Software Description

Advant Controller 31
Intelligent Decentralized
Automation System

Special Functions 90 Series

ABB
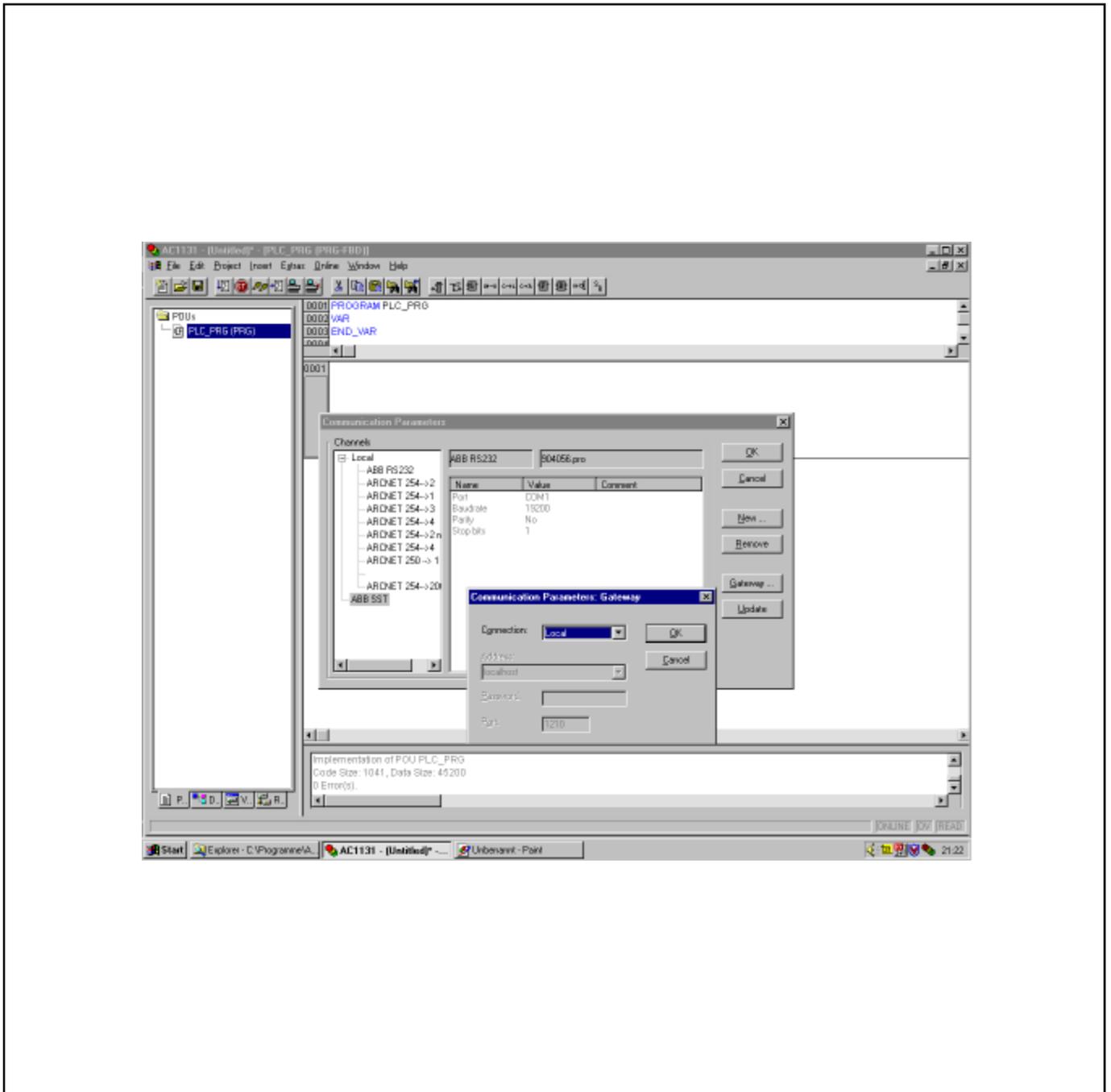
Contents

1 Creating a project for the controller 07 KT 97

1.1 Example project for the controller 07 KT 97

In the following procedure we want to create a small example project for the controller 07 KT 97, download it and finally test it.

Task: Input E62_00 shall be routed to output A62_00 of the controller 07 KT 97.

Perform the following steps:

1. Connect the controller 07 KT 97 to 24 V DC (refer to volume 2 – Hardware 07 KT 97).

2. Connect COM1 of the controller 07 KT 97 to the serial interface of your PC.

3. Start the 907 AC 1131 software. The project you edited last is opened. If necessary, close this project by selecting "File/Close".

