

---

# Getting started

*Creating a simple Siemens MPI protocol application*

## Document revision

Date	Edition	Comments
24/11/2009	1.0	-
28/10/2010	2.0	-

**Sielco Sistemi srl**  
via Roma, 24  
I-22070 Guanzate (CO)  
<http://www.sielcosistemi.com>

## 1. Introduction

In this guide we propose, as an example, the design of a simple supervision application Siemens MPI protocol based; this example is a little step towards the design of more complex SCADA applications, but it can be useful for anybody who approaches for the first time to a SCADA, and in particular to **Winlog Pro** software, to quickly understand how to communicate with external devices (Siemens Simatic S7-200/300/400 PLCs and VIPA PLCs).

Every time you design a new application, it is necessary to know, for each external device, the communication protocol, the address and the list of variables that you want read or write.

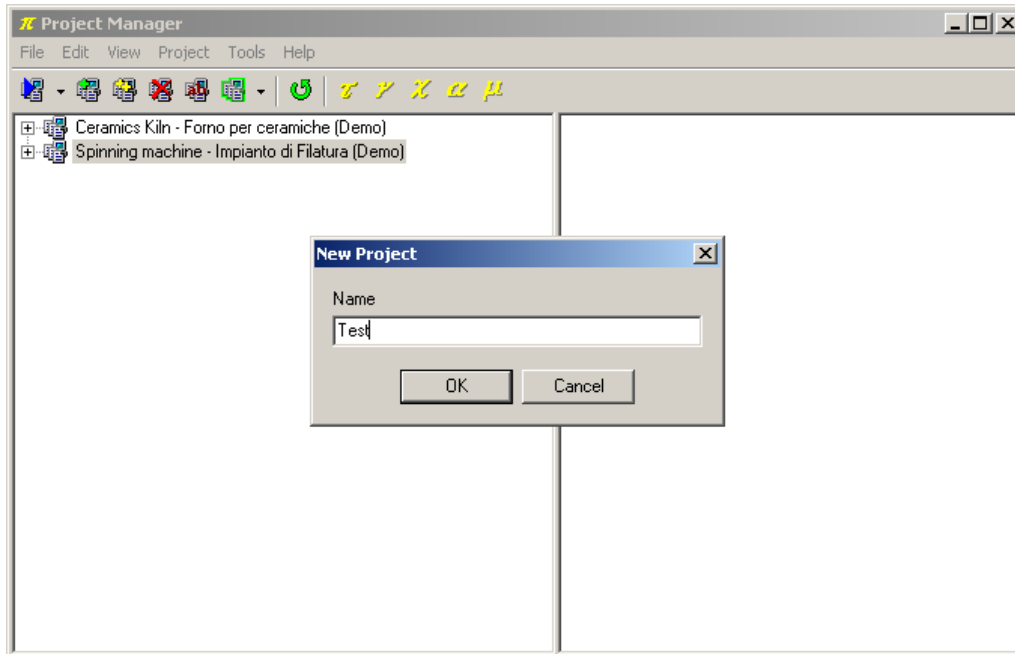
In our example we have to communicate using Siemens MPI protocol with two devices (Test Device#1 e Test Device#2) whose address are 1 and 2; for each device we want to read 3 numeric variables (Temp, Sp e Out) and 1 digital variable (Alarm).

## 2. Creating the project

To create a new supervision project, it is necessary to use **Project Manager**, the **Winlog Pro** integrated development environment that provides different tools (**Gate Builder**, **Template Builder**, **Code Builder**).

Run **Project Manager** selecting own icon from Start menu.

Select New from **Project** menu and insert the project name (for example `Test`).



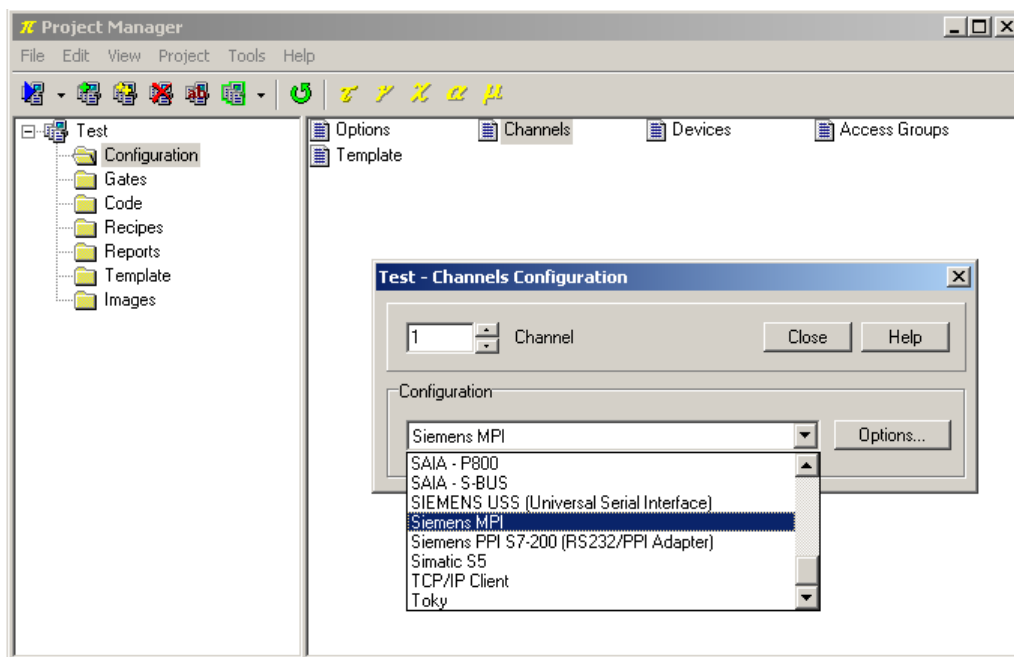
*Project creation*

In this way you create a tree structure with all supervision project elements.

### 3. Communication channel configuration

From elements in Configuration folder select Channels.

Define the logic channel 1 to communicate using Siemens MPI protocol with Siemens S7-200/300/400 PLCs and VIPA PLCs.

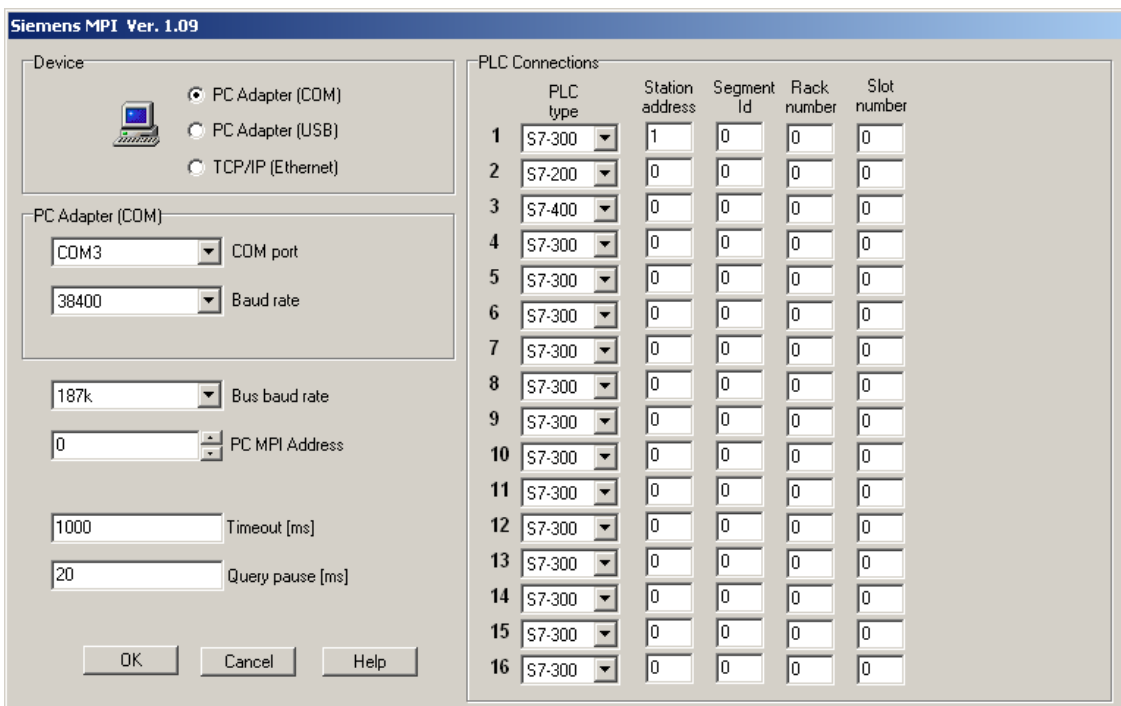


*Protocol selection*

Press button Options... to enter channel configuration.

