visualizzata dal terminale è quella del valore corrispondente:
Variabile = 0 → Stringa1
Variabile = 1 → Stringa2
Variabile = 2 → Stringa3
Variabile = 3 → Stringa4
Variabile = 4 → Stringa5
...
La Figura 17.31 contiene un oggetto Campo Testo visualizzato come **Stringa1**.

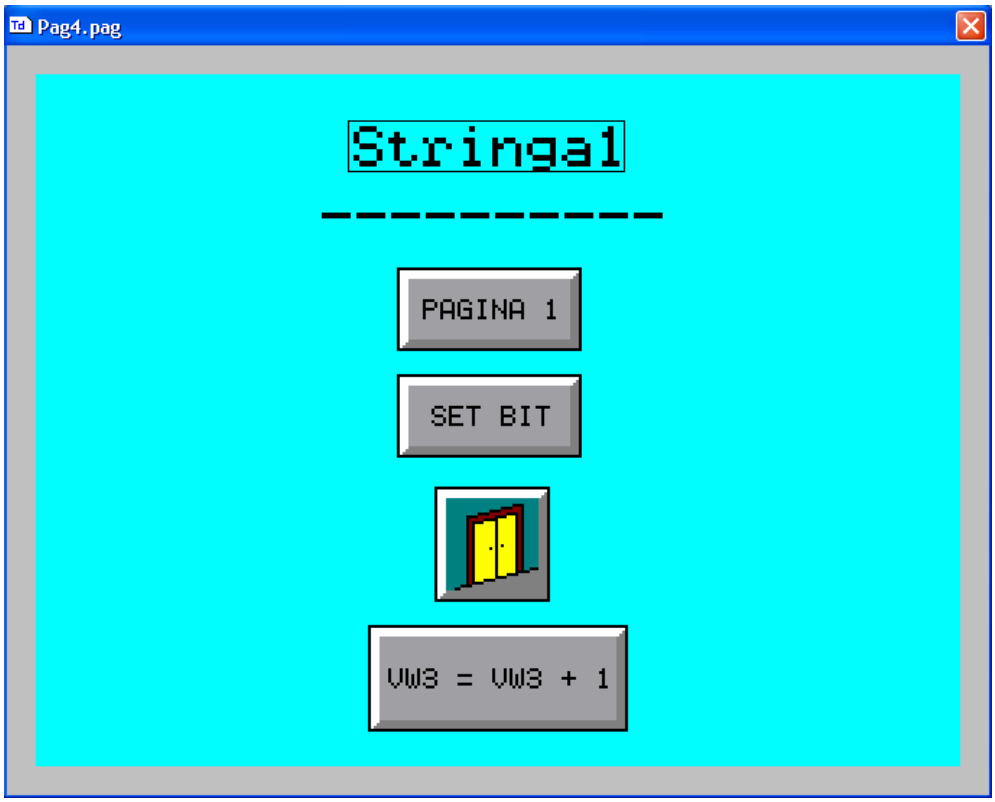


Figura 17.31: Oggetto Campo Testo "Stringa1" nella pagina Pag4

L'elenco delle stringhe è accessibile dal campo **Aspetto / Tabella Stringhe** e si presenta come in Figura 17.32

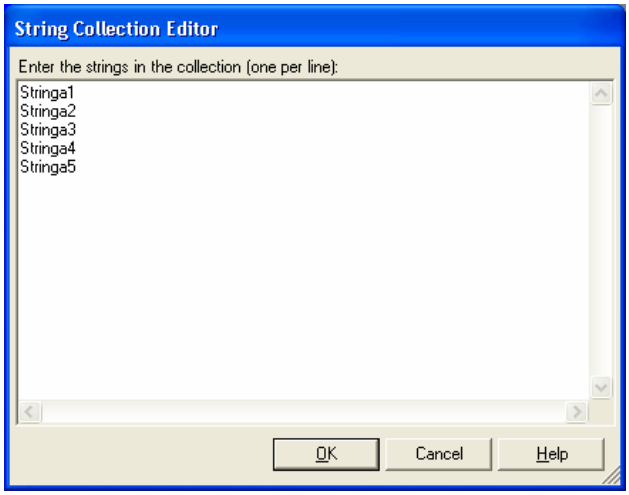


Figura 17.32: Impostazione della Tabella Stringhe per Campo Testo "Stringa1"

La lunghezza di tutte le stringhe dell'elenco è automaticamente riportata a quella della prima stringa, quindi nel caso di stringhe di lunghezza diversa è necessario calcolare la più lunga ed eventualmente riempire la prima con degli spazi.

In riferimento al Campo Testo **Stringa1**, si riporta la corrispondente Tabella Proprietà.

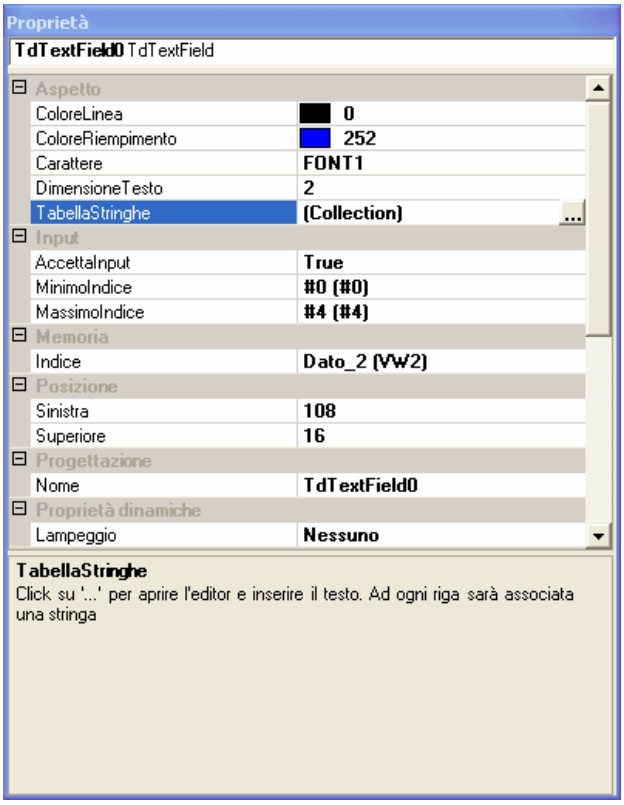


Figura 17.33: Prima parte della Tabella Proprietà dell'oggetto Campo Testo "Stringa1" di pagina Pag4.

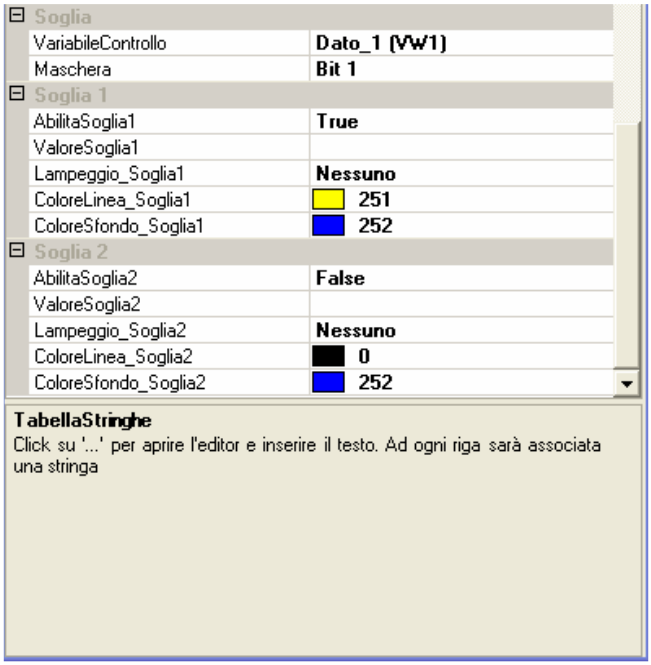


Figura 17.34: Seconda parte della Tabella Proprietà dell'oggetto Campo Testo "Stringa1" di pagina Pag4.

I campi **ColoreLinea**, **ColoreRiempimento**, **Carattere**, **DimensioneTesto**, **Sinistra**, **Superiore, Nome**, **Lampeggio, VariabileControllo, Maschera, AbilitaSoglia, ValoreSoglia, Lampeggio_Soglia, ColoreLinea_Soglia, ColoreSfondo_Soglia** sono analoghi a quelli degli Campo Numerico ed Etichetta (vedi par. 17.6.2 e 17.6.3).

Input: il campo **AccettaInput** definisce se l'oggetto può essere modificabile dall'operatore (**True**) oppure solo visualizzabile (**False**).
I campi **MinimoIndice** e **MassimoIndice** delimitano l'insieme delle stringhe selezionabili dall'operatore.

Memoria: il campo **Indice** contiene la variabile, preventivamente impostata, che indicizza la lista di stringhe inserite. Per visualizzare correttamente tutte le N stringhe della lista, la variabile **Indice** deve variare da 0 a N-1.
In riferimento alla Tabella Proprietà di Figura 17.33 e Figura 17.34, l'oggetto Campo Testo è una stringa indicizzata dalla variabile **VW2**, modificabile dall'utente che può scorrerle dalla prima (**VW2=0**) all'ultima (**VW2=4**). Il colore della stringa visualizzata sarà nero su sfondo blu finchè il bit **1** della variabile **VW1** sarà

uguale a **0**, e giallo su sfondo blu quando il bit **1** della variabile **VW1** sarà uguale a **1**.

17.6.5 Oggetto Campo ASCII

Questo oggetto permette all'operatore di inserire un testo libero, formato da lettere, cifre e altri caratteri. Il numero massimo dei caratteri disponibili è deciso dal programmatore in fase di definizione dell'oggetto.
Ecco come si presenta nell'ambiente di sviluppo un campo ASCII di 10 caratteri.