4. Select "File/New".
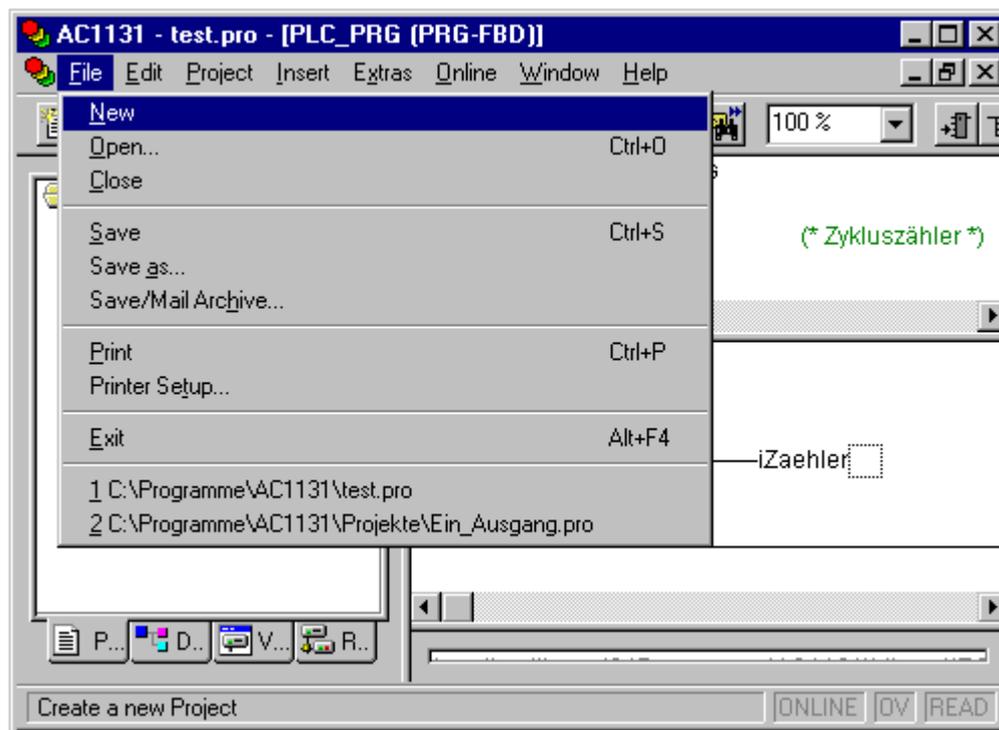
5. A window appears where you can select the PLC. Using the mouse, select the controller "07KT97" and click on "OK".
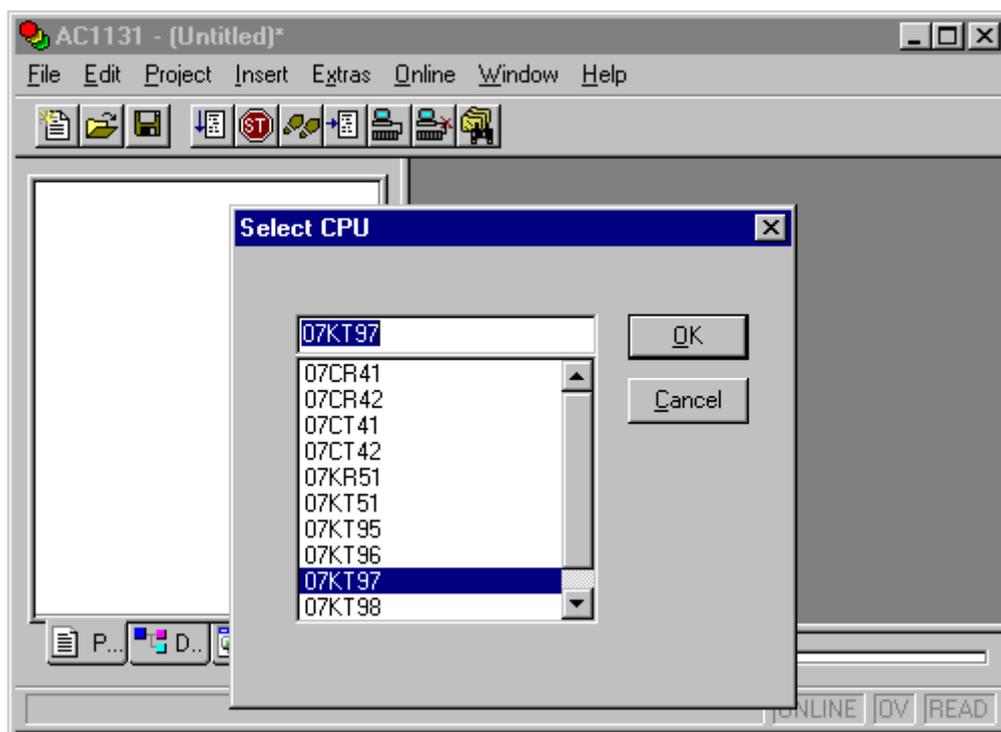
6. A window appears where you can select the POU name, type and language. For the name and type keep the settings PLC_PRG and PROGRAM. Select "FBD" for the language and confirm with "OK".
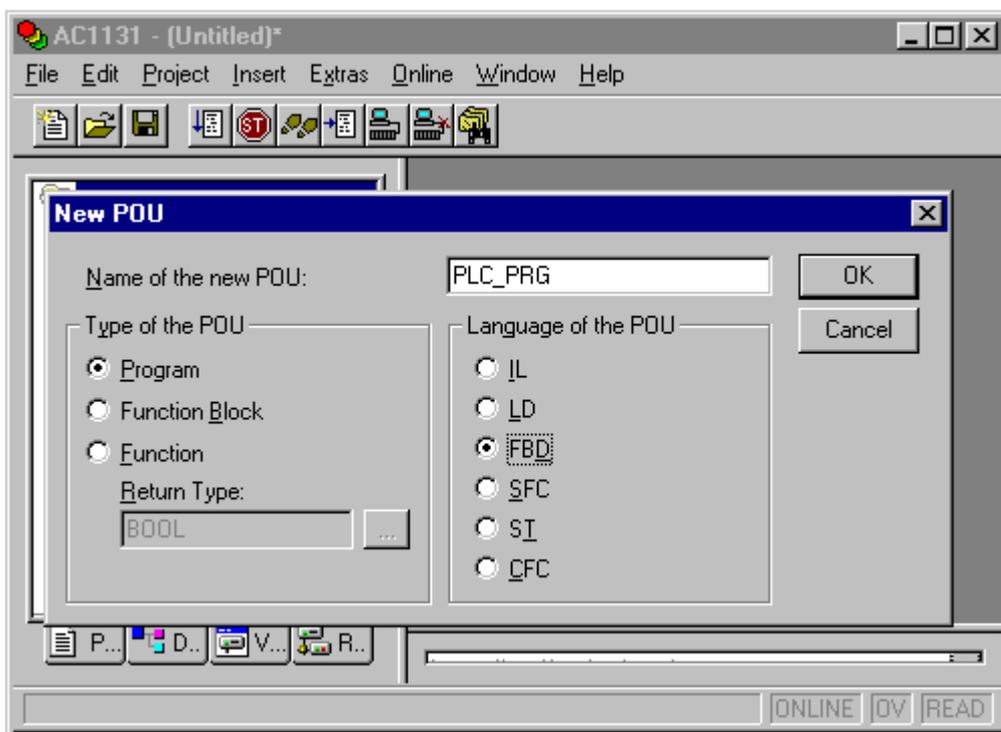
7. You have now opened a new project. With the insertion of the POU PLC_PRG the following global variables have been created in the "Global Variables" folder of the "Resources" page.
– Object "Systemkonstanten" containing the default values of the system constants (refer to volume 15, System technology, section "System constants").
– Object "Fehlermerker" containing all the system and error flags (refer to volume 15 - System technology, section "System and diagnosis flags").
– Object "Auslastung" containing the system flags for the indication of the controller capacity utilization (refer to volume 15 - System technology, section "System and diagnosis flags").
– The empty objects "Global_Variables" and "Variable_Configuration" to enter further global variables.

Switch to the "Resources" page by double clicking on the "Resources" tab.
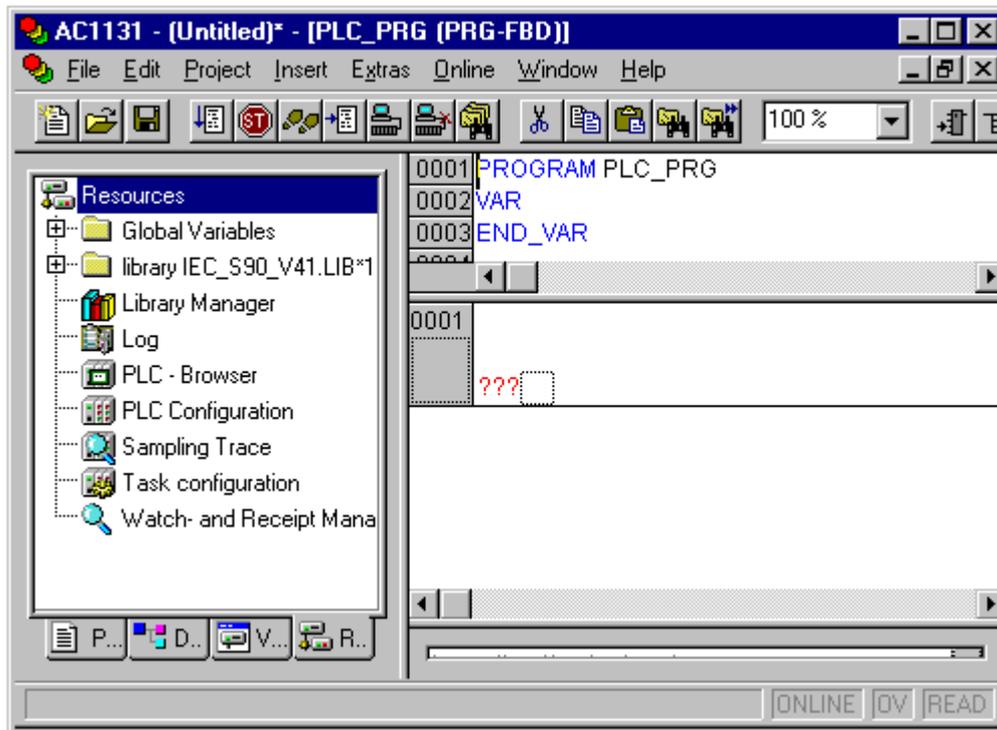
In this page, click on "Global Variables" or open the respective folder. You can see the respective variables if you open the individual objects with a double click.
(In the example, the object "Systemkonstanten" is opened).

```
AC1131 - (Untitled)* - [Systemkonstanten]

File  Edit  Project  Insert  Extras  Online  Window  Help

Resources                          0001 VAR_GLOBAL
  Global Variables                 0002 (* Standard - Systemkonstanten *)
      Auslastung                   0003 (* -1 Master / -2 Standalone / 0..61 Slave *)
      Fehlermerker                 0004     KW00_00 AT %MW3000.0 : INT := -2;
      Global_Variables             0005 (* Pufferung Binärmerker %MX000.00-%MX51
      Systemkonstanten             0006     KW00_01 AT %MW3000.1 : INT := 0 ;
      Variable_Configurat          0007 (* Pufferung Wortmerker %MW1000.00-%MW
  library IEC_S90_V41.LI           0008     KW00_02 AT %MW3000.2 : INT := 0 ;
  Library Manager                  0009 (* Pufferung Doppelwortmerker %MW2000.00
  Log                              0010     KW00_03 AT %MW3000.3 : INT := 0 ;
  PLC - Browser                    0011 (* Pufferung Schrittkettenmerker %MW5000-%
  PLC Configuration                0012     KW00_04 AT %MW3000.4 : INT := 0 ;
                                   0013     KW00_05 AT %MW3000.5 : INT := 0 ;
  P...  D...  V...  R..            0014     KW00_06 AT %MW3000.6 : INT := 0 ;

                                                              ONLINE  OV  READ
```