In the left side of the window, PC communication parameters are defined; select the serial port to assign to the channel (for example COM3) and related Baud rate. Set adapter Bus Baud, PC MPI Address Rate, Time out, Query Pause.

In the left side of the window, PLC communication parameters are defined; PLC type (S7-200, S7-300, S7-400), Station address, Segment id, Rack number, Slot number (for these parameters refer to manufacturer data device, an example is provided in the figure)



*Channel and adapter configuration with the PC Adapter (COM and USB)*

**N.B. The MPI address of PC (PC MPI Address) and the MPI address of PLC (Station address) must be different from each other e different from any other MPI device in the network (es. Operator panel). And the MPI address of PLC (Station address) sarà il numero del dispositivo (Dispositivo/Device) will be the device number to specify at the creation of variables database (Gate builder see par 5.1)**

If you want to communicate with the PLC, using the TCP / IP over Ethernet support, you must also configure the PC communication indicating the IP address of the PC from those available (Client IP Address) and the TCP / IP port (TCP Port Number). Configure the PLC communication indicating, in addition to the parameters already seen, the IP address of the PLC (IP Station address)

The screenshot shows the Siemens MPI Ver. 1.09 configuration window. On the left, the 'Device' section has three radio buttons: 'PC Adapter (COM)', 'PC Adapter (USB)', and 'TCP/IP (Ethernet)', with the last one selected. Below this, the 'TCP/IP (Ethernet)' section contains fields for 'Client IP addresses' (192.168.0.114), 'Port number' (102), 'Bus baud rate' (187k), 'PC MPI Address' (0), 'Timeout [ms]' (1000), and 'Query pause [ms]' (20). On the right, the 'PLC Connections' table lists 16 connections with columns for PLC type, Station address, Segment Id, Rack number, Slot number, and IP station address.

	PLC type	Station address	Segment Id	Rack number	Slot number	IP station address
1	S7-300	1	0	0	0	192.168.0.100
2	S7-200	0	0	0	0	255.255.255.255
3	S7-400	0	0	0	0	255.255.255.255
4	S7-300	0	0	0	0	255.255.255.255
5	S7-300	0	0	0	0	255.255.255.255
6	S7-300	0	0	0	0	255.255.255.255
7	S7-300	0	0	0	0	255.255.255.255
8	S7-300	0	0	0	0	255.255.255.255
9	S7-300	0	0	0	0	255.255.255.255
10	S7-300	0	0	0	0	255.255.255.255
11	S7-300	0	0	0	0	255.255.255.255
12	S7-300	0	0	0	0	255.255.255.255
13	S7-300	0	0	0	0	255.255.255.255
14	S7-300	0	0	0	0	255.255.255.255
15	S7-300	0	0	0	0	255.255.255.255
16	S7-300	0	0	0	0	255.255.255.255

*Channel configuration (Ethernet)*

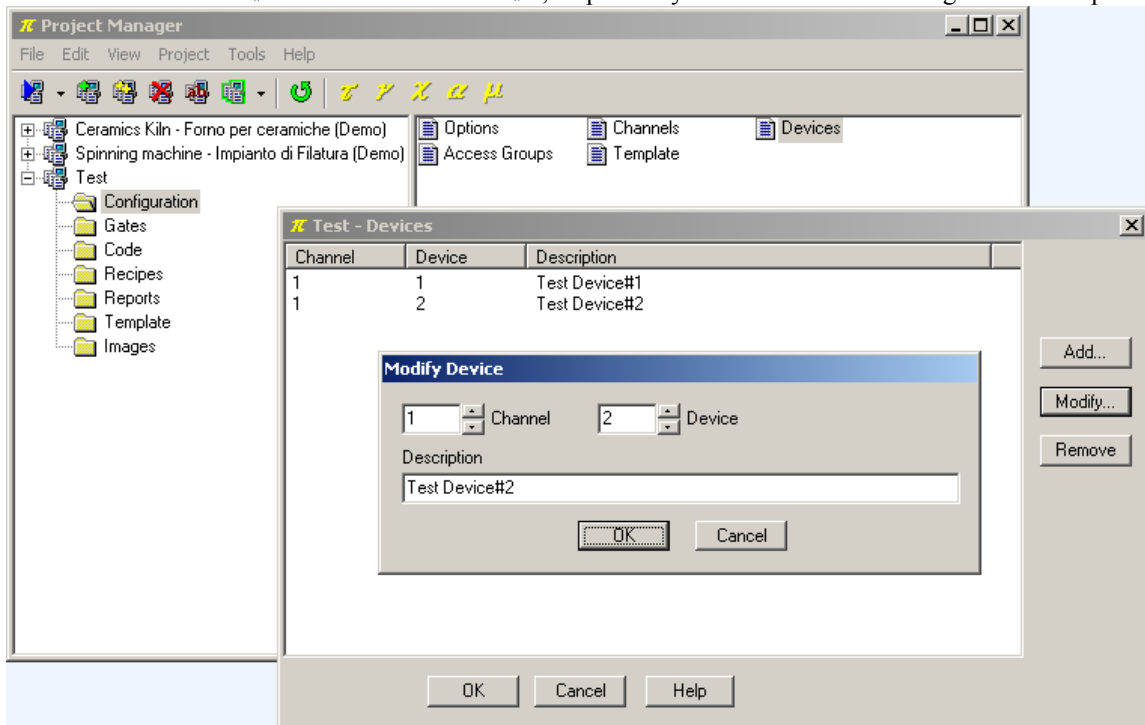
Communication between Personal Computer and PLC is possible using the following devices:

- via COM port using SIEMENS SIMATIC S7 - PC Adapter V5.1 - Code **6ES7 972-0CA23-0XA0**
- - via USB port using SIEMENS SIMATIC S7 - PC Adapter USB - Code **6ES7 972-0CB20-0XA0**
- - via ethernet card.

## 4. Devices declaration

From elements in Configuration folder select Devices.

Insert Test Device#1 and Test Device#2, respectively at address 1 and 2 on logic channel 1 previously set



*Devices declaration*

## 5. Creating variables database

Now we can insert the devices variables in gates database

In this example we only consider numeric and digital variables (gates).

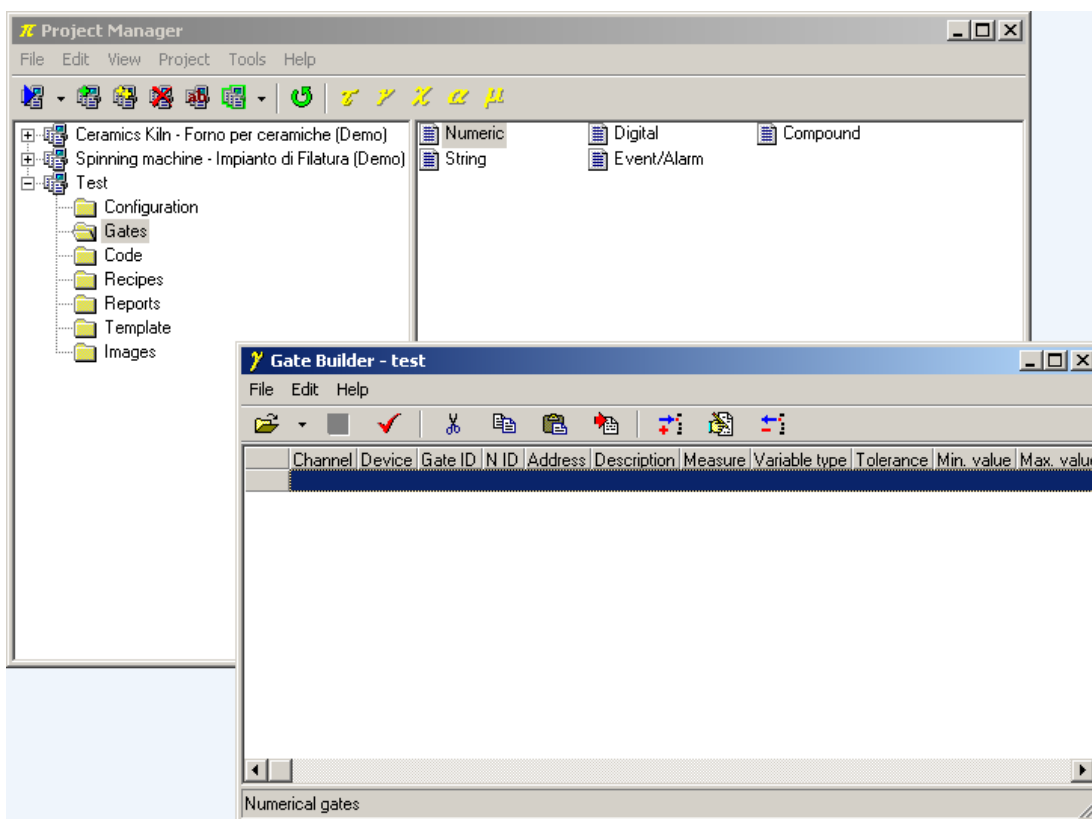
Numeric gates include all those variables that refer to an analog quantity (for example measured variables, set-points, alarm threshold ..) and can be expressed by a byte, a word, a double word, an integer or by a floating-point variable.