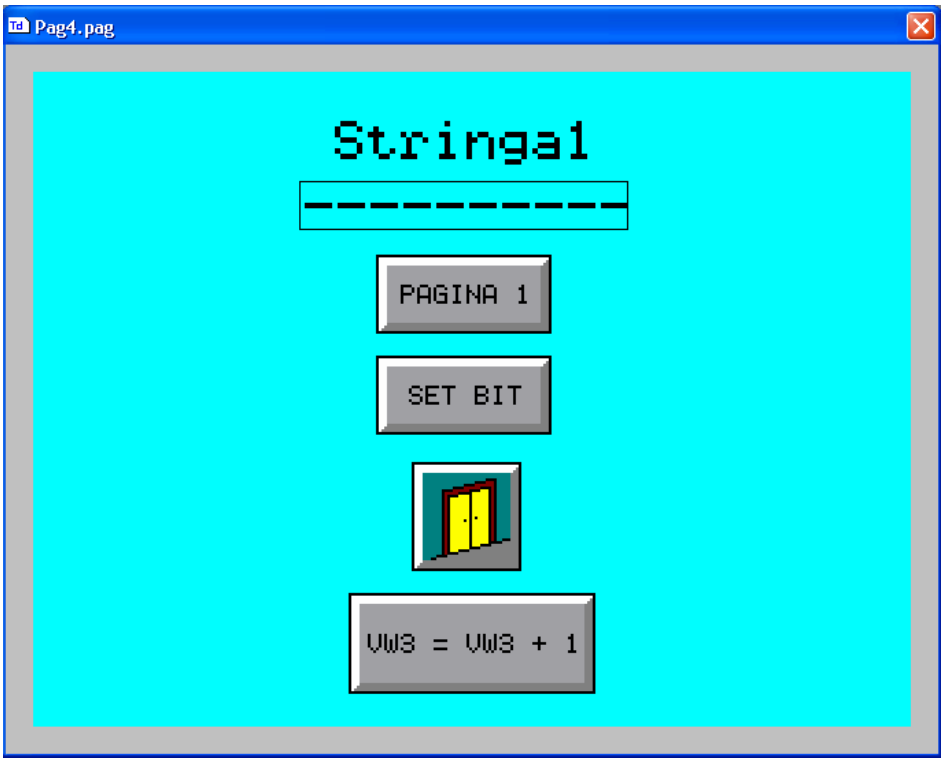


Figura 17.35: Campo ASCII in pagina Pag4 (dieci trattini consecutivi)

In riferimento al campo ASCII visualizzato in Figura 17.35, si riporta la corrispondente Tabella Proprietà.

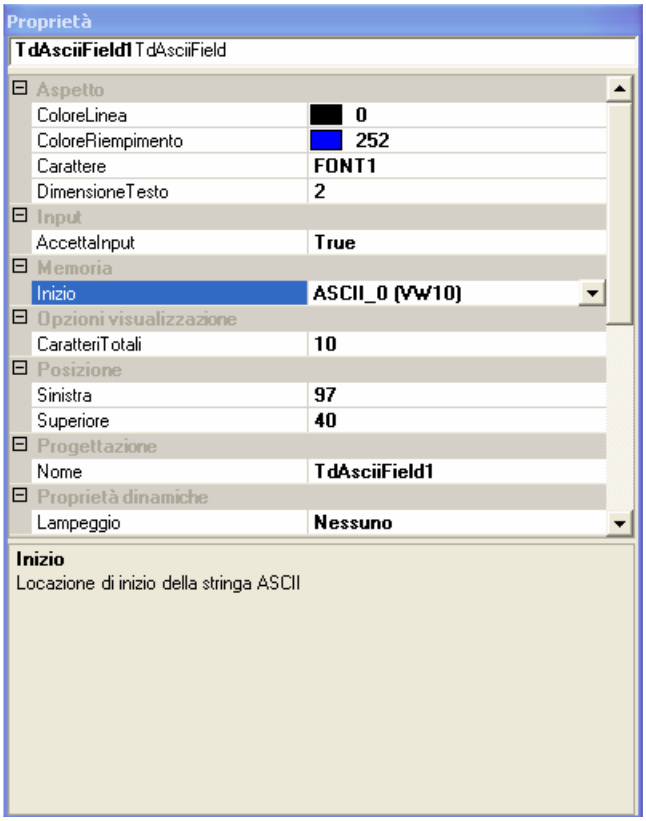


Figura 17.36: Prima parte della Tabella Proprietà dell'oggetto Campo ASCII

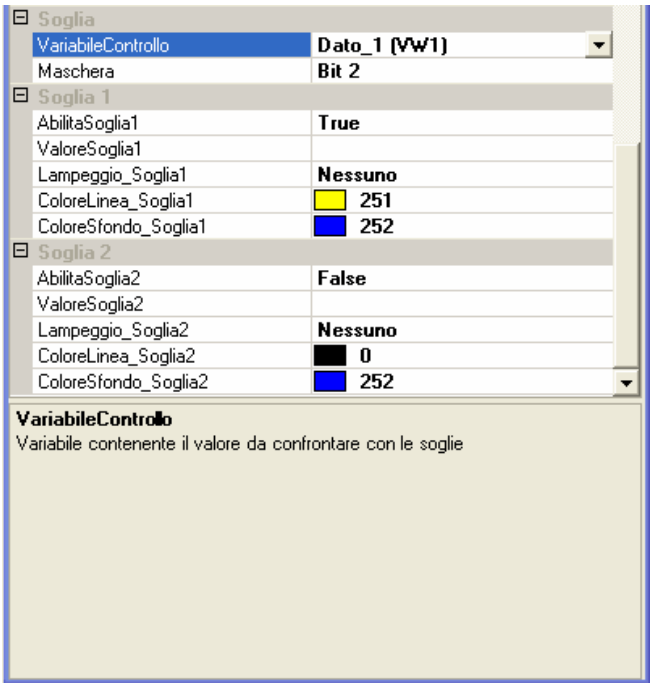


Figura 17.37: Seconda parte della Tabella Proprietà dell'oggetto Campo ASCII

I campi **ColoreLinea**, **ColoreRiempimento**, **Carattere**, **DimensioneTesto**, **Sinistra**, **Superiore, Nome**, **Lampeggio, VariabileControllo, Maschera, AbilitaSoglia, ValoreSoglia,**

Lampeggio_Soglia, ColoreLinea_Soglia, ColoreSfondo_Soglia sono analoghi a quelli degli Campo Numerico, Etichetta e Testo (vedi par. 17.6.2, 17.6.3 e 17.6.4).

Input: il campo **AccettaInput** definisce se l'oggetto può essere modificabile dall'operatore (**True**) oppure solo visualizzabile (**False**).

Memoria: il campo **Inizio** fissa l'area di memoria che sarà utilizzata per salvare il primo carattere ASCII.

Opzioni visualizzazione: il campo **CaratteriTotali** determina la lunghezza totale del Campo ASCII, e può essere solo una costante numerica.
Se al campo **Inizio** è associata la variabile $\mathbf{VW_x}$, e i **CaratteriTotali** sono 10, le aree di memoria da $\mathbf{VW_x}$ a $\mathbf{VW_{x+9}}$ conterranno le codifiche ASCII dei caratteri inseriti.

In riferimento alla Tabella Proprietà di Figura 17.36 e Figura 17.37, l'oggetto Campo ASCII occupa le 10 word da **VW10** a **VW19**. La scritta sarà nera su sfondo blu finchè il bit **2** della word **VW2** sarà **0**, e giallo su sfondo blu quando il bit **2** della word **VW2** sarà **1**.

17.6.6 Oggetti Pulsante

Pulsante cambio pagina

Pulsante set bit

Pulsante assegnamento variabile

Dal punto di vista della visualizzazione, i pulsanti sono uguali. In particolare è possibile impostare lo stile grafico (3D, piatto o nascosto), il testo o l'immagine sulla superficie del tasto e i colori di linea e di riempimento.

Questi oggetti prevedono inoltre la possibilità di cambiare alcune caratteristiche in base al confronto di una variabile di controllo con due possibili valori di soglia: possono variare colori, testo, l'immagine e l'abilitazione della funzione ad essi associata.

17.6.6.1 Pulsante Cambio Pagina

Il pulsante Cambio Pagina è caratterizzato dal campo **Azione/NumeroPagina** a cui va associata una costante o una variabile che identifica la pagina che il terminale deve visualizzare alla pressione del tasto. La Figura 17.38 riporta il pulsante **PAGINA 1**, e la Figura 17.39 e Figura 17.40 la sua Tabella Proprietà.

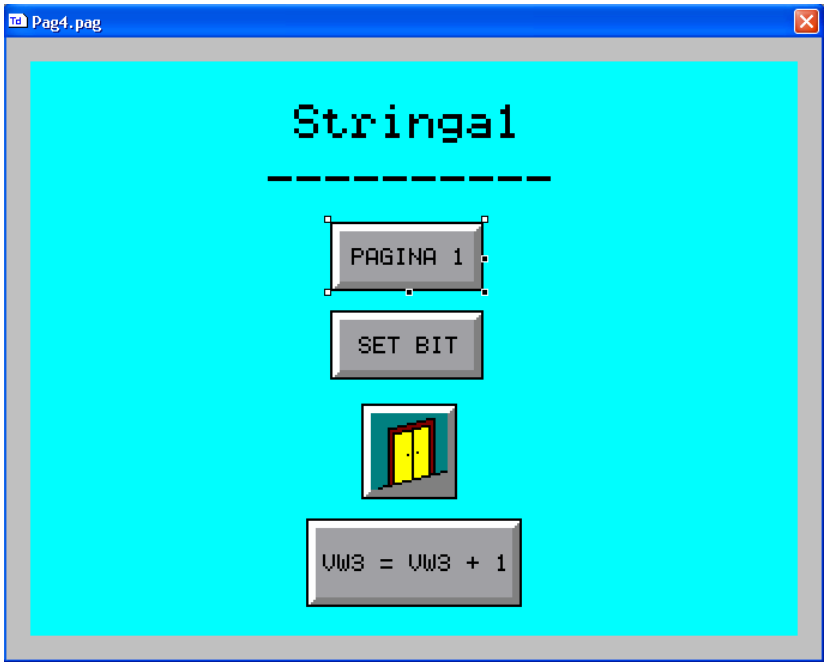


Figura 17.38: Pulsante Cambio Pagina "PAGINA 1" di pagina Pag4

I campi **ColoreLinea**, **ColoreRiempimento**, **Carattere**, **DimensioneTesto**, **Sinistra**, **Superiore, Nome, VariabileControllo, Maschera, AbilitaSoglia, ValoreSoglia, ColoreLinea_Soglia, ColoreSfondo_Soglia** sono analoghi a quelli degli Campo Numerico, Etichetta, Testo ed ASCII (vedi par. 17.6.2, 17.6.3, 17.6.4 e 17.6.5).