In the "Resources" page, select the folder "Global Variables". Now select "Project/Import".
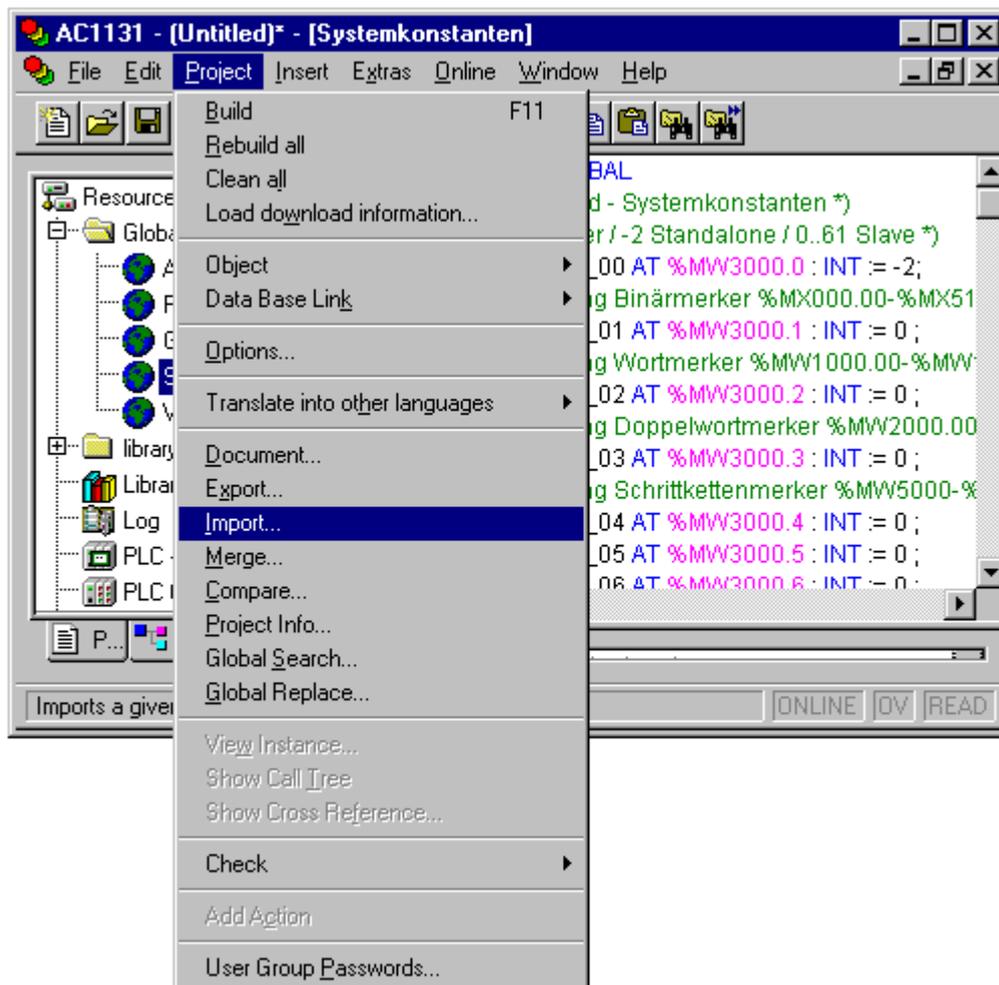
All *.exp files are displayed now (in the Library directory). Select "KT97_onb" and confirm it by clicking on "Open". Now the import file "KT97_onb.exp" is loaded. This file contains the declaration of all onboard in- and outputs of the 07 KT 97.
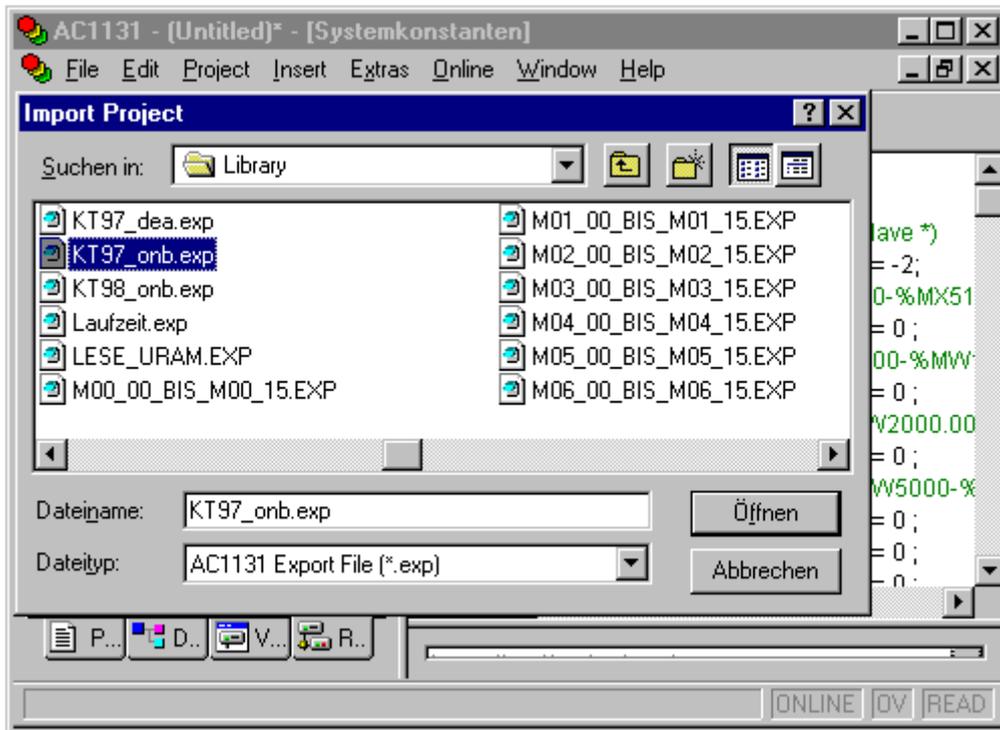
When you create a new project the IEC standard library IEC_S90_V41.LIB is also loaded into the library management. Select "Window/Library Manager".
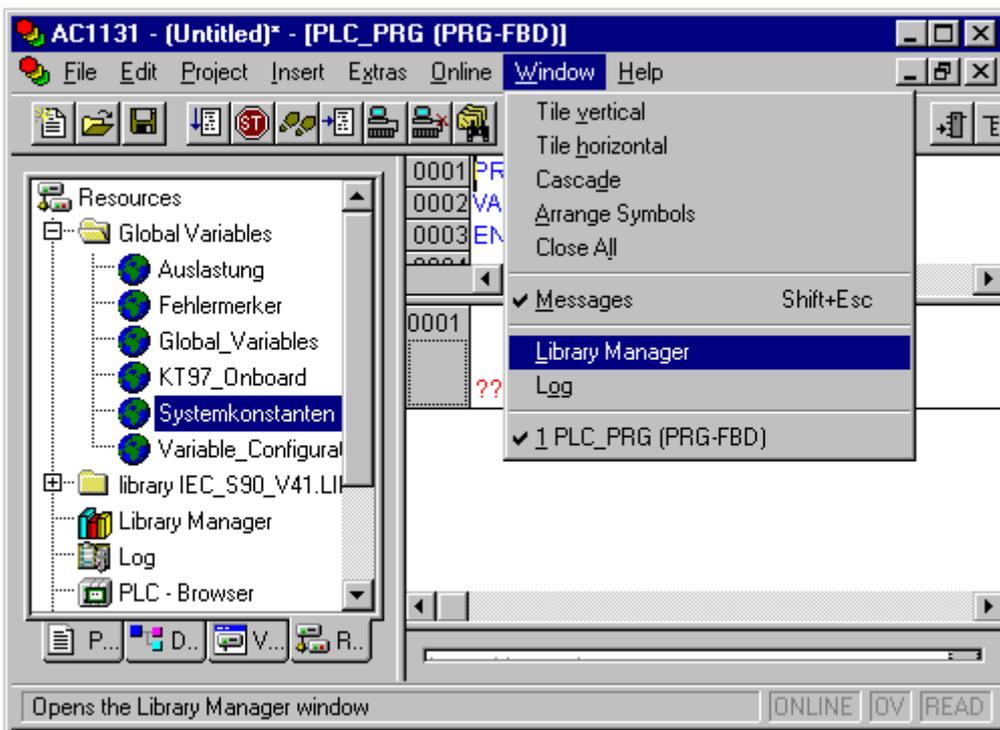
In the upper left window of the library management you can see "IEC_S90_V41...". (If you need further libraries for your project, you can open a menu in this window with a right mouse click and then select "Additional Library" from this menu. A window is opened where you can select the desired library).
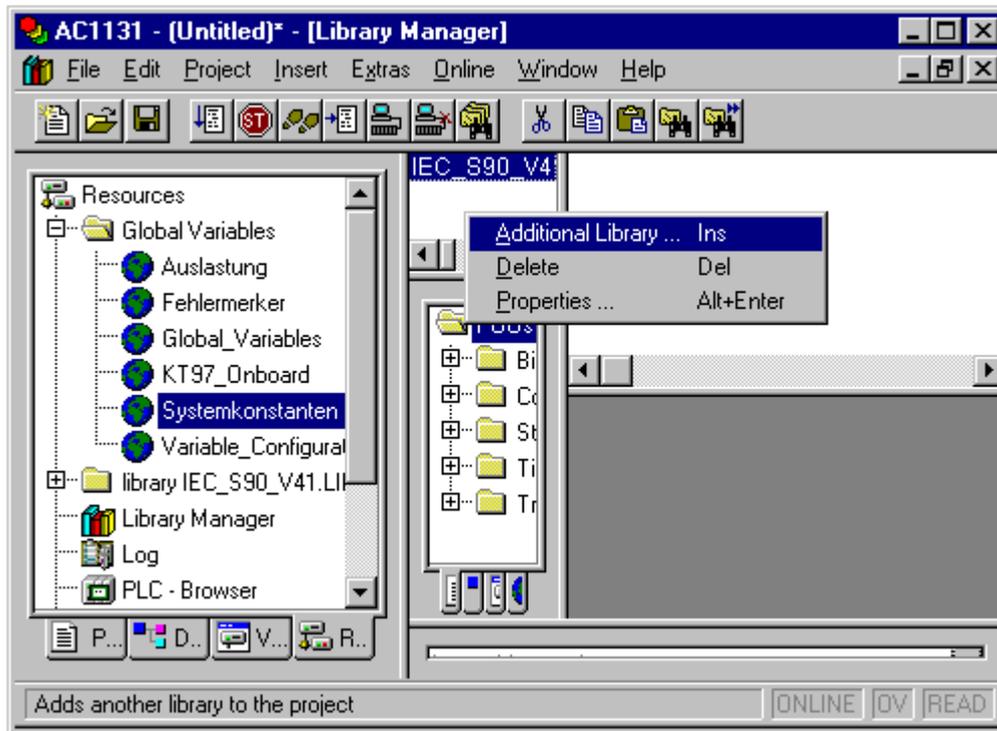
For our small example shown here we do not need a library.