Digital gates include all those variables that refer to digital status (for example an alarm conditions, a configuration option, ...) and can be expressed by a single bit.

Sometimes more digital conditions can be gathered in a single numeric variable, but this case will not be explained in this example.

To edit the variables database, you need to run **Gate Builder**

From **Project Manager**, select Gates folder and double-click on each of icons (Numeric, Digital, ...).



*Variables database creation*

Suppose you need to read the following variables (to adapt this example to a real case it is enough to modify the gates details below).

Name	Channel	Device	MPI address*	Variable type	Gate type	Unit	Description
TEMP	1	1	(DB5) Word 5	Signed Word	Numeric	°C	Temperature - Measure
SP	1	1	(DB5) Word 10	Signed Word	Numeric	°C	Temperature - Setpoint
OUT	1	1	(DB6) Word 15	Unsigned Word	Numeric	%	Control Output - Value
ALARM	1	1	(DB8) Byte 12 Bit 1 Bit	Bit	Digitale		Internal alarm status
TEMP	1	2	(DB5) Word 5	Signed Word	Numeric	°C	Temperature - Measure
SP	1	2	(DB5) Word 10	Signed Word	Numeric	°C	Temperature - Setpoint
OUT	1	2	(DB6) Word 15	Unsigned Word	Numeric	%	Control Output - Value
ALARM	1	2	(DB8) Byte 12 Bit 1 Bit	Bit	Digital		Internal alarm status

\* The code between brackets represents the DB to which Word, Byte and Bit used to read the variable gather (refer to protocol manual in Project Manager Help).



## 5.1 Numeric variables configuration

Repeat numeric gates configuration (PLC Siemens S7300/400 DB area) for both devices, having care to change device number (Device) and N ID.

### *Numeric variable TEMP configuration*

The screenshot shows the 'Numerical gates' dialog box with the 'General' tab selected. The 'Gate ID' is set to 'Temp'. The 'N ID' is set to '1'. The 'Record on DB' checkbox is checked, and the 'Writing enabled' checkbox is unchecked. The 'Description' field contains 'Temperature - PV - Measured value - Test Device#1'. The 'Access groups' field is empty, with a 'Choose...' button next to it. The 'Ok', 'Cancel', and 'Help' buttons are at the bottom right.

*TEMP numeric variable configuration – General folder*

The screenshot shows the 'Numerical gates' dialog box with the 'Sampling' tab selected. The 'Channel' is set to '1'. The 'Device' is set to '1'. The 'Address' is set to 'DB5.DBW5'. The 'Sample' is set to 'Always'. The 'Sample freq. [Sec.]' is set to '1'. The 'Read block' and 'Write block' fields are empty. The 'Ok', 'Cancel', and 'Help' buttons are at the bottom right.

*TEMP numeric variable configuration – Sampling folder*

**Numerical gates**

General | Sampling | Value | Tolerance

0 Min. value

0 Max. value

0 Start value

1 Decimal digits

°C Measure

S\_WORD Variable type

Conversion factor

1 Measured val. 1

1 Engineering val. 1

1 Measured val. 2

1 Engineering val. 2

Ok Cancel Help

*TEMP numeric variable configuration – Value folder*

**Numeric variable SP configuration**

**Numerical gates**

General | Sampling | Value | Tolerance

Sp Gate ID  Record on DB

1 NID  Writing enabled

Description

Temperature - SP- Setpoint value - Test Device#1

Access groups

Choose...

Ok Cancel Help

*SP numeric variable configuration – General folder*

**Numerical gates**

General | Sampling | Value | Tolerance

1 Channel  Read block

1 Device  Write block

DB5.DBW10 Address

Always Sample

1 Sample freq. [Sec.]

Ok Cancel Help

*SP numeric variable configuration – Sampling folder*

**Numerical gates**

General | Sampling | Value | Tolerance

0 Min. value

0 Max. value

0 Start value

1 Decimal digits

°C Measure

S\_WORD Variable type

Conversion factor

1 Measured val. 1

1 Engineering val. 1

1 Measured val. 2

1 Engineering val. 2

Ok Cancel Help

*SP numeric variable configuration – Value folder*

### *Numeric variable OUT configuration*

**Numerical gates**

General | Sampling | Value | Tolerance

Out Gate ID  Record on DB

1 N ID  Writing enabled

Description

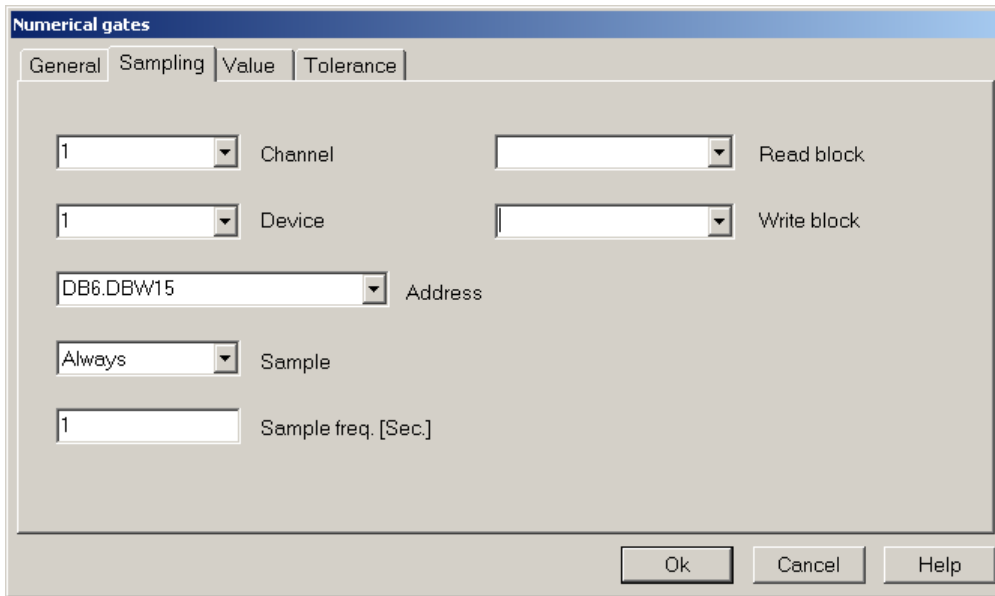
Control Output - OP - Value - Test Device#1

Access groups

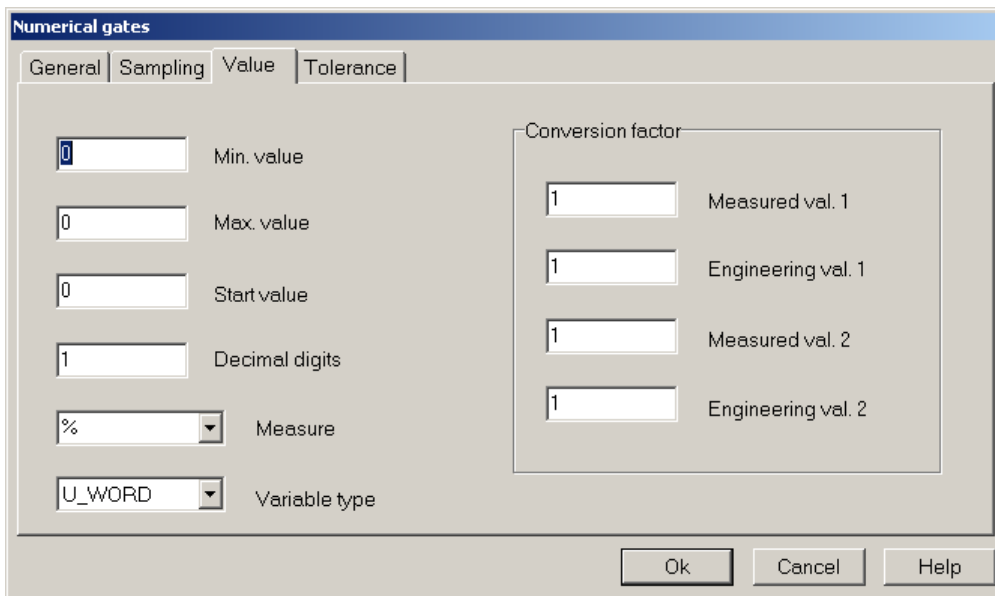
Choose...