Aspetto: nel campo **Testo** va inserita la scritta che si vuole visualizzare sul pulsante.
Nel campo **ListaImmagini** è possibile associare una o più immagini bitmap (vedi nota 4 a pag. 57) da visualizzare. Per importarle, fare un semplice click con il pulsante sinistro del mouse nel punto indicato dalla Figura 17.41. Apparirà l'editor di Figura 17.42.
La proprietà **ListaImmagini** ha la precedenza sulla proprietà **Testo**.

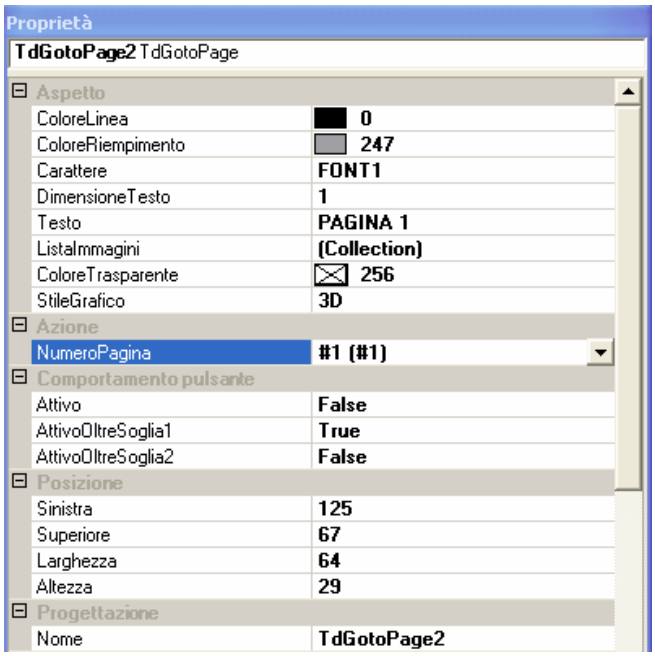


Figura 17.39: Prima parte Tabella Proprietà dell'oggetto Pulsante Cambio Pagina "PAGINA 1" di pagina Pag4

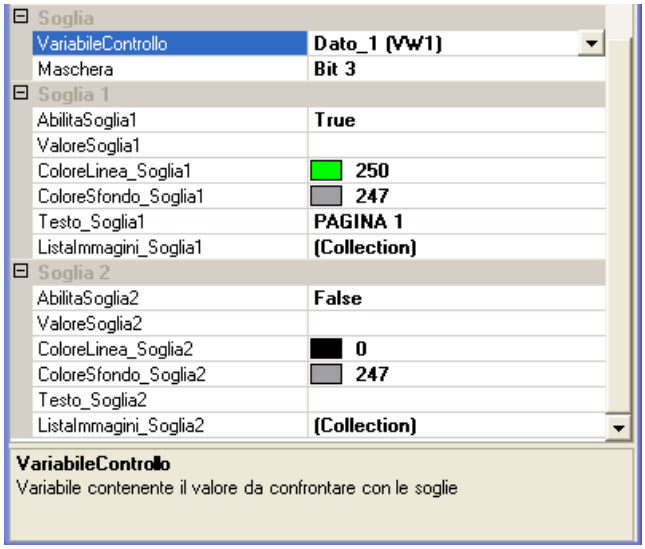


Figura 17.40: Seconda parte Tabella Proprietà dell'oggetto Pulsante Cambio Pagina "PAGINA 1" di pagina Pag4

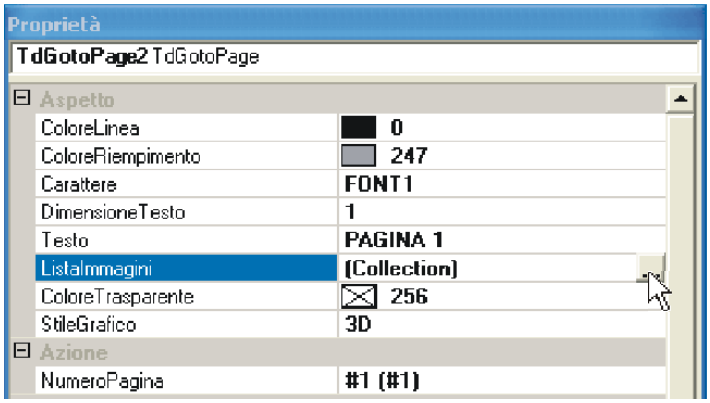


Figura 17.41: Apertura dell'editor per ListaImmagini dell'oggeto Pulsante Cambio Pagina.

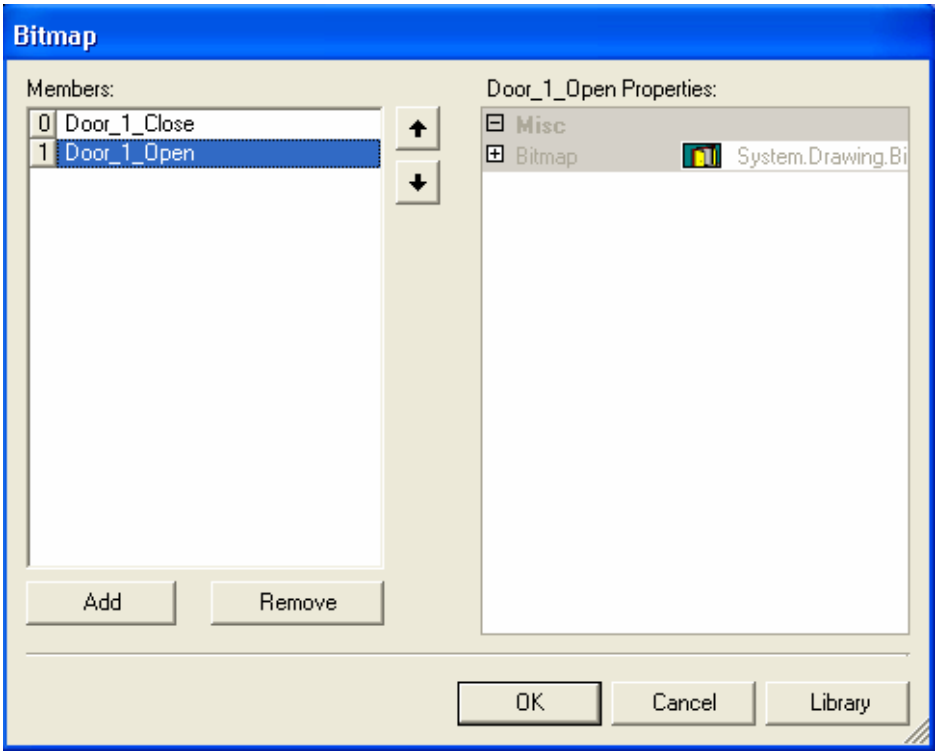


Figura 17.42: Editor. Per inserire una nuova immagine nell'insieme premere il tasto "Aggiungi" e selezionare un file in formato bitmap. Le frecce a destra dell'elenco servono per cambiare l'ordine delle immagini inserite. Il tasto "Library" ha la stessa funzione di "Aggiungi" ma apre direttamente una cartella di immagini predefinite.

Il campo **ColoreTrasparente** ha lo stesso significato che ha per il campo Etichetta. Il colore scelto come trasparente non sarà più un colore dell'immagine bitmap, verrà sostituito con ciò che c'è sotto (il colore dell'interno del pulsante). E' utile per eliminare il contorno rettangolare delle bitpmap importate e "fondere" in maniera

ottimale l’immagine sopra il pulsante (**256**, default, significa nessun colore trasparente).
Il campo **StileGrafico** permette di scegliere tra 3 tipi di visualizzazione:

4. **Nascosto**: il pulsante non è visibile sul terminale.
5. **Piatto**: pulsante in due dimensioni, piatto, senza effetto visivo alla pressione.
6. **3D**: pulsante in 3 dimensioni, con effetto visivo alla pressione sul terminale.

Azione: al campo **NumeroPagina** va associata la costante o la variabile che identifica la pagina che il terminale deve visualizzare alla pressione del tasto.
Comportamento pulsante: il pulsante può essere attivo (campo **Attivo True**) oppure non attivo (campo **Attivo False**) anche con riferimento alla **VariabileControllo** e ai valori di soglia (**AttivoOltreSoglia1** e **AttivoOltreSoglia2**).

Posizione: i campi **Larghezza** ed **Altezza** rappresentano le dimensioni del pulsante. Nel caso di immagine sovrapposta, le dimensioni devono essere tali da contenere la bitmap.

Soglia, **Soglia1**, **Soglia2**: il pulsante da la possibilità di cambiare, oltre ai colori della scritta e dello sfondo, anche il testo (**Testo_Soglia**) e l’immagine bitmap associata (**ListaImmagini_Soglia**), su eventi legati a due soglie (**Soglia1** e **Soglia2**) (come oggetto Campo Numerico, vedi par. 17.6.3).

In riferimento alla Tabella Proprietà di Figura 17.39 e Figura 17.40, l’oggetto “**PAGINA 1**” è un pulsante di cambio pagina, normalmente non attivo. Sarà attivo al verificarsi dell’evento legato a **Soglia1**, cioè quando il bit **3** della word **VW1** sarà uguale a **1**, e la scritta “**PAGINA 1**” apparirà in verde invece che in nero. Solo in questa condizione alla pressione del tasto il terminale visualizzerà la pagina **1** (**#1**).

17.6.6.2 Pulsante Set Bit

Questo pulsante opera esclusivamente su alcuni bit (maschera) di una variabile. La maschera è da intendersi nella sua rappresentazione binaria:
$38_{10} = 0010\ 0110_2$
bit 0 = 0
bit1 = 1
bit2 = 1
bit3 = 0
…
quindi un valore della maschera pari a 38 corrisponde a ad una modifica dei bit 1, 2 e 5.
La Figura 17.43 riporta il pulsante "**SET BIT**", e Figura 17.44 e Figura 17.45 la sua Tabella Proprietà.