8. Switch back to the POUs by clicking on the "POUs" tab and select the only existing entry "PLC_PRG".
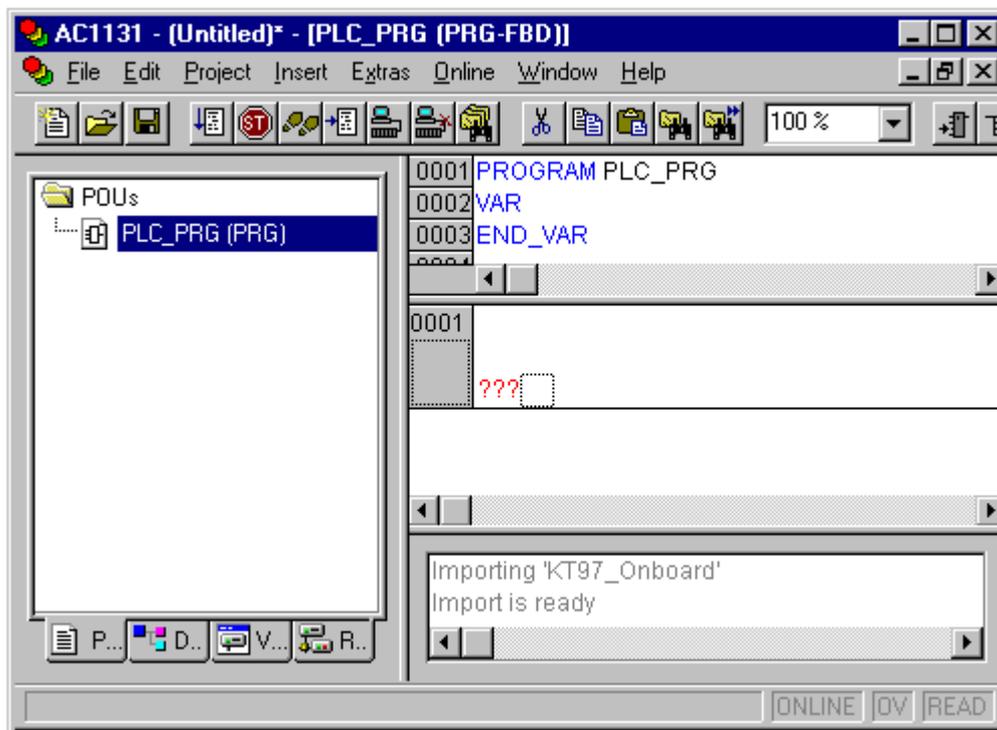
9. In the network "0001", position the mouse cursor on the project cursor (which is represented by a "quadrilateral") and press <CTRL><A> (press the keys <CTRL> or <STRG> and <A> simultaneously) or click with the right mouse button and then select "Assign" from the appearing context menu. The display is as following:
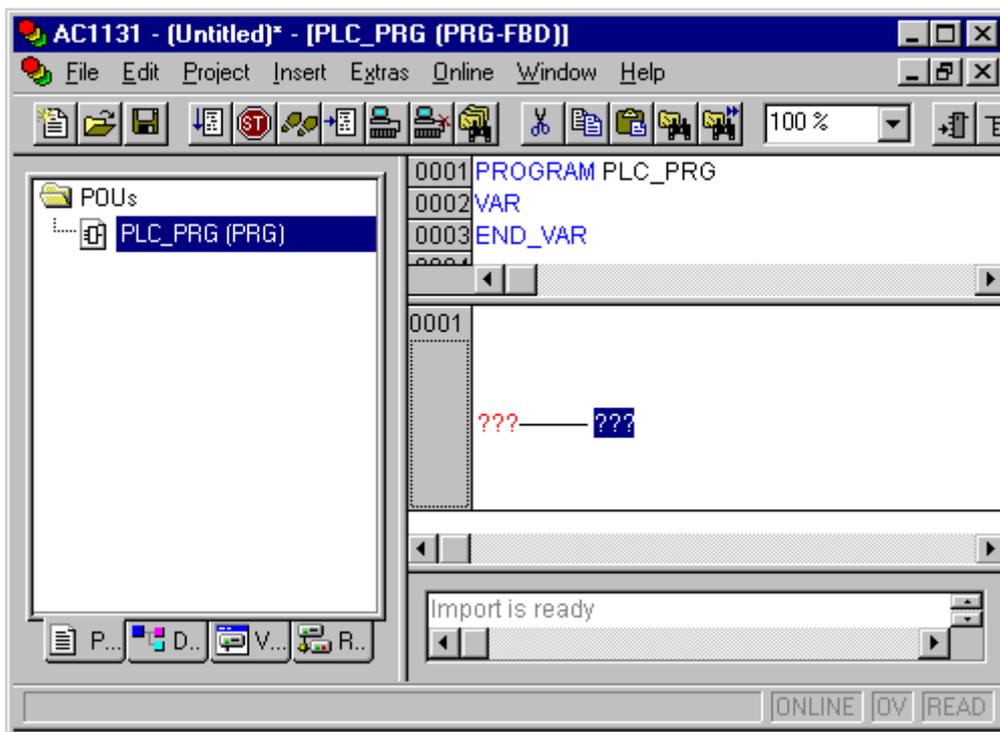
10. Instead of the question marks, enter the symbolic name for input E62_00 on the left side and on the right side the name for output A62_00. To do this, left-click on the question marks, type the name and confirm with <Enter>. These names are predefined in the import file KT97_onb.exp. If you want to use other symbolic names, you have to change the names in the object "KT97_Onboard" located in the "Global Variables" folder of the "Resources" tab. After changing the names you have to translate the project by pressing the <F11> key or by selecting "Project/Rebuild All" from the menu in order to announce the global variables to the POU(s).