Ok Cancel Help

*OUT numeric variable configuration – General folder*



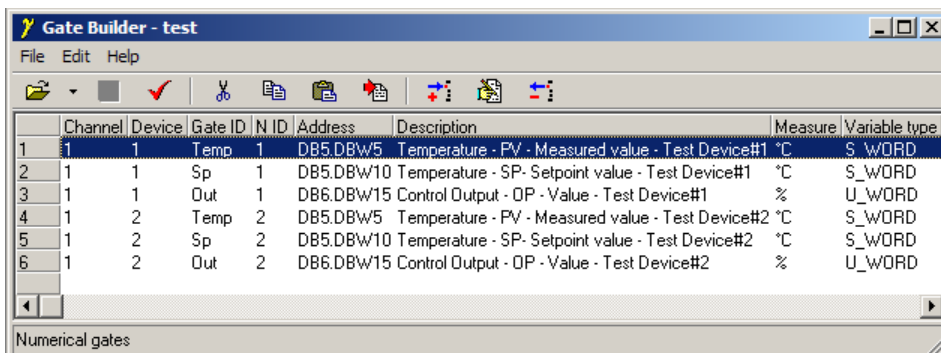
*OUT numeric variable configuration – Sampling folder*



*OUT numeric variable configuration – Value folder*

### End result

After you have defined all numeric variables, you should see the **Gate Builder** main page similar to the one shown below.



*Numeric variable database*

## 5.2 Digital variables configuration

Repeat numeric gates configuration for both devices(PLC Siemens S7300/400 DB area), having care to change device number (Device) and N ID.

Digital gates dialog box, General tab. Fields include: Alarm (dropdown), Gate ID (Alarm), Record on DB (checked), Writing enabled (unchecked), N ID (1), Description (Internal Alarm Status - Test Device #1), and Access groups (empty). Buttons: Ok, Cancel, Help.

*ALARM digital variable configuration – General folder*

Digital gates dialog box, Sampling tab. Fields include: Channel (1), Read block (empty), Device (1), Write block (empty), Address (DB8.DBB12.1), Sample (Always), and Sample freq. [Sec.] (1). Buttons: Ok, Cancel, Help.

*ALARM digital variable configuration – Sampling folder*

### Final result

After you have defined all numeric variables, you should see the **Gate Builder** main page similar to the one shown below.

Channel	Device	Gate ID	N ID	Address	Description
1	1	Alarm	1	DB8.DBB12.1	Internal Alarm Status - Test Device #1
2	1	Alarm	2	DB8.DBB12.1	Internal Alarm Status - Test Device #2

Digital gates

*Digital variables database*

### 5.3 Alarms gates configuration

So we have created numeric and digital gates database; now we will create as example an event/alarm gate for each device.

These gates are not read from devices but are software generated and their status will be displayed in runtime as "event and alarm status" and "event and alarm history".

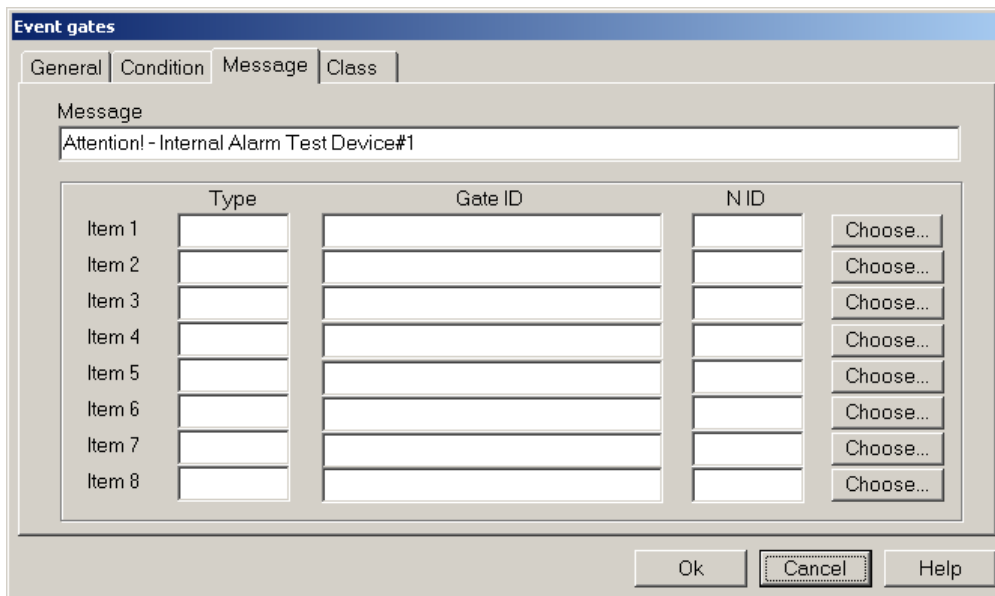
Let's create alarm gates with the following conditions.

Name	Condition	Filter time	Message	Registration
Internal_Alarm,1	Alarm,1 = 1	10 s	Attention! Internal Alarm Test Device#1	yes
Internal_Alarm,2	Alarm,2 = 1	10 s	Attention! Internal Alarm Test Device#2	yes

#### Configuration of alarm gate Internal\_Alarm

*Internal\_Alarm ALARM gate configuration – General folder*

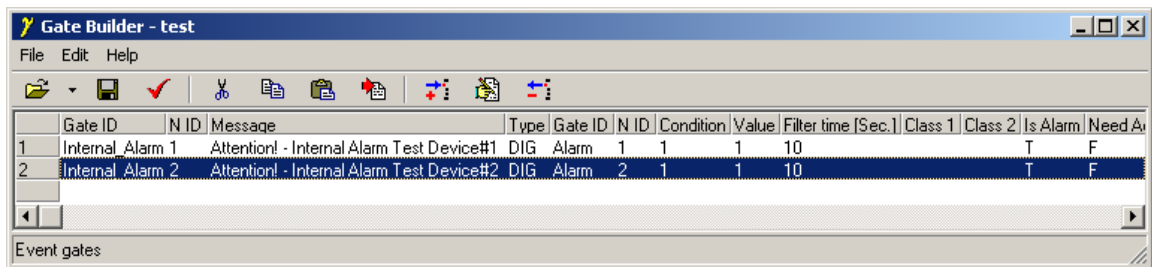
*Internal\_Alarm ALARM gate configuration – Condition folder*



*Internal\_Alarm ALARM gate configuration –Message folder*

### Final result

After you have defined all numeric variables, you should see the **Gate Builder** main page similar to the one shown below.