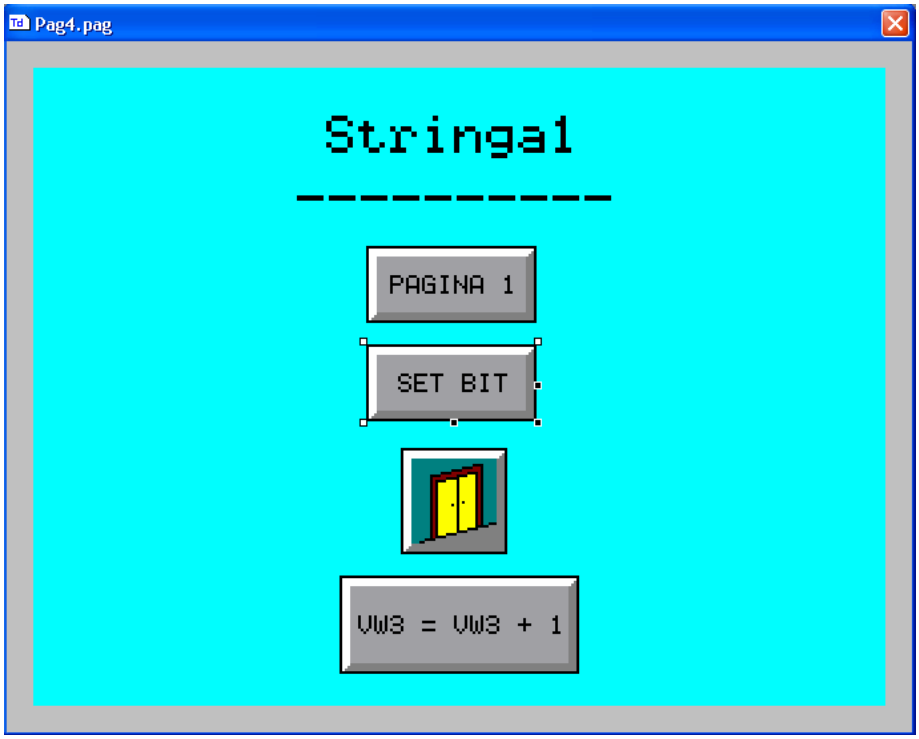


Figura 17.43: Oggetto Pulsante Set Bit "SET BIT" in pagina Pag4

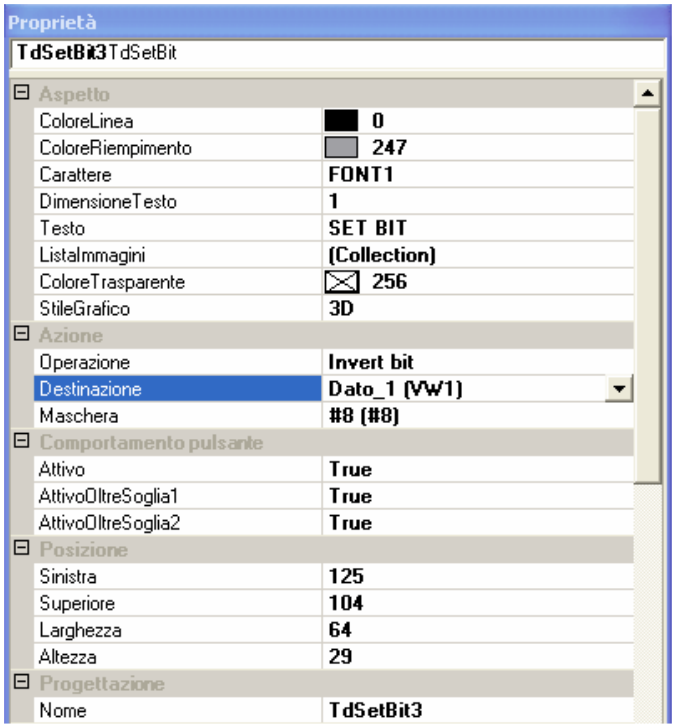


Figura 17.44: Prima parte Tabella Proprietà dell'oggetto Pulsante Set Bit "SET BIT" di pagina Pag4

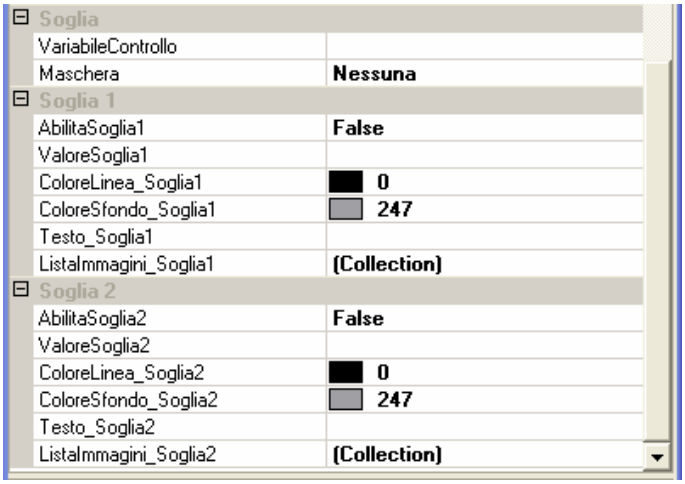


Figura 17.45: Seconda parte Tabella Proprietà dell'oggetto Pulsante Set Bit "SET BIT" di pagina Pag4

Le proprietà di questo oggetto sono del tutto simili a quelle del Pulsante Cambio Pagina (vedi par. 17.6.6.1), a parte chiaramente l'azione:

Azione: il campo **Operazione** ha le seguenti opzioni:

- **Set bit**: Imposta a 1
- **Reset bit**: Imposta a 0
- **Invert bit**: Inverte il bit

- **Set real-time**: Continua ad impostare a 1 finchè il pulsante è premuto
- **Reset real-time**: Continua ad impostare a 0 finchè il pulsante è premuto.

Al campo **Destinazione** va associata la variabile da modificare. **Maschera**, che va inserita in decimale, come costante o variabile, va sempre intesa nella su rappresentazione binaria.

In riferimento alla Tabella Proprietà di Figura 17.44 e Figura 17.45, l'oggetto "**SET BIT**" è un pulsante che, ogni volta premuto, inverte il bit **3** della word **VW1** (la maschera è la costante numerica decimale #8 = 8_{10} = 0000000000001000_2). E' sempre attivo , non è prevista alcuna variabile di controllo e le soglie sono disabilitate.

17.6.6.3 Pulsante Assegnamento Variabile

Questo tipo di pulsante viene utilizzato per due tipiche funzioni:

3. Assegnazioni del tipo X1 = X2
4. Operazioni matematiche del tipo X3 = X3 + X4

La Figura 17.46 riporta un Pulsante Assegnamento Variabile.

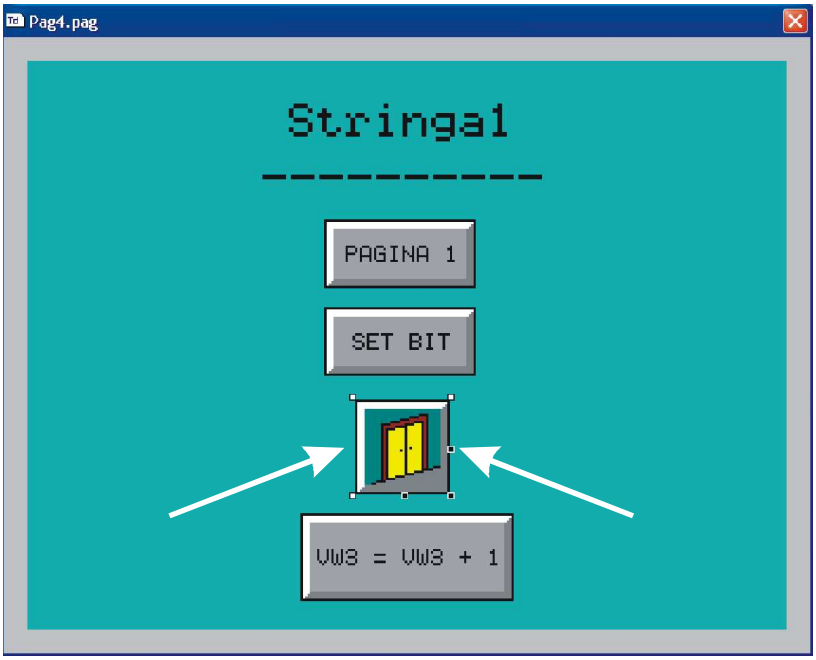


Figura 17.46: Oggetto Pulsante Assegnamento Variabile in pagina Pag4.

La Figura 17.47 e Figura 17.48 riportano la Tabella Proprietà dell'oggetto.