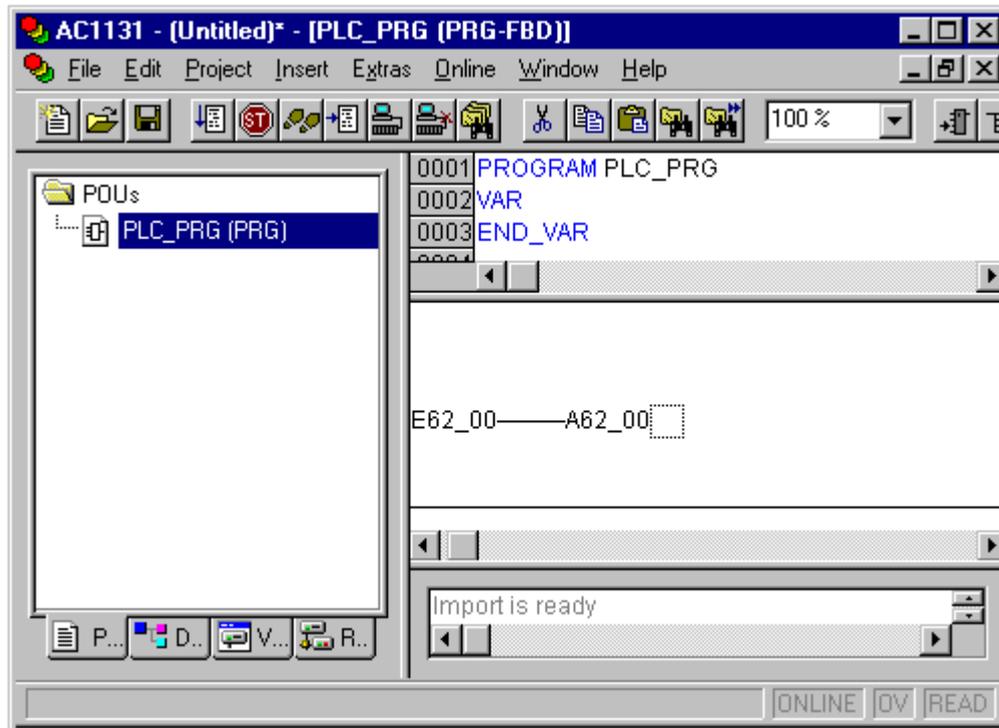
11. The definition of the project is now completed and the project has to be translated. To do this, press the <F11> key or select "Project/Rebuild All" from the menu.
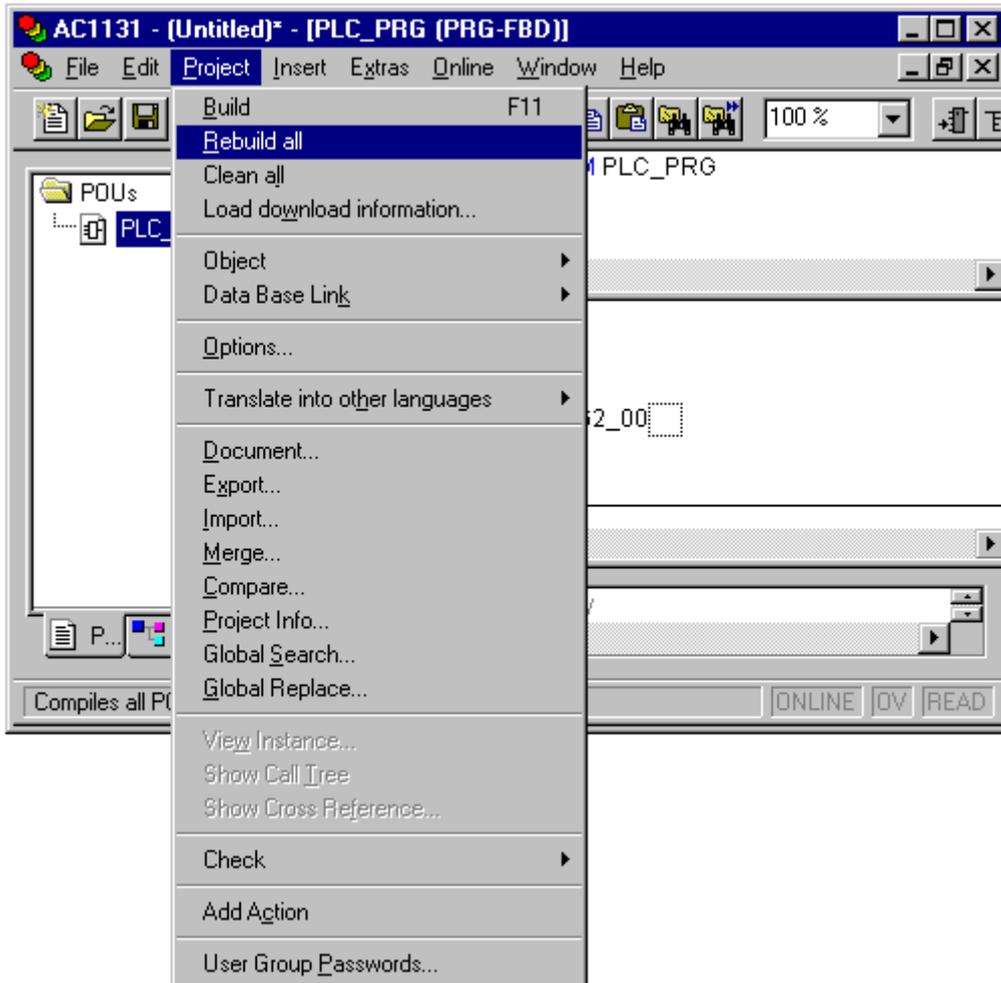
12. If no errors were detected, the message "0 errors" is displayed in the last line of the message box. In the next to last line the code and data size of the program is displayed in bytes. For the data size the size of the data segments is displayed. The default setting is always 2 data segments.
Now the communication parameters for the connection to the controller have to be set. To perform this, please select "Online/Communication Parameters".
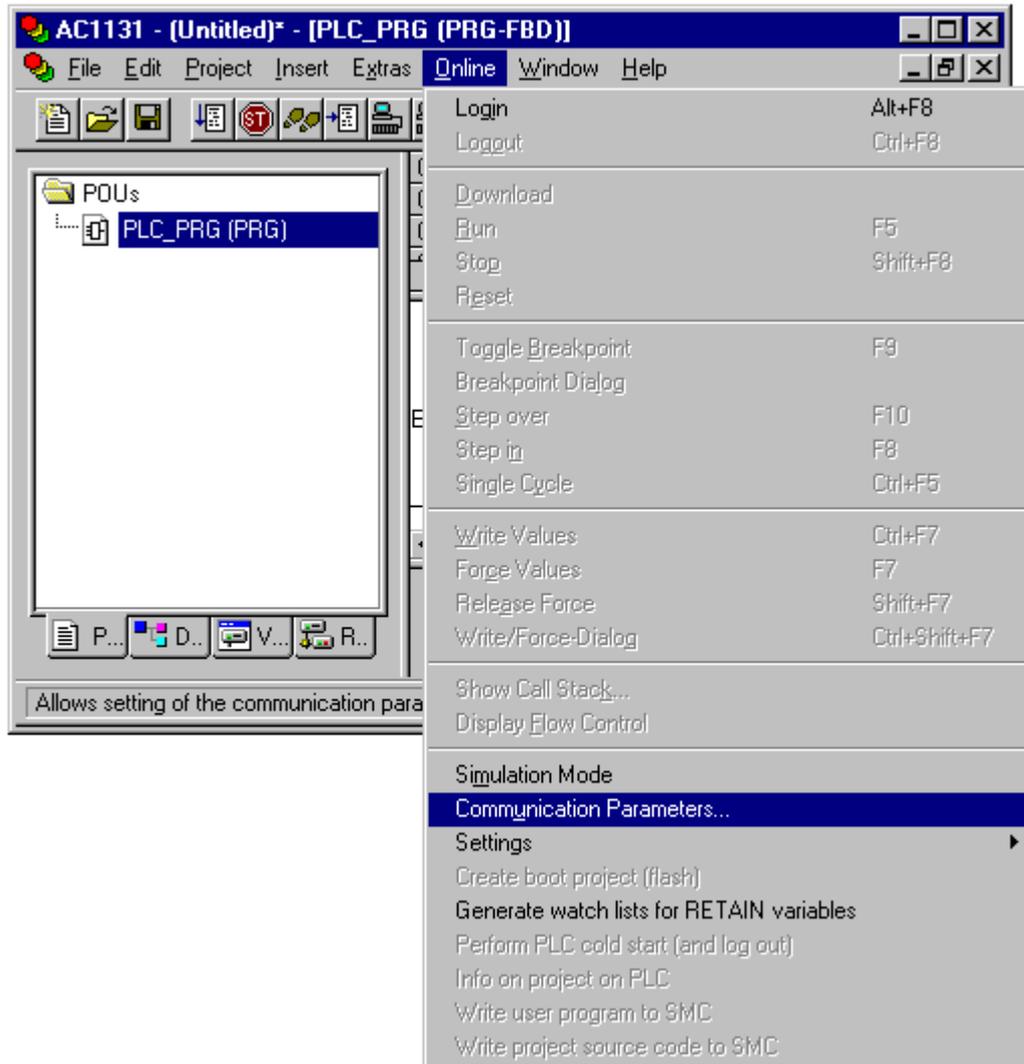
A window for setting the communication parameters appears. First click on "Gateway" and then select the "Connection" type "Local" in the subsequently appearing window. Confirm the setting with "OK".
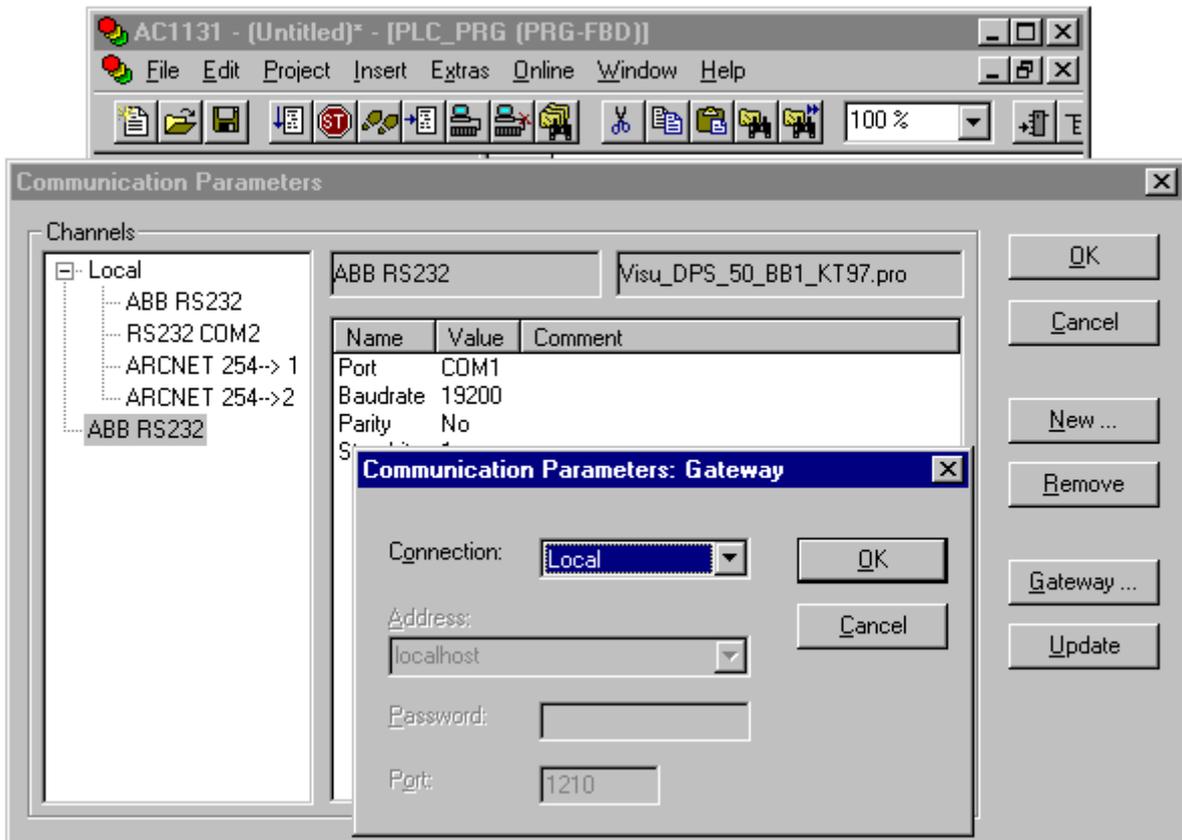
Select the driver "ABB SST" or "ABB RS232". This selects the standard serial driver for COM1 of the PC with the settings 19200 baud, no parity, 8 data bits and 1 stop bit.