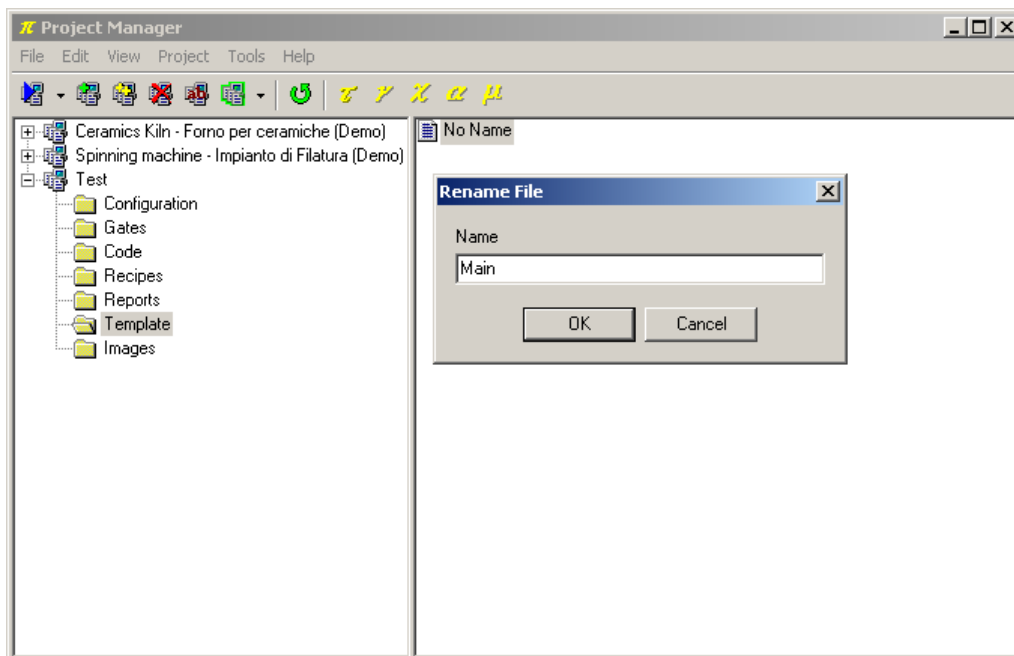
*Alarm gates database*

## 6. Creating a template

Now supervision network has been set; we have defined the logical channel and its link to PC COM port and we have connected it to Siemens MPI protocol; we have linked to this channel two devices (Test Device#1 e Test Device#2); for both we have declared sampling variables and alarm/event internal variables.

Now it is the moment to build a template for the application.

Select **Template** folder and create a new template, selecting the item **New>File** from **Edit** menu. Rename the just created template using the name **Main**, do this selecting it and then using **Rename** item from **Edit** menu.




*Template creating*

Double-clicking on created template, **Template Builder** start in order to build the graphic page.

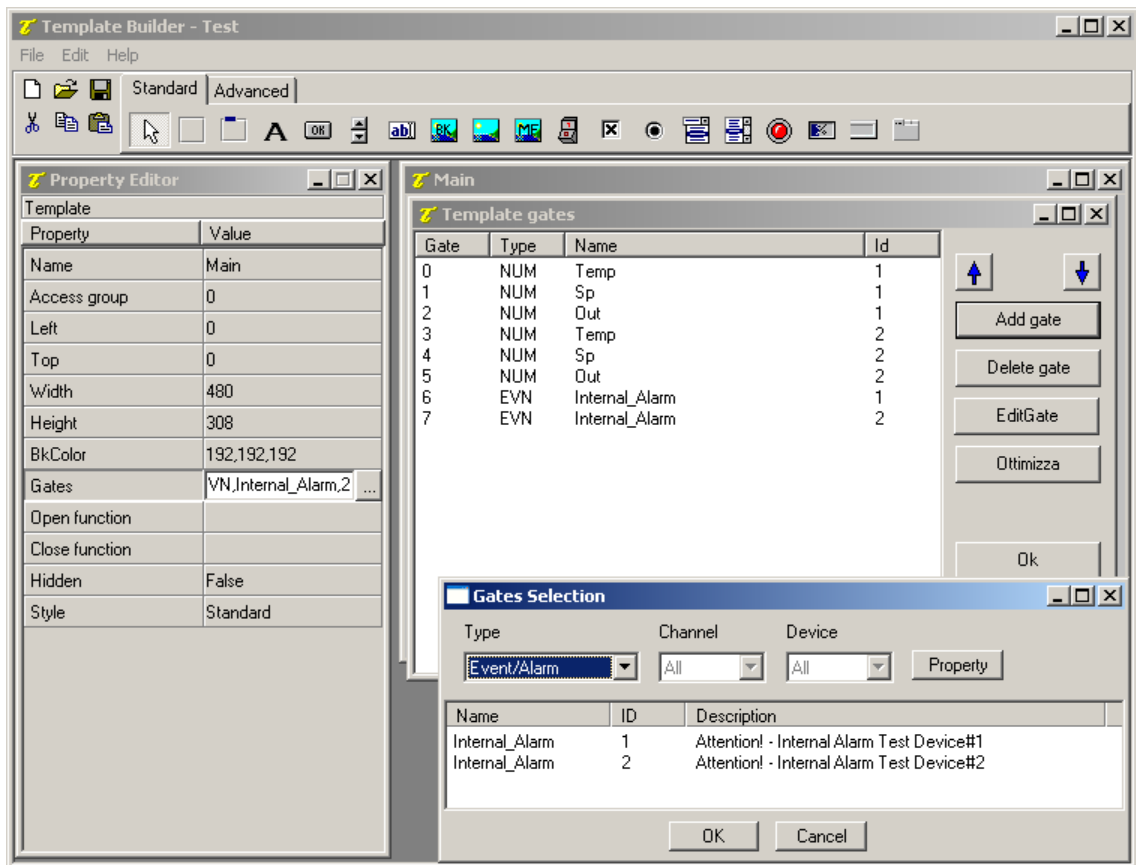
### 6.1 Declaring template variables

First it is necessary to declare which variables we will use in the template; in this example we will use all of them.

Click on button  alongside of the **Gates** item in the *Property Editor* (*Property Editor* is the window on the left side of the screen that allows to modify template elements properties).

A new windows will appear; press **Add gate** button, select the first numeric gate and press **Ok**. Repeat this operation for each numeric, digital and alarm gate that belongs to the application.







Template variables declaration


## 6.2 Inserting a Label object

Firstly build a *Frame* that will contain all the elements that will be inserted later.

To do this, select *Frame* object among the ones on the upper bar ( , it is the first on the left) and click on the template, a void rectangle will be displayed.

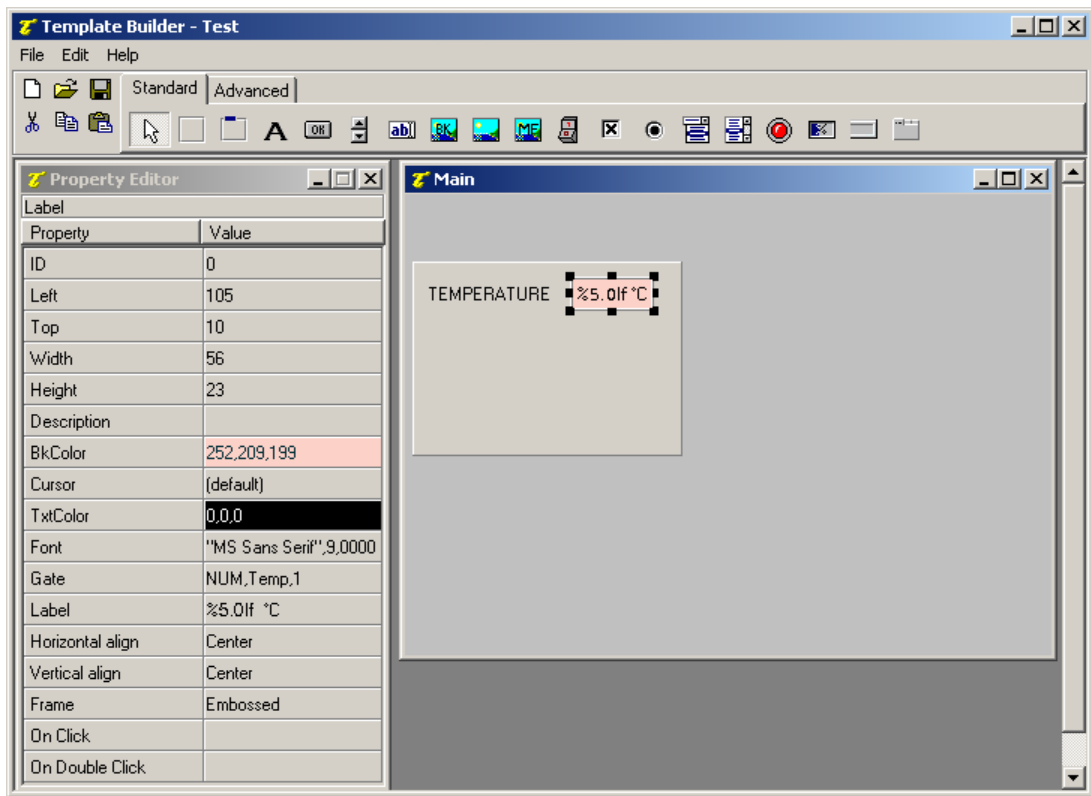
The next step is to insert into the created frame a static label that is a static text; select *Label* object among the ones on the upper bar ( , then click into the frame. To modify the text displayed into the object, use Property Editor, click alongside of the property `Label` and digit `TEMPERATURE`.