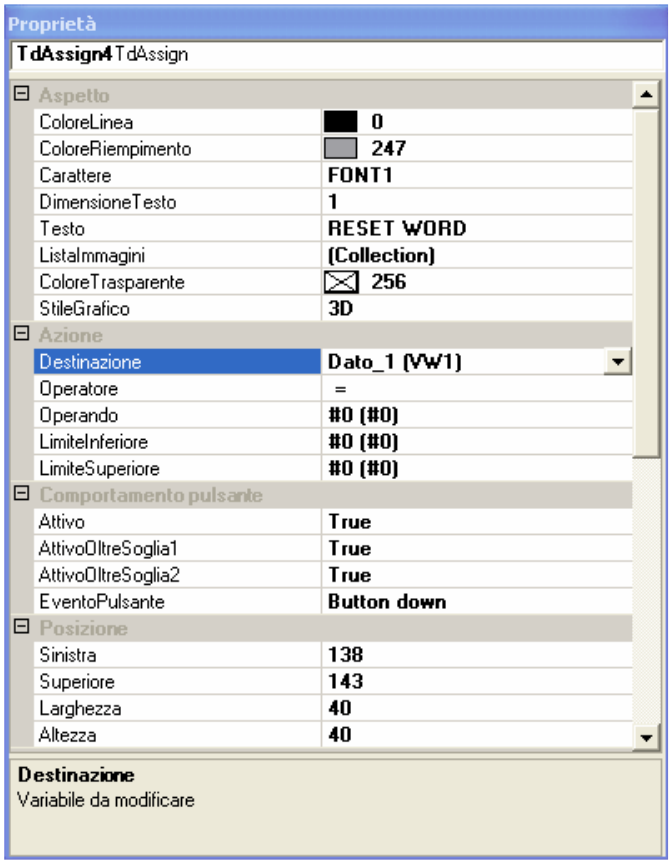


Figura 17.47: Prima parte Tabella Proprietà dell'oggetto Pulsante Assegnamento Variabile di pagina Pag4

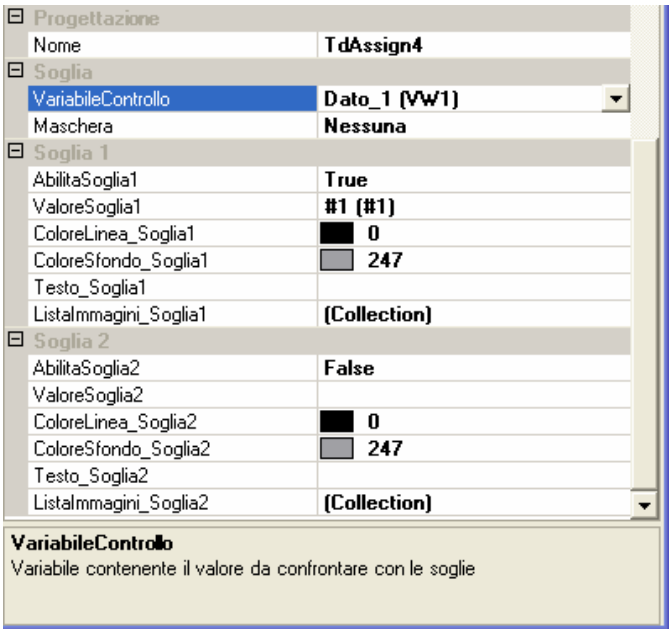


Figura 17.48: Seconda parte Tabella Proprietà dell'oggetto Pulsante Assegnamento Variabile di pagina Pag4

Le proprietà di questo oggetto sono del tutto simili a quelle del Pulsante Cambio Pagina (vedi par. 17.6.6.1) e del Pulsante Set Bit (vedi par 17.6.6.2) a parte chiaramente l'azione:

Azione: il campo **Destinazione** contiene l'operando destinatario dell'operazione, cioè la variabile da modificare.
Operatore: "**+**", "**-**" per operazioni di incremento e decremento, "**=**" per le assegnazioni.
Operando: è l'operando sorgente dell'operazione, che può essere una costante o una variabile.
LimiteIferiore e **LimiteSuperiore** sono i limiti entro cui deve mantenersi il risultato dell'operazione di assegnamento.

Per la formula **X1 = X1 + #1**
Destinazione: **X1**
Operatore: **+**
Operando: **#1**

Per la formula **X2 = #0**
Destinazione: **X2**
Operatore: **=**
Operando: **#0**

Comportamento pulsante: il campo **EventoPulsante** definisce il tipo di pressione del tasto sul terminale che ne determina l'attivazione. Sono 3 le tipologie:

- **Button down**: l'operazione viene eseguita una sola volta alla pressione del tasto.
- **Button up**: l'operazione viene eseguita una sola volta al rilascio del tasto.
- **Autorepeat**: l'operazione viene eseguita più volte con frequenza crescente all'aumentare del tempo di pressione del tasto.

In riferimento alla Tabella Proprietà di Figura 17.47 e Figura 17.48, l'oggetto indicato con le frecce in Figura 17.46 è un pulsante, sempre attivo, che alla pressione sul terminale esegue

l'operazione **VW1 = 0**, assegna cioè il valore **0** alla variabile **VW1**. Se la variabile **VW1** è minore (<) di **1**, l'immagine sopra il pulsante è la "porta chiusa" (l'immagine **Door_1_Close.bmp** è caricata dalla cartella **Library** nel campo **ListaImmagini**). Se la variabile **VW1** è maggiore o uguale (≥) a **1**, l'immagine sopra il pulsante è la "porta aperta" (l'immagine **Door_1_Open.bmp** è caricata dalla cartella Library nel campo **ListaImmagini_Soglia1**).

La Figura 17.49 riporta un altro Pulsante Assegnamento Variabile, "**VW3 = VW3 + 1**", Figura 17.50 e Figura 17.51 la sua Tabella Proprietà.

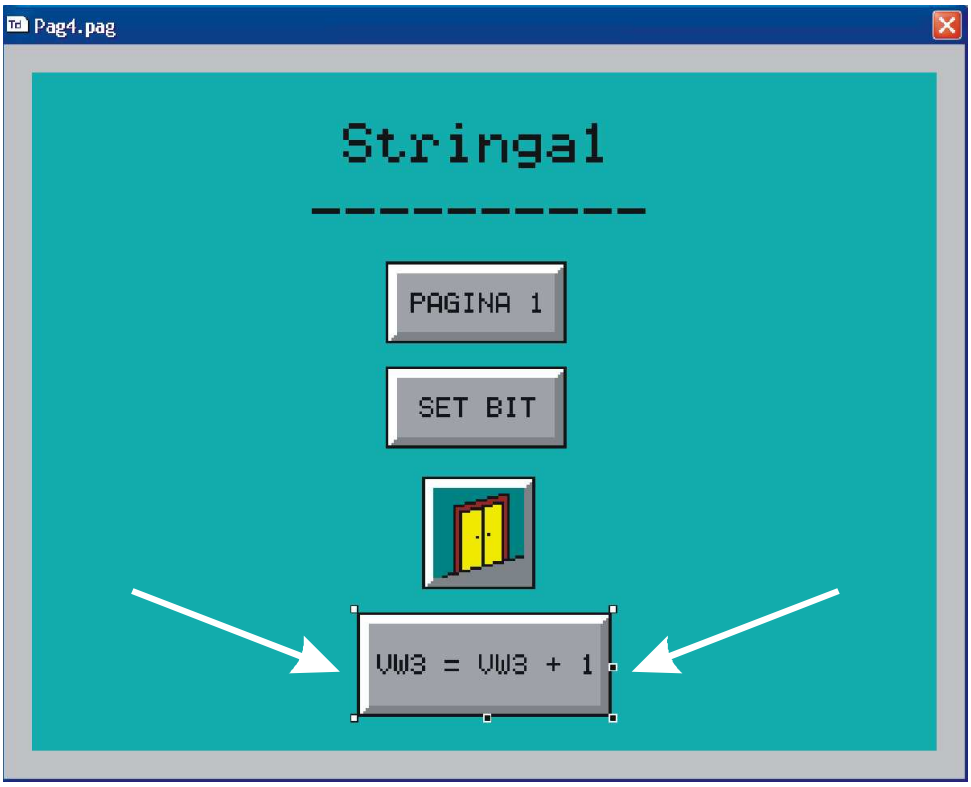


Figura 17.49: Oggetto Pulsante Assegnamento Variabile "VW3 = VW3 + 1" in pagina Pag4.

L'oggetto indicato è un pulsante che esegue l'operazione corrispondente al testo che riporta, **VW3 = VW3 + 1**. E' sempre attivo, entro i limiti riportati (l'operazione viene eseguita se la variabile **VW3** è compresa entro i limiti riportati, **0** e **5**), ed i controlli sulle soglie non sono abilitati.

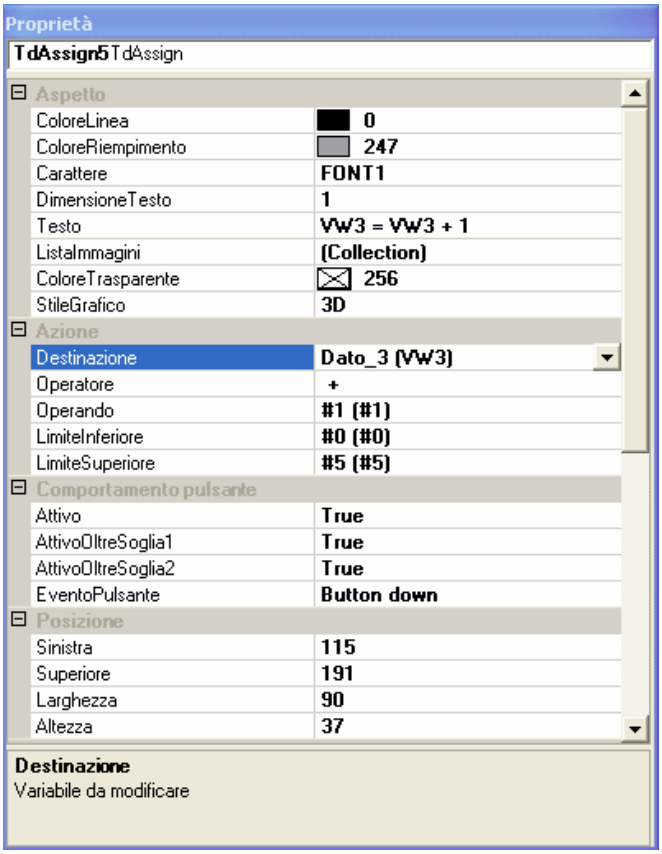


Figura 17.50: Prima parte Tabella Proprietà dell'oggetto Pulsante Assegnamento Variabile “VW3 = VW3 + 1” di pagina Pag4