If you use the serial interface COM2 of your PC, you have to add a new gateway setting. To perform this, please click on "New":
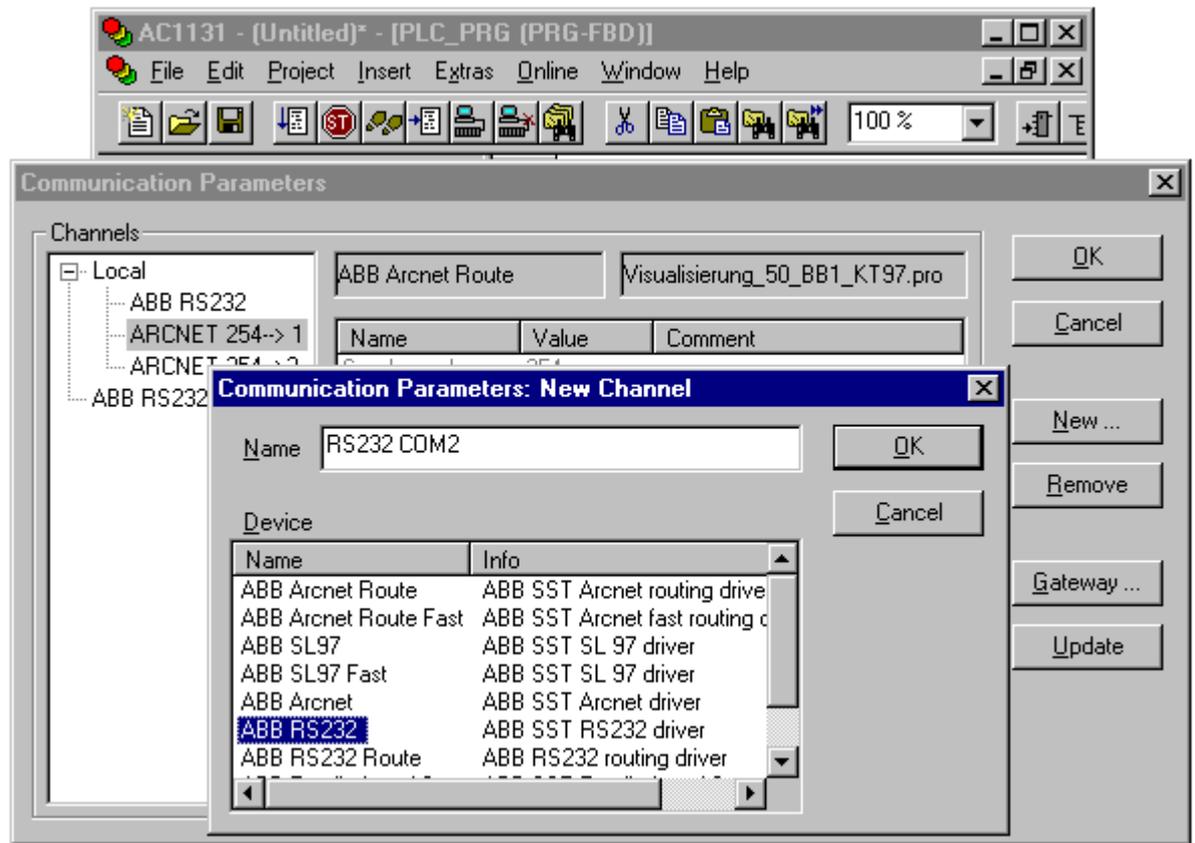
Enter any name of your choice for the gateway channel into the "Name" field (in our example "RS232 COM2"), select "ABB RS232" for the device and confirm with "OK".
You'll return to the "Communication Parameters" window.
In the window on the right, select the "Port" by double clicking on "COM1".
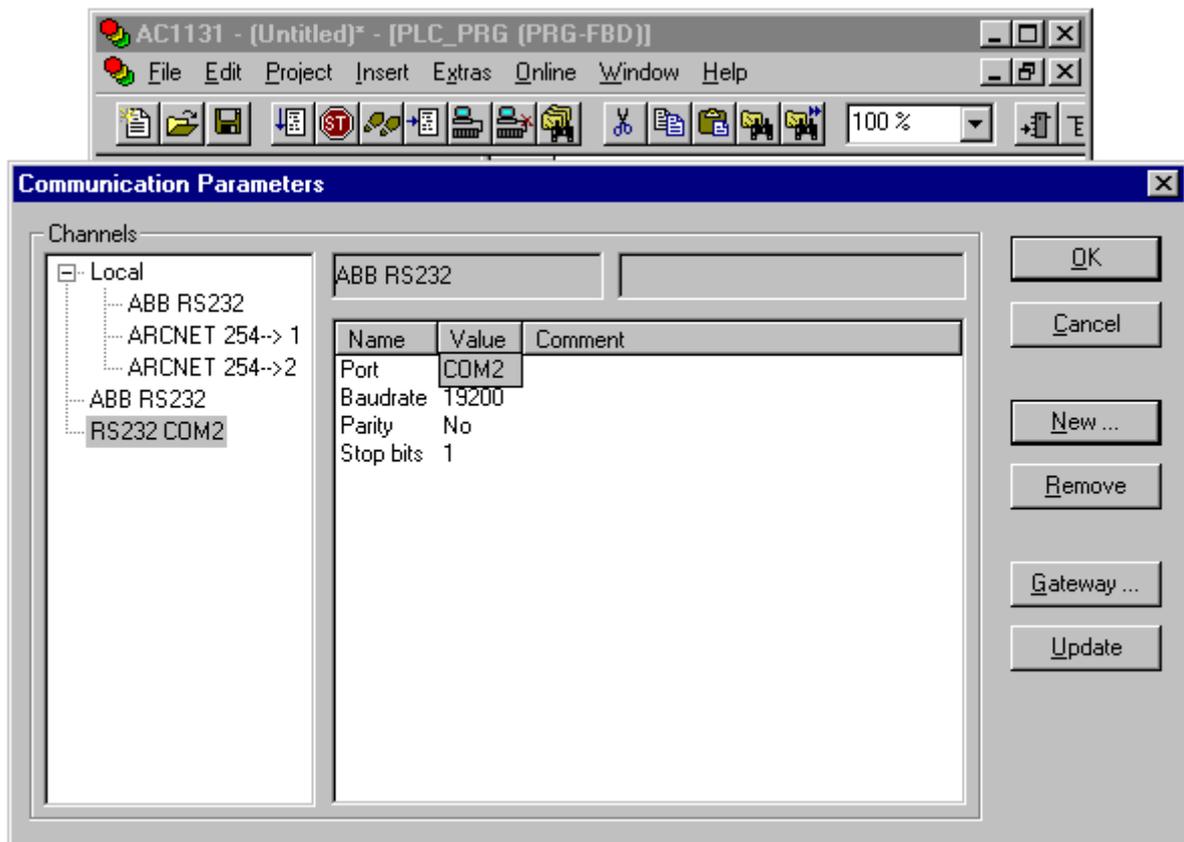
Click on "OK" to apply the communication parameters.