Alongside of just inserted label, position another one to visualise temperature read form the device.

To link the Label to the numeric variable `TEMP`, click on the button  alongside of the item Gate in Property Editor and select `NUM, Temp, 1` among the available gates.

Modify in addition the property `Label` inserting `%5.01f °C`.

Every described object can be formatted and placed as you like using Property Editor.




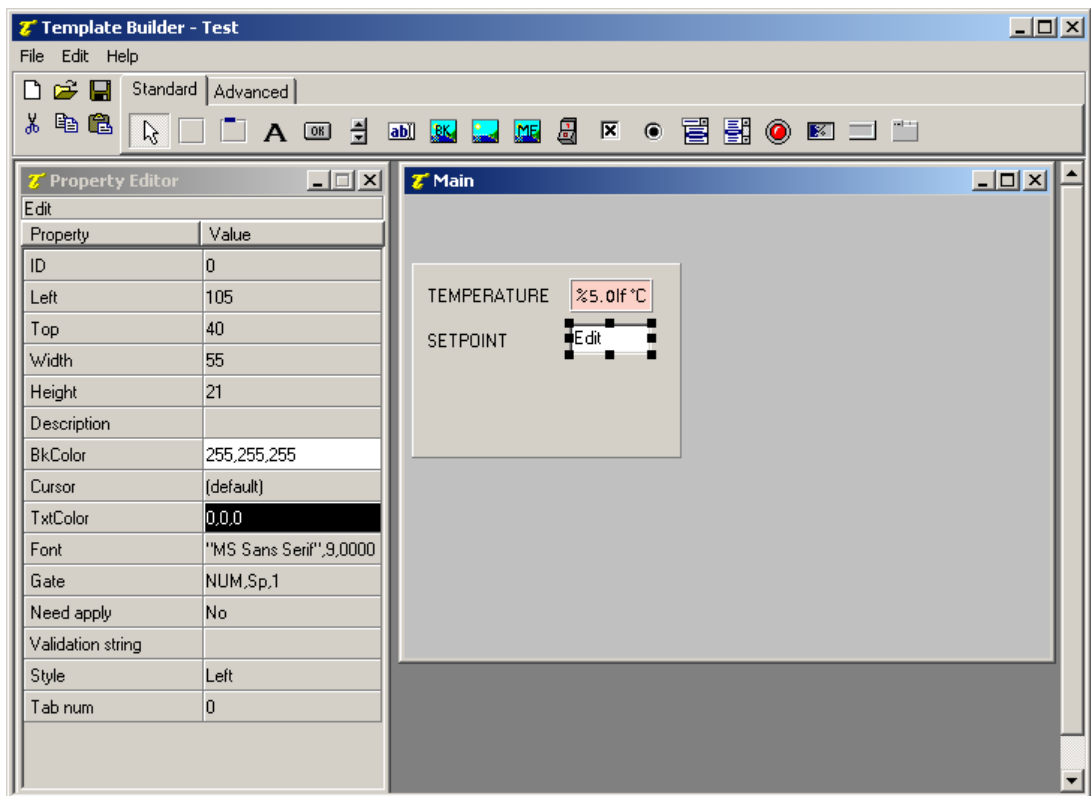
Label object inserting

### 6.3 Inserting an *Edit* object

Insert another Label, positioning it below *TEMPERATURE* and modify the text in *SETPOINT*

A control will be inserted that will allow to modify the value of the *SP* gate and to send it to the device.


Select *Edit* (  ) object from tool bar; and, as done before, link it to *NUM, Sp, 1* gate using the Property Editor.



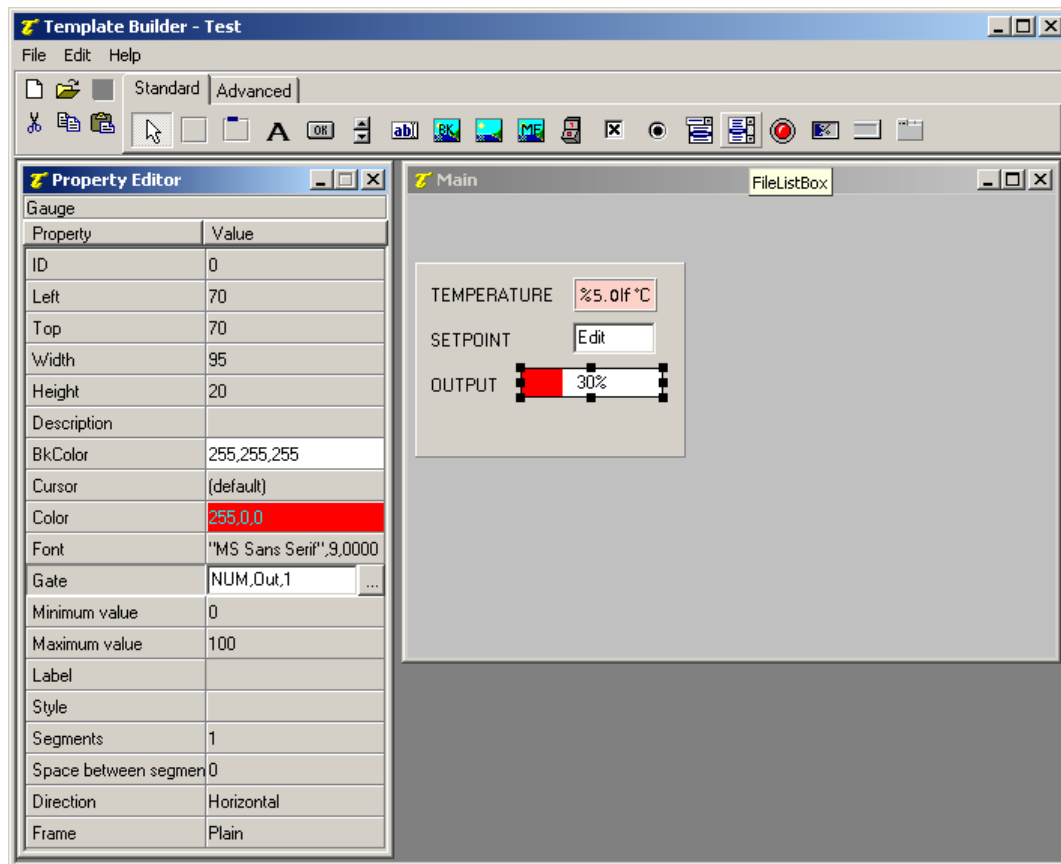
Edit object inserting

## 6.4 Inserting a Gauge object

Insert another Label, positioning it below SETPOINT and modify the text in OUTPUT.

Insert now a *Gauge* object (  ) alongside of the previous Label; link it to NUM, Out, 1 gate using the Property Editor.


In this way the value of the device output power will be displayed in bar format.