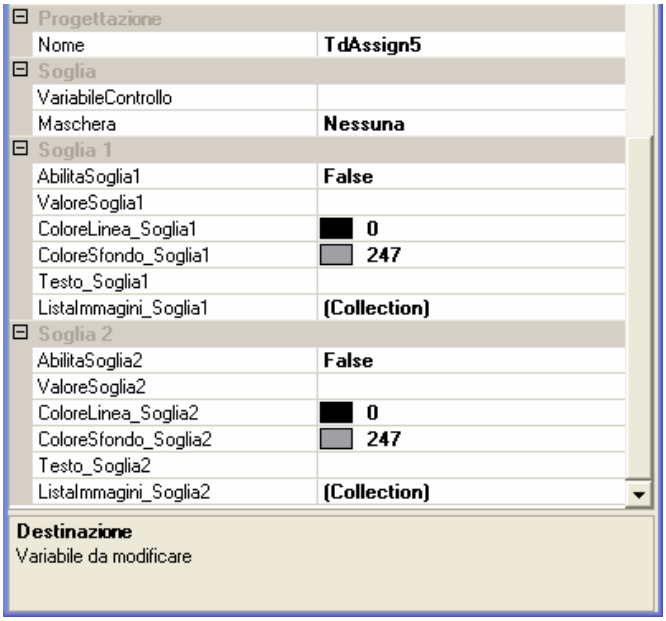


Figura 17.51: Seconda parte Tabella Proprietà dell'oggetto Pulsante Assegnamento Variabile “VW3 = VW3 + 1” di pagina Pag4

17.6.7 Oggetto Bitmap

Per visualizzare un'immagine a partire da un file in formato bitmap, selezionare il pulsante corrispondente nella Casella Strumenti e fare click sulla finestra che rappresenta la pagina. Verrà visualizzata un'immagine come quella riportata in Figura 17.52

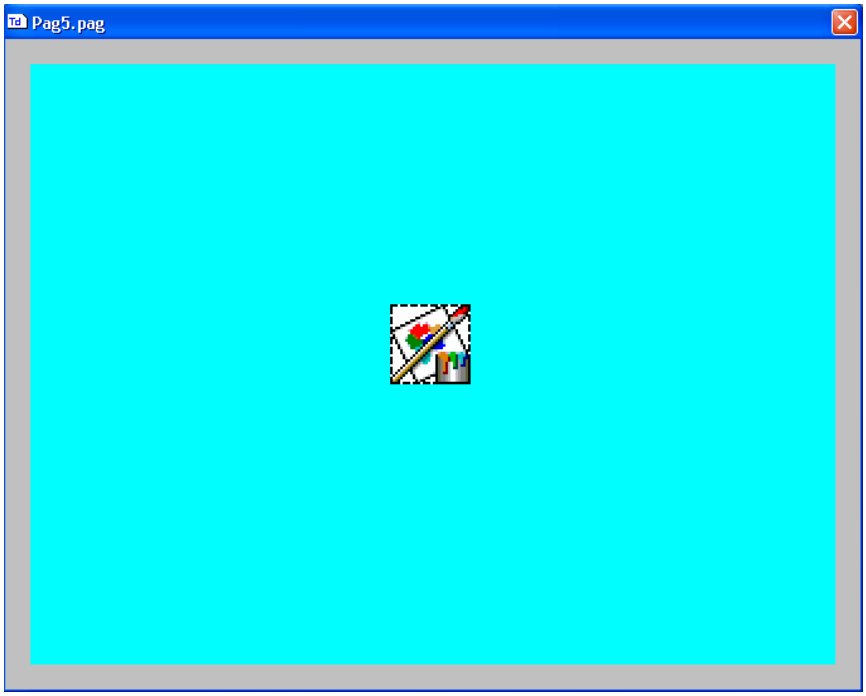


Figura 17.52: Oggetto Bitmap in paginaPag5.

Per sostituire questa immagine preliminare con una voluta, è necessario aprire l'apposito editor di insieme, come mostrato in Figura 17.53

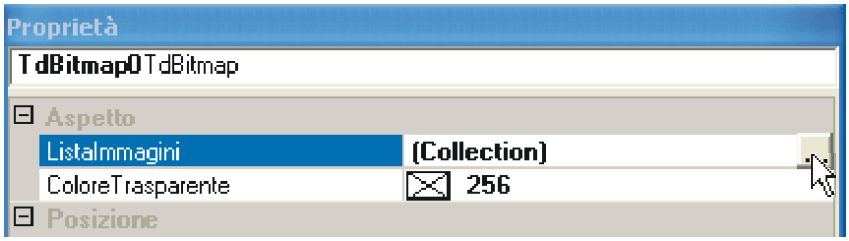


Figura 17.53: apertura dell'editor per ListaImmagini dell'oggetto Bitmap.

L’editor è uguale a quello di Figura 17.42. L’immagine preliminare si chiama **Bitmap**, e finchè non verrà rimossa (pulsante Remove dentro l’editor) o spostata dalla prima posizione, l’immagine voluta non potrà essere visualizzata.
Dopo aver selezionato un’immagine (l’immagine **Link_Login.bmp** è caricata dalla cartella **Library**), l’editor si presenta così:

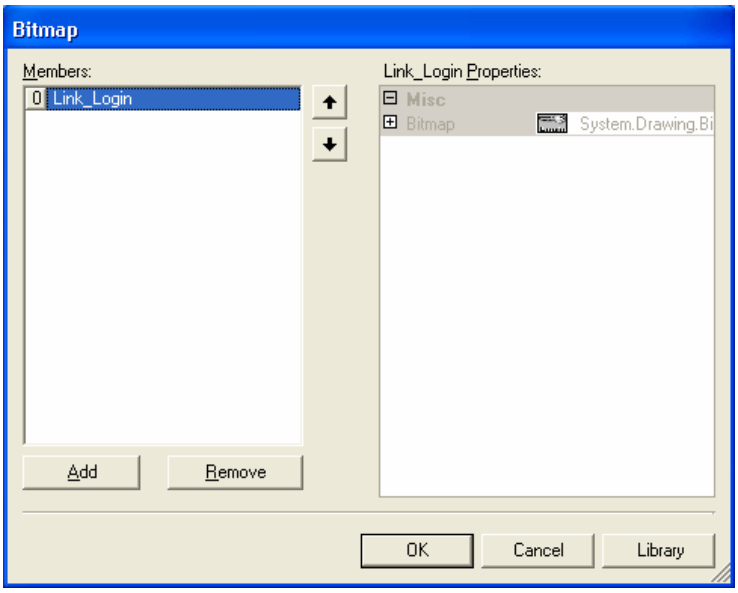


Figura 17.54: Editor per ListaImmagini dell’oggetto Bitmap.

La nuova immagine caricata è visibile in Figura 17.55.

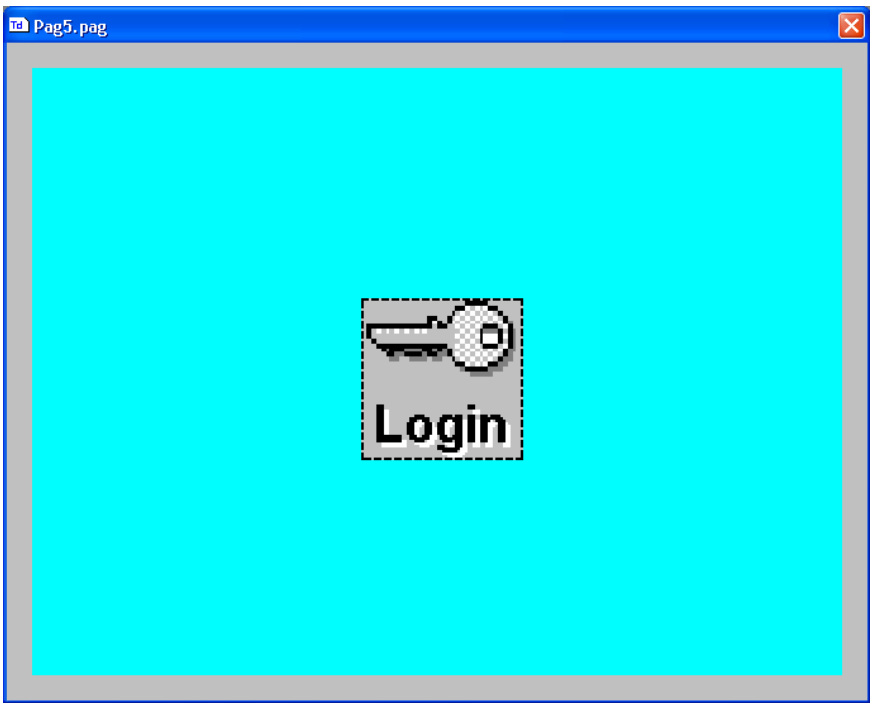


Figura 17.55: Oggetto Bitmap in pagina Pag5.

La Figura 17.56 riporta la Tabella Proprietà dell'oggetto.

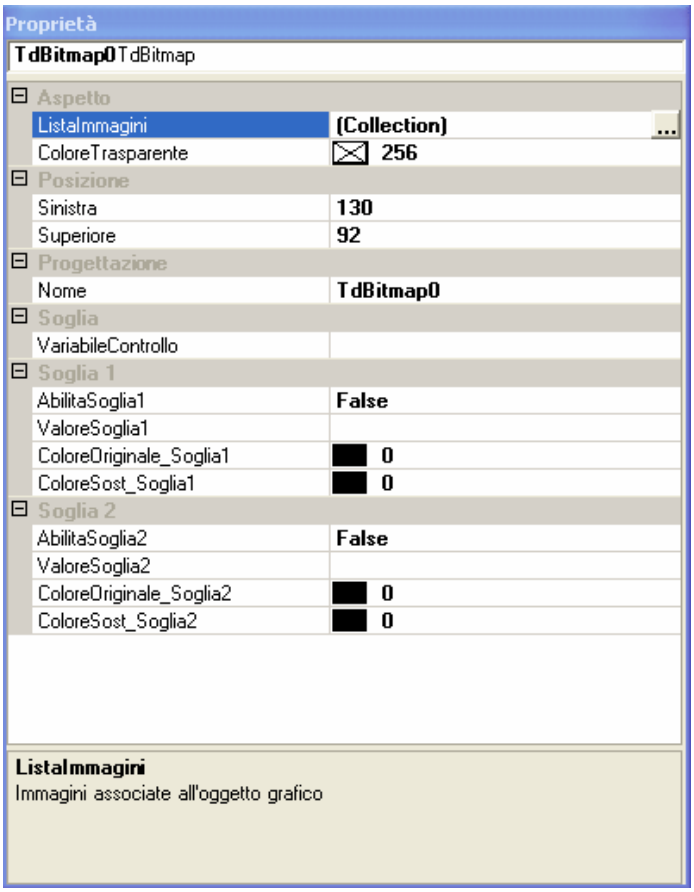


Figura 17.56: Tabella Proprietà dell'oggetto Bitmap in pagina Pag5.