13. In the next step we will save the project. (You can also perform this much earlier, e.g. immediately after creation.) For this, select "File/Save as"
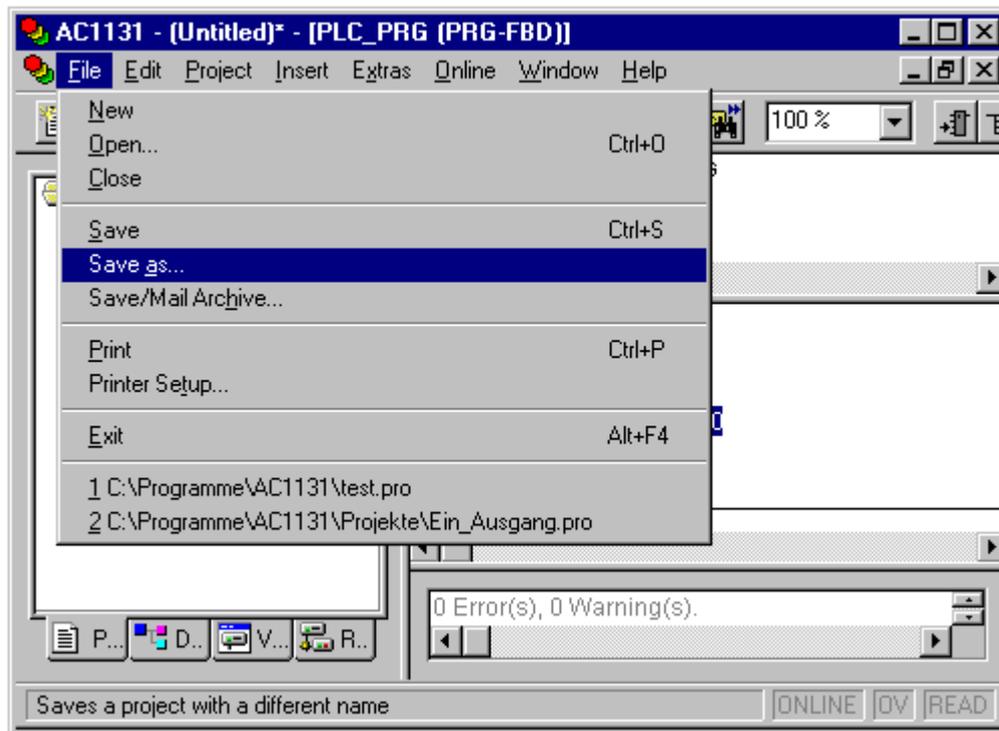
and then specify the desired path and project name:
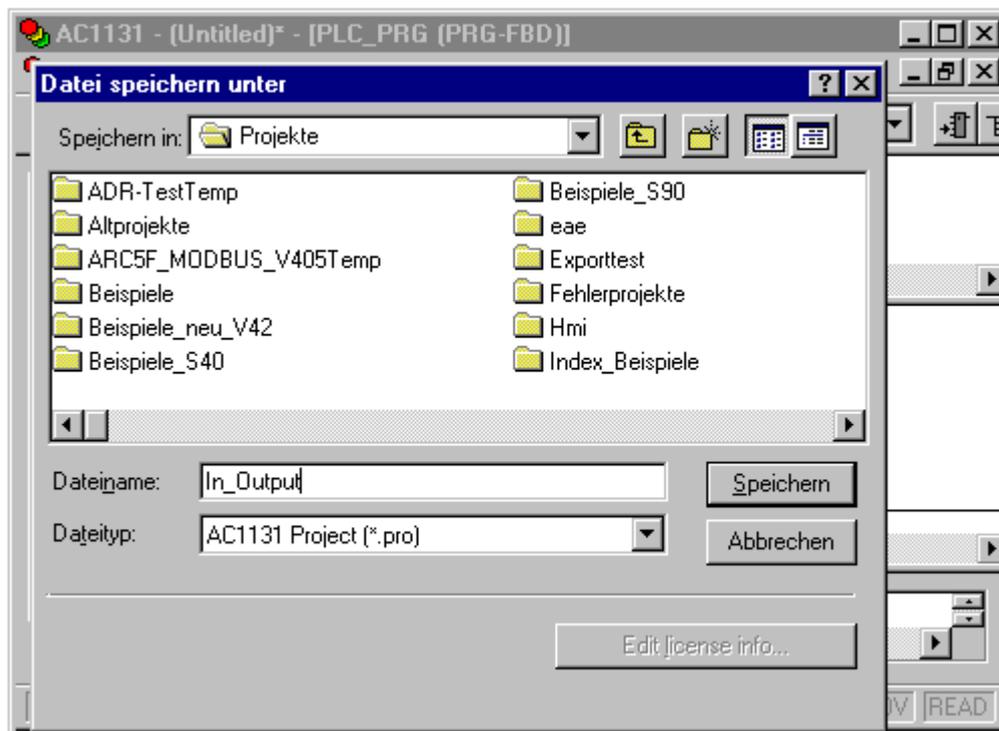
In our example the project is saved under the name "In_Output" to the folder "Projekte". The file extension "pro" is automatically created by the 907 AC 1131 software.