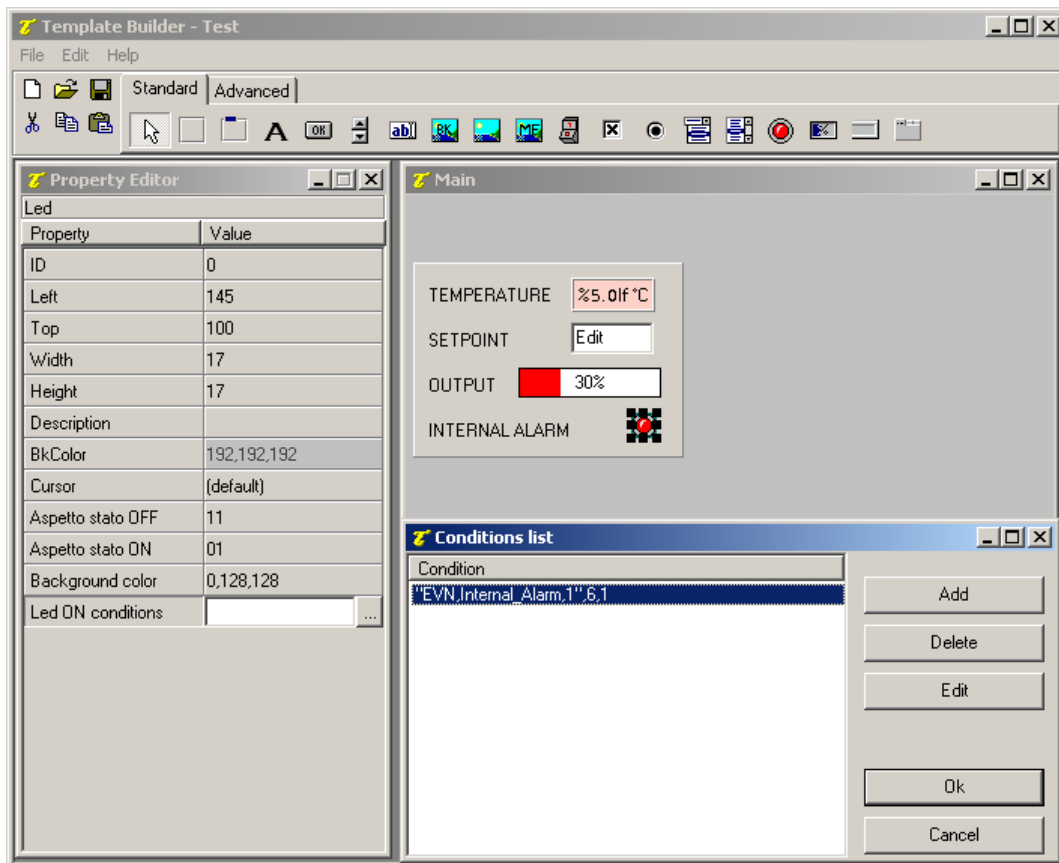


Gauge object inserting

## 6.5 Inserting a Led object

Insert another Label, positioning it below OUTPUT and modify the text in INTERNAL ALARM.

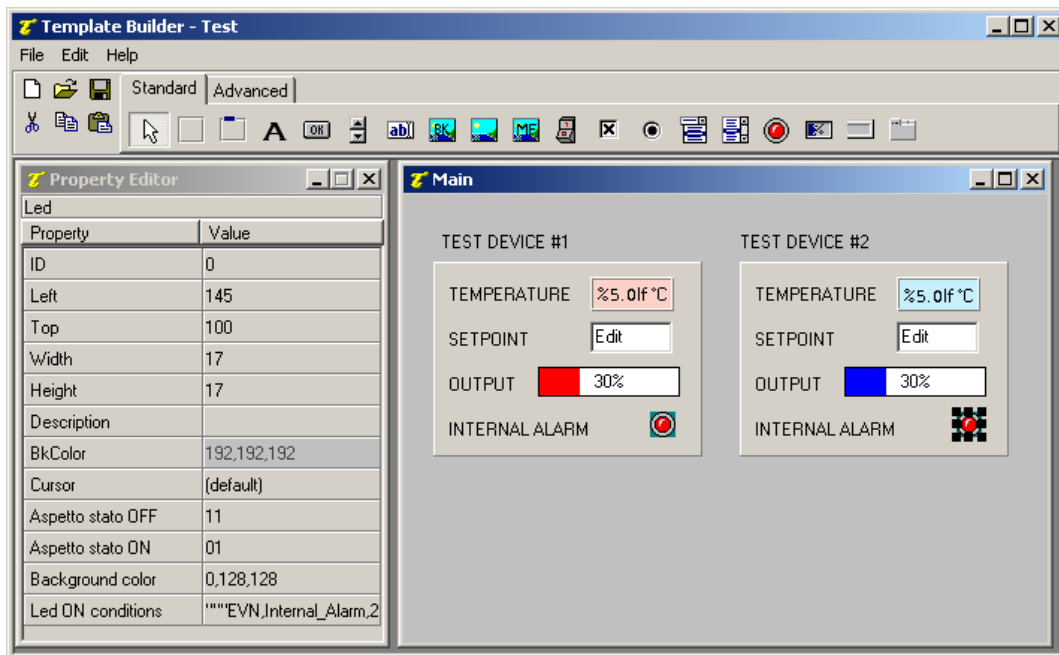
Insert now a *Led* (  ). alongside of the previous Label. To "give animation" to the object it is necessary to specify which is the condition that make it change colour; modify Led ON conditions property linking led activation condition to *Internal\_Alarm,1* (*Internal\_Alarm,1 == true*) alarm activation. A red led will be shown in presence of the alarm, otherwise led will be green.




Led object inserting

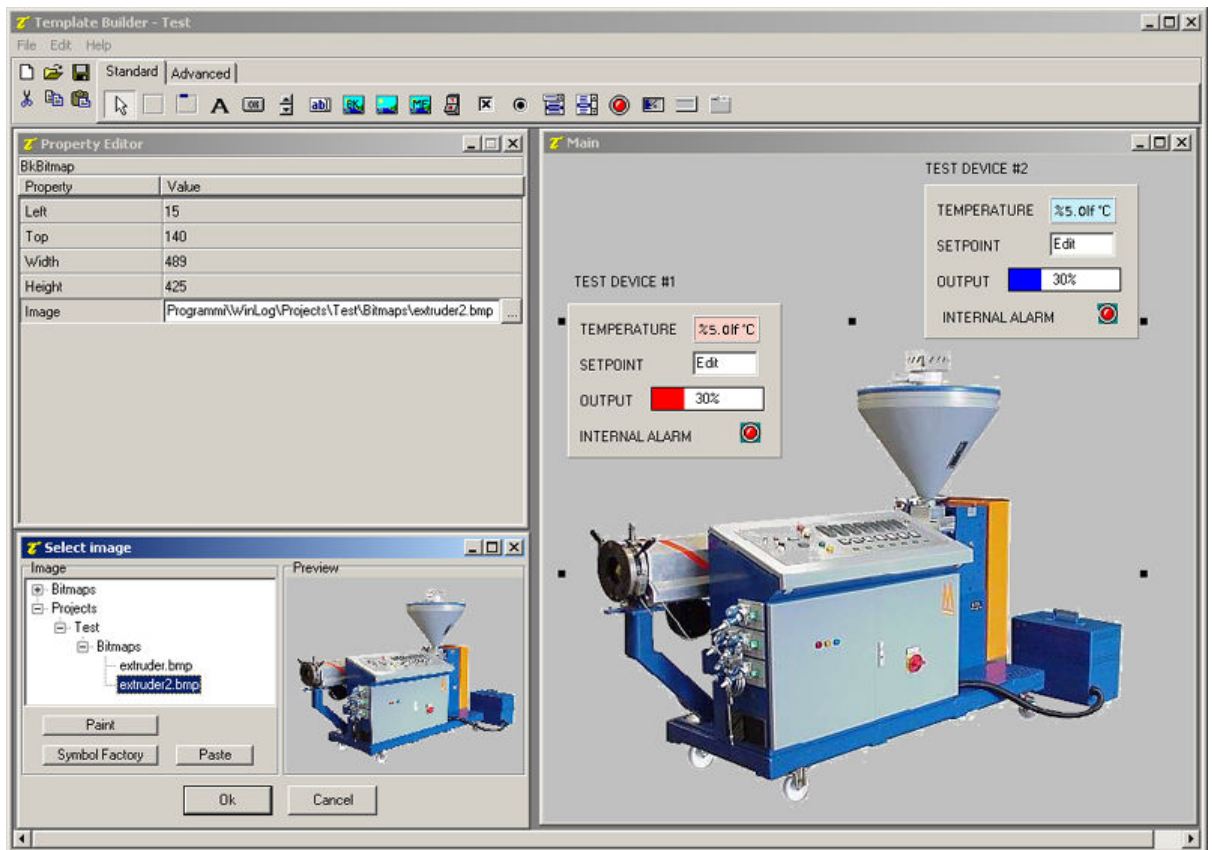
## 6.6 Completing template

All variables read from device 1 are now displayed; to display also device 2 variables it is enough to select the Frame we have created, copy and paste it in the template. Be careful not to paste it in the source frame; to avoid this mistake click in a free object area of the template before pasting it. Now we have only to modify variables links in Label, Edit, Gauge and Led objects to obtain a supervision interface for the *Test Device #2*.



Two devices supervision template

To complete the template, insert now a *BkBitmap* object (background bitmap, ) previously created using any graphic design software (for example *Paint*) and saved in project *Bitmaps* folder.