Aspetto: il campo **ColoreTrasparente** ha lo stesso significato che ha per gli oggetti Pulsante (vedi par. 17.6.6).

Tutti gli altri campi sono già stati analizzati per altri oggetti. Anche per l'oggetto Bitmap è possibile utilizzare una **VariabileControllo** e due soglie.
E' possibile sostituire un colore (**ColoreOriginale_Soglia**) con un altro (**ColoreSost_Soglia**) al verificarsi delle condizioni imposte dalla variabile di controllo, con le stesse modalità descritte per gli altri oggetti.
La Figura 17.55 mostra un oggetto bitmap senza colore trasparente e senza variabili di soglia imposte.

17.6.8 Oggetto Campo Simbolico

A partire da un elenco di bitmap, visualizza un'immagine indicizzata dal valore di una variabile. L'inserimento delle immagini avviene come nel caso dell'oggetto Bitmap.
Se il campo **Input** è impostato **True** l'operatore potrà anche scegliere l'immagine da visualizzare.
Per importare questo oggetto sulla pagina occorre ridimensionarlo con il trascinamento del mouse o, successivamente, dalla Finestra Proprietà.
La Figura 17.57 mostra un oggetto Campo Simbolico.

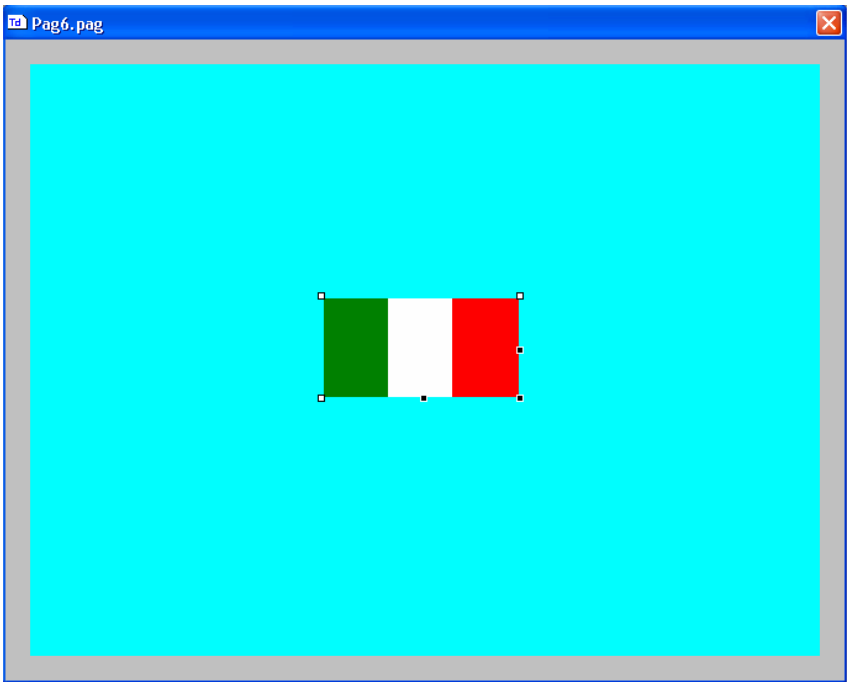


Figura 17.57: Oggetto Campo Simbolico in pagina Pag6

L'editor di inserimento delle immagini è quello di Figura 17.58; la Tabella Proprietà è quella riportata in Figura 17.59. In questo caso l'oggetto è una lista di 4 immagini (4 bandiere di altrettanti stati), indicizzata dalla variabile **VW4**, non modificabile dall'utente.

Aspect: il campo **RipetiLista** va ricondotto al numero di lingue selezionate per il progetto. Se impostato come **True**, la lista delle immagini inserite sarà duplicata automaticamente per ciascuna

delle lingue di progetto. Nel caso in esame, le lingue sono 2 (vedi par. 17.5.1), quindi si avranno le stesse 4 immagini per entrambe.

Lingua1: Italiano
VW4 = 0 FlagIT.bmp
VW4 = 1 FlagGB.bmp
VW4 = 2 FlagFR.bmp
VW4 = 3 FlagESP.bmp

Lingua2: Inglese
VW4 = 0 FlagIT.bmp
VW4 = 1 FlagGB.bmp
VW4 = 2 FlagFR.bmp
VW4 = 3 FlagESP.bmp

Se invece il campo **RipetiLista** fosse impostato come **False** si dovrebbe inserire una lista di immagini per ogni lingua di progetto. Nel nostro esempio, cambiando solo il campo **RipetiLista**, si avrebbero 2 immagini per ogni lingua selezionata.

Lingua1: Italiano
VW4 = 0 FlagIT.bmp
VW4 = 1 FlagGB.bmp

Lingua2: Inglese
VW4 = 0 FlagFR.bmp
VW4 = 1 FlagESP.bmp

Aspetto: i campi **ListaImmagini** e **ColoreTrasparente** sono identici a quelli del campo Bitmap (vedi par. 17.6.7).

Input: se il campo **AccettaInput** è **True**, l'operatore può scegliere quale immagine visualizzare nei limiti impostati in **MinimoIndice** e **MassimoIndice**.

Memoria: nel campo **Indice** va collegata la variabile che indicizza la lista di immagini.

Il campo **BitMask** è la maschera che determina lo scorrimento delle immagini Se selezionata come **Nessuna**, ad ogni immagine è associato un valore della variabile Indice, inteso come numero decimale (0, 1, 2,..., N-1, dove N è il numero delle immagini nella lista). E' il caso dell'esempio riportato in Figura 17.57. Se selezionata come **bitN**, la selezione è possibile solo tra due immagini: l'immagine in posizione **0** (vedi Figura 17.58) se il bit **N** è uguale a **0**, l'immagine in posizione **1** se il bit **N** è uguale a **1**. Le altre immagini eventualmente presenti nella lista verranno ignorate.

I campi **Posizione** e **Progettazione** sono identici a quelli del campo Bitmap (vedi par. 17.6.7).

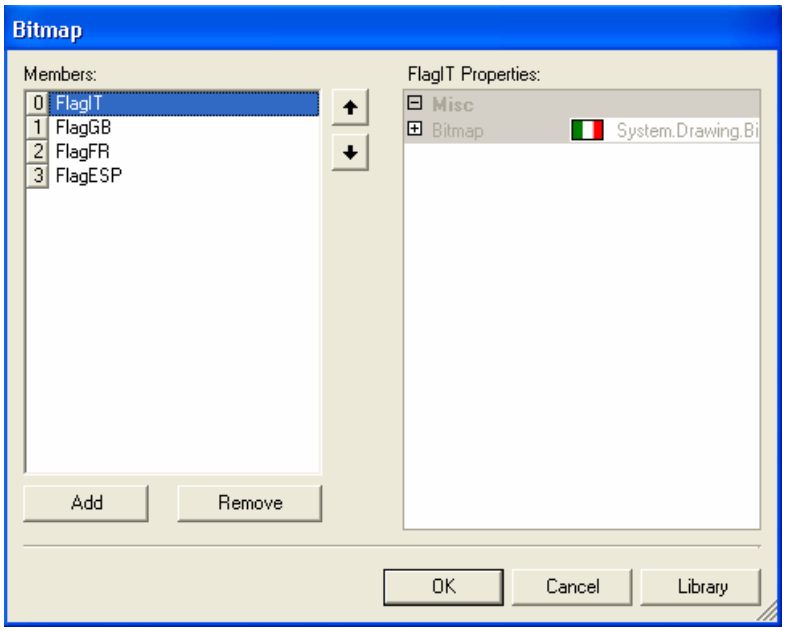


Figura 17.58: Editor per ListaImmagini dell'oggetto Campo Simbolico.

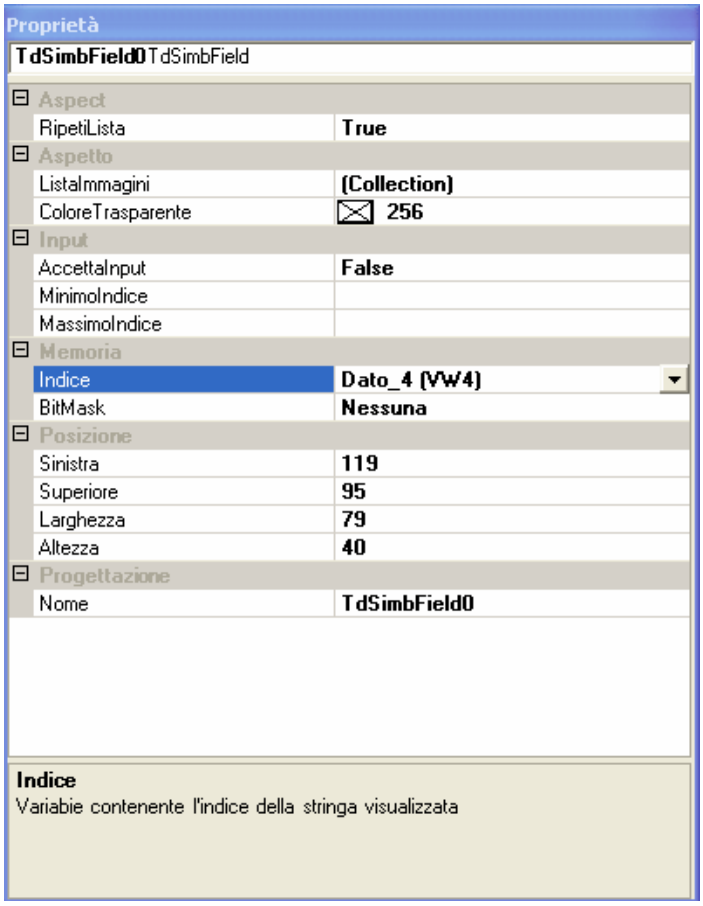


Figura 17.59: Tabella Proprietà dell'oggetto Campo Simbolico di pagina Pag6

17.7 Compilazione e trasferimento del programma

L'ultima fase del lavoro di configurazione consiste nel trasferimento del programma nella memoria del terminale grafico. Come già accennato all'inizio del capitolo, lo sviluppo di un'applicazione richiede l'utilizzo contemporaneo di TdDesigner e PlProg. In particolare, **l'operazione di compilazione** (da menu "File/Compila" o da icona) **deve essere fatta con PlProg aperto** per dare modo a TdDesigner di modificare le impostazioni grafiche del file .plp. Si consiglia a tal proposito una lettura del manuale relativo al terminale che contiene una trattazione dettagliata dell'interazione tra logica e grafica (cap. 4).