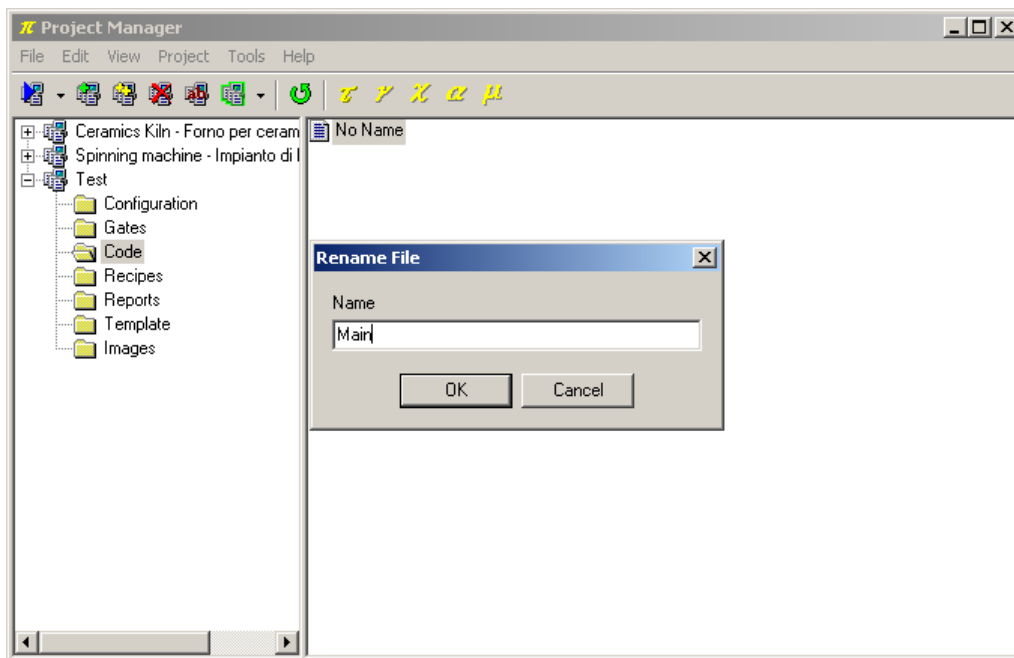


*Background bitmap inserting*

## 7. Winlog Pro code example

Now create the code function that allows showing the template at runtime startup.

In Code folder create a file and rename it Main; opening it, **Code Builder** starts.



Creazione di un file di codice

**Code Builder** is the **Winlog Pro** programming environment; we will use it only to define a function that will open the main template at the application startup.


Copy and paste the following code:

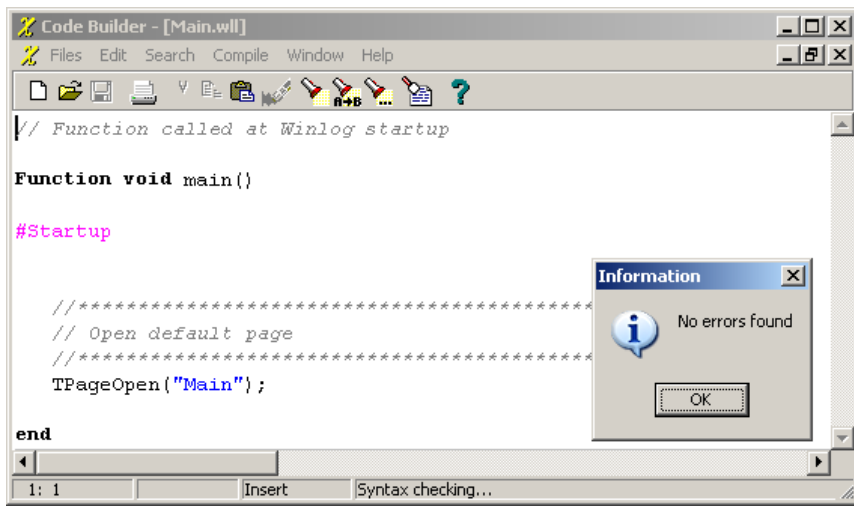
```
// Function called at Winlog startup

Function void Main()
#Startup

    //*****
    // Open default page
    //*****
    TPageOpen ("Main");

end
```

To check syntax of the code use function `Check syntax` ()



*Code syntax checking*

## 8. Project execution

Our example is complete.

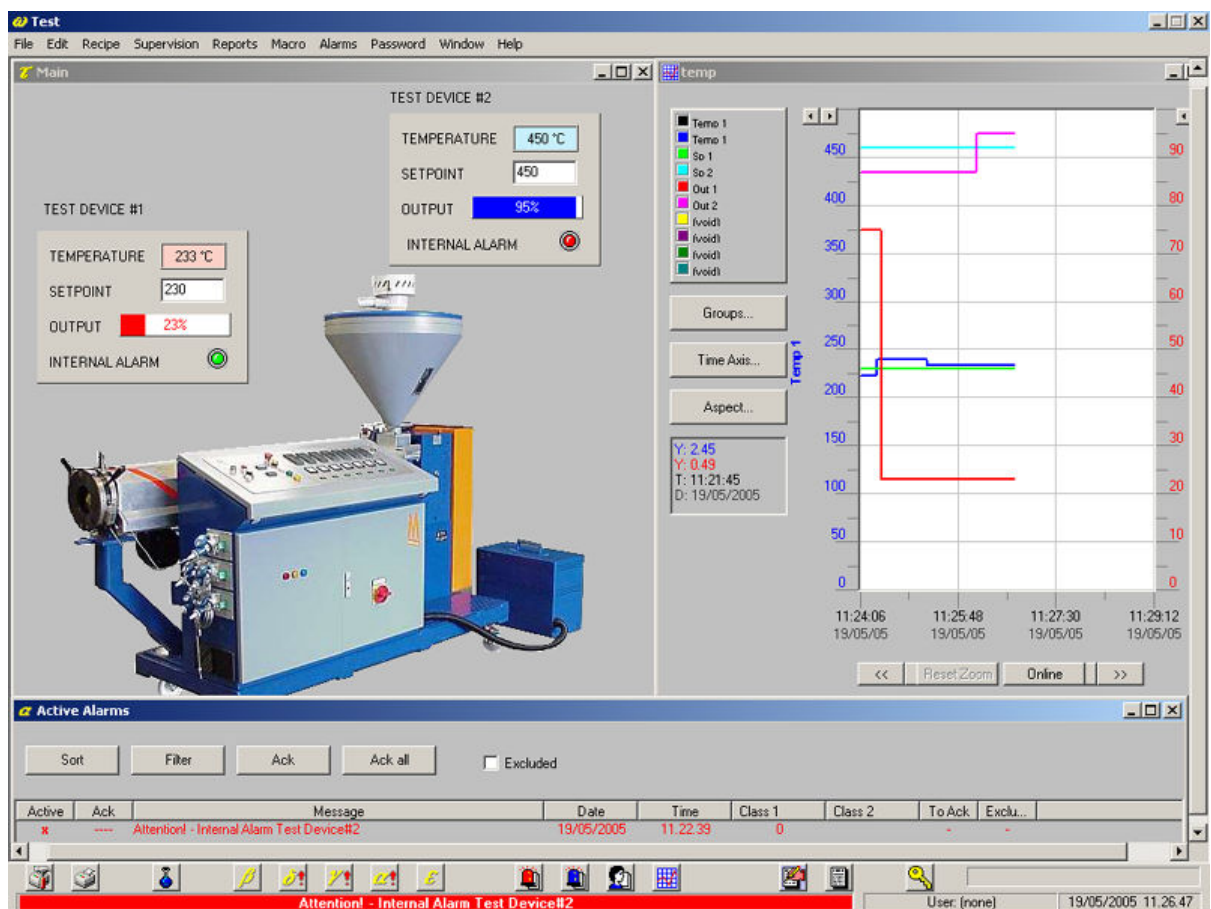
Wire devices to the serial port; to run the project, in **Project Manager** select **Execute...** from **Project** menu.

Now we are entering in the "run-time" phase that is application execution mode. **Winlog Pro** samples variables from devices and processes results in graphical representations (trends and template) and in tabular representations (reports and historical data).

At project startup, main template will appear automatically.

From **Supervision** menu you can display graphical trends; select menu item **Charts...** and define the group of variables that you want to display as graphical trends.

Again in **Supervision** menu you can display both the online status (**Status>Alarms...**) and the story (**Historical>Alarms...**) of all alarms that have been created with **Gate Builder**.



Project execution