Vale anche la pena di sottolineare che, contrariamente a quanto accade per PlProg, TdDesigner salva automaticamente il progetto compilato. Se si salva anche il file .plp questo conterrà tutte le informazioni necessarie per ogni successiva programmazione, grafica inclusa.

In Figura 17.60 e Figura 17.61 sono schematizzate le procedure con la sequenza corretta delle operazioni rispettivamente per la creazione di un nuovo progetto e per l'utilizzo e l'eventuale modifica di un progetto già esistente.

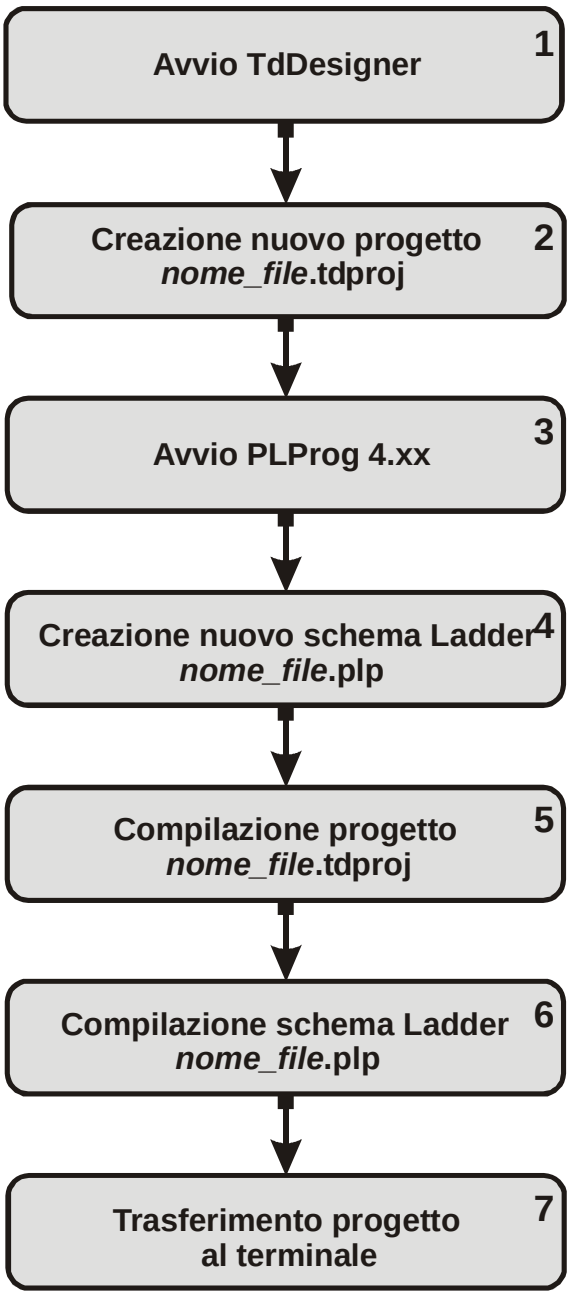


Figura 17.60: creazione di un nuovo progetto.

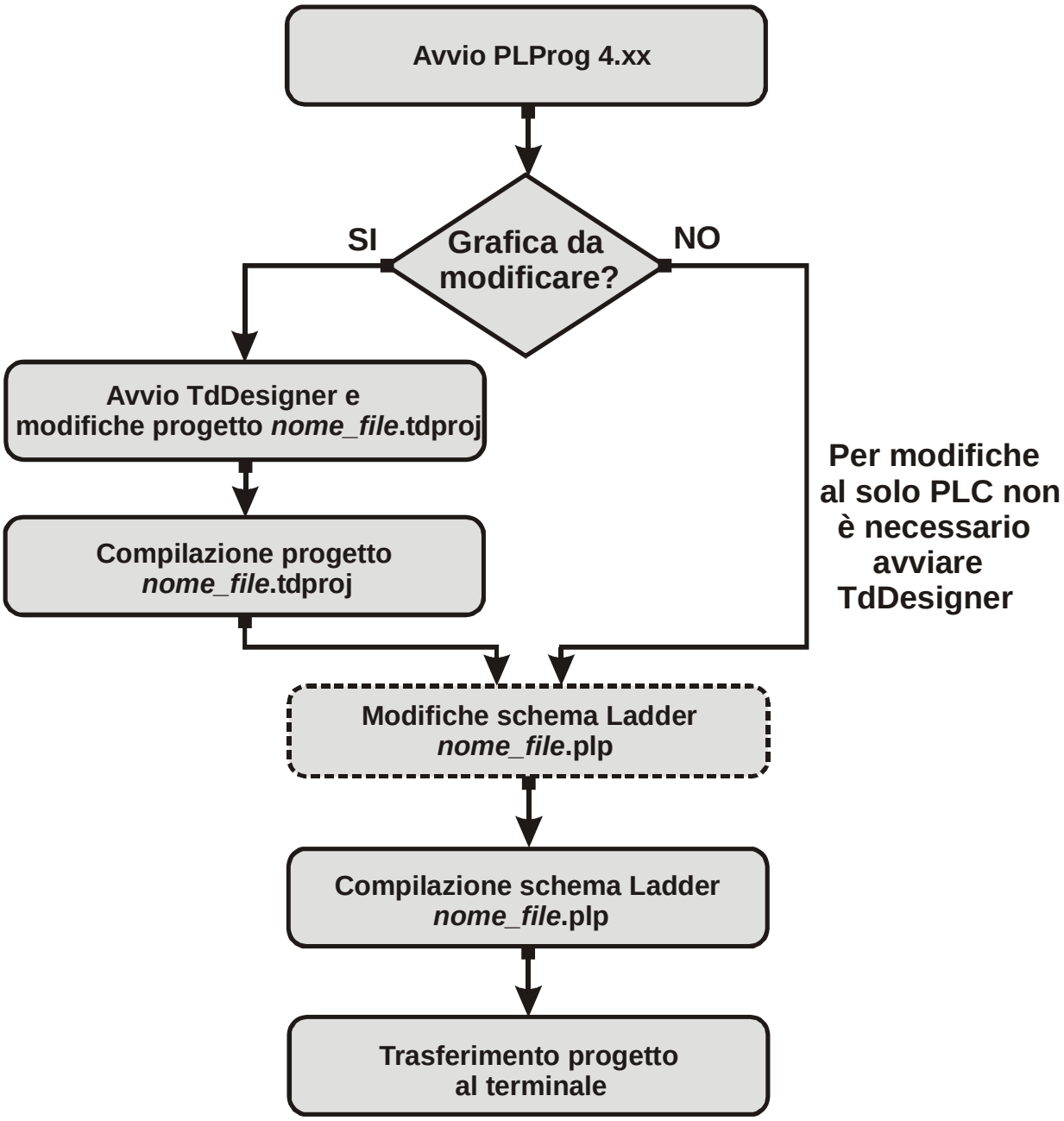


Figura 17.61: utilizzo ed eventuale modifica di un progetto già esistente.

18 Glossario

Boot: programma residente nella memoria del PLC, gestisce il caricamento del firmware. Non è accessibile all'utente finale.

Compilazione: nel caso specifico di PLProg, con questo termine si indica il processo che traduce il Diagramma Ladder sviluppato dall'utente in un formato adatto ad essere trasmesso al PLC.

Debug: azione volta a individuare errori nella logica di un programma. PLProg dispone di una modalità di debug attraverso la quale è possibile visualizzare lo stato di contatti e bobine durante il funzionamento del PLC direttamente sul diagramma ladder. Non si tratta di una simulazione, il programma deve essere in esecuzione sul PLC.

Editor: programma per la scrittura e la memorizzazione di documenti. L'editor di PLProg mette a disposizione gli strumenti per la creazione di diagrammi ladder.

Firmware: programma che gestisce il funzionamento del PLC.

Ladder: abbreviazione di "Ladder Diagram" (Diagramma a scala). E' un linguaggio di programmazione a contatti derivato dai disegni dei sistemi di controllo realizzati con relé elettromeccanici.
Si basa sui concetti di contatto e bobina; inizialmente era in grado di gestire solamente funzioni di logica binaria ; in seguito è stato esteso per trattare anche numeri interi e/o reali.

Memory-Card: piccola scheda elettronica dotata di memoria EEPROM sulla quale è possibile memorizzare un programma per PLC.

Modbus: protocollo di comunicazione seriale usato tipicamente dai sistemi di supervisione.

Special Marker: area di memoria che contiene tutti i dati necessari al programma ladder per interagire con l'hardware del PLC; tra questi, ad esempio, ci sono i valori degli ingressi

analogici e dei trimmer, i conteggi e i set degli encoder, le impostazioni per la comunicazione attraverso le porte seriali.

Variabile: Il programma in esecuzione sul PLC può trasferire dati velocemente da e per determinate locazioni di memoria RAM. Queste locazioni sono accessibili a livello di diagramma ladder e sono comunemente chiamate variabili. L'utente dovrà scegliere il tipo di variabile da impiegare a seconda del dato che intende trasferire, ad esempio se si vuole memorizzare nell'area V un intero maggiore di 32767 è necessario usare una variabile VD (V Double Word).

19 Aggiornamenti / Updates

PIXSYS
Via Tagliamento, 18
30030 Mellaredo di Pianiga (VE)
www.pixsys.net
e-mail: sales@pixsys.net - support@pixsys.net

Software Rev. 4.63

2300.10.057-RevE 050907