14. In the next step we will load the program into the controller. To perform this, please select "Online/Login":
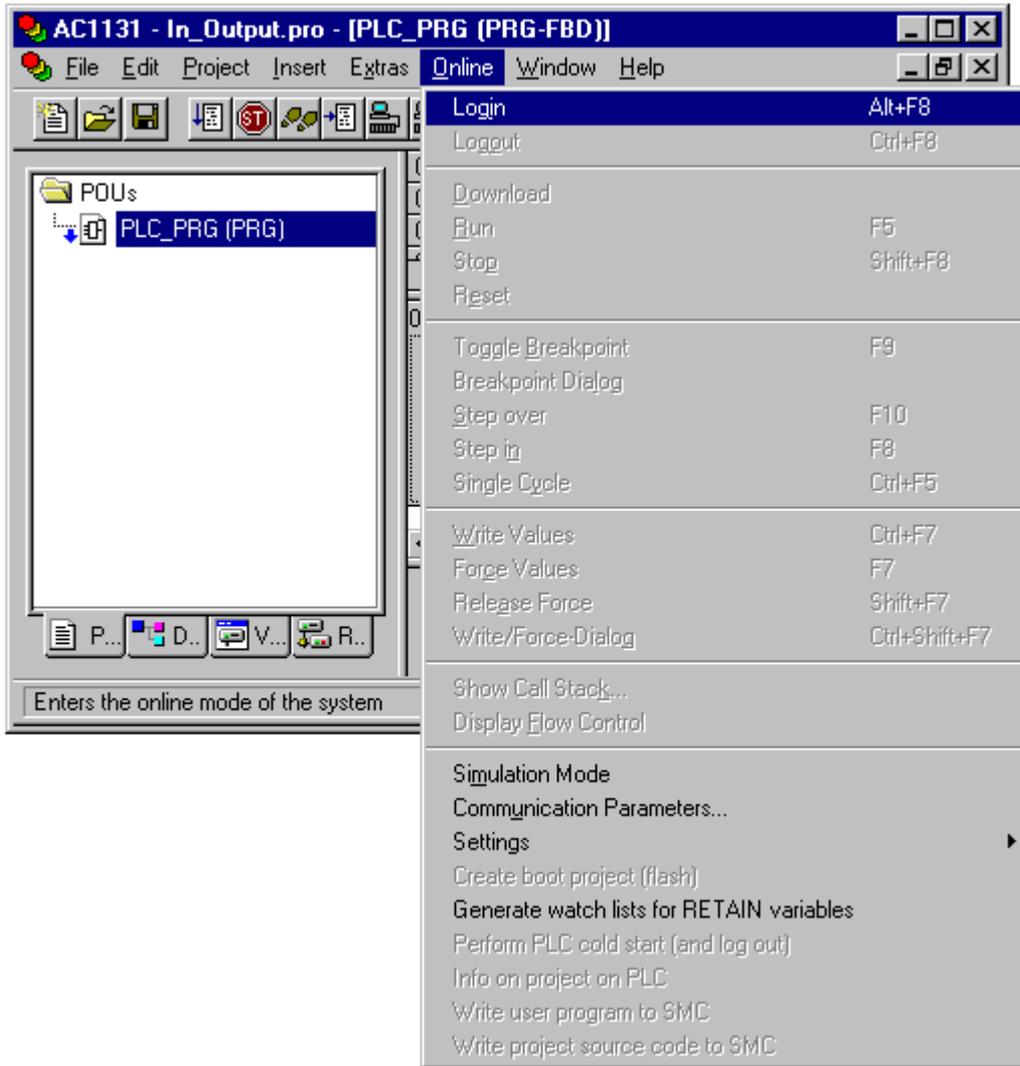
The gateway icon appears on the right side of the Windows status bar and the controller reports the project information and the runtime system version:
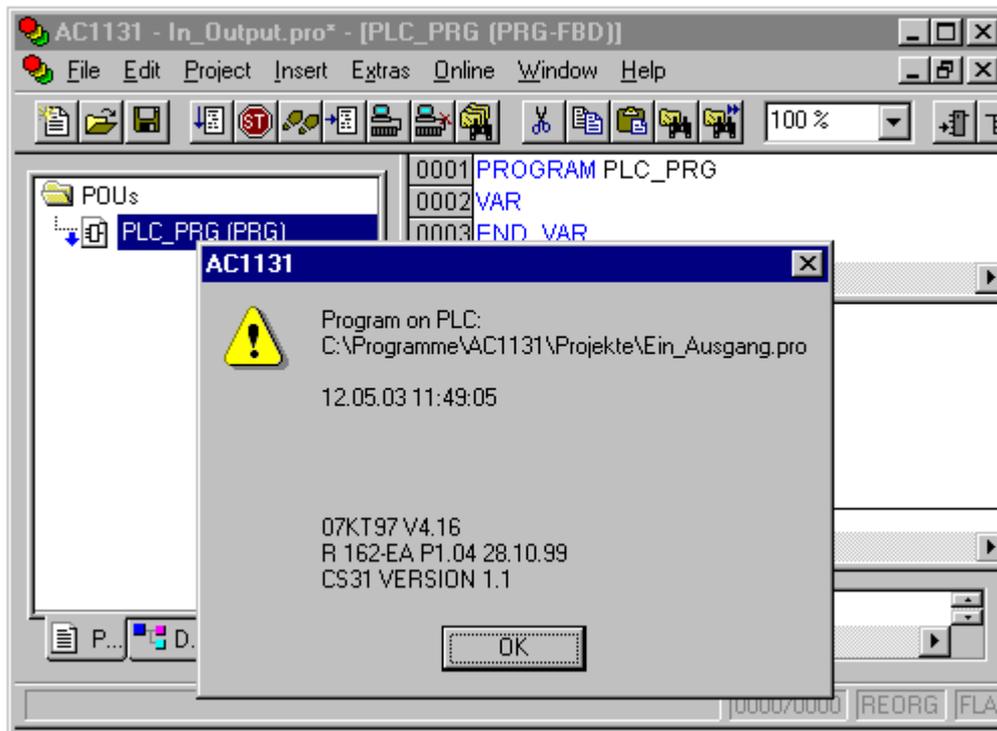
Confirm with "OK".

Now a verification is performed whether the project is identical to the project on the PC. If there are any differences the PC reports the following message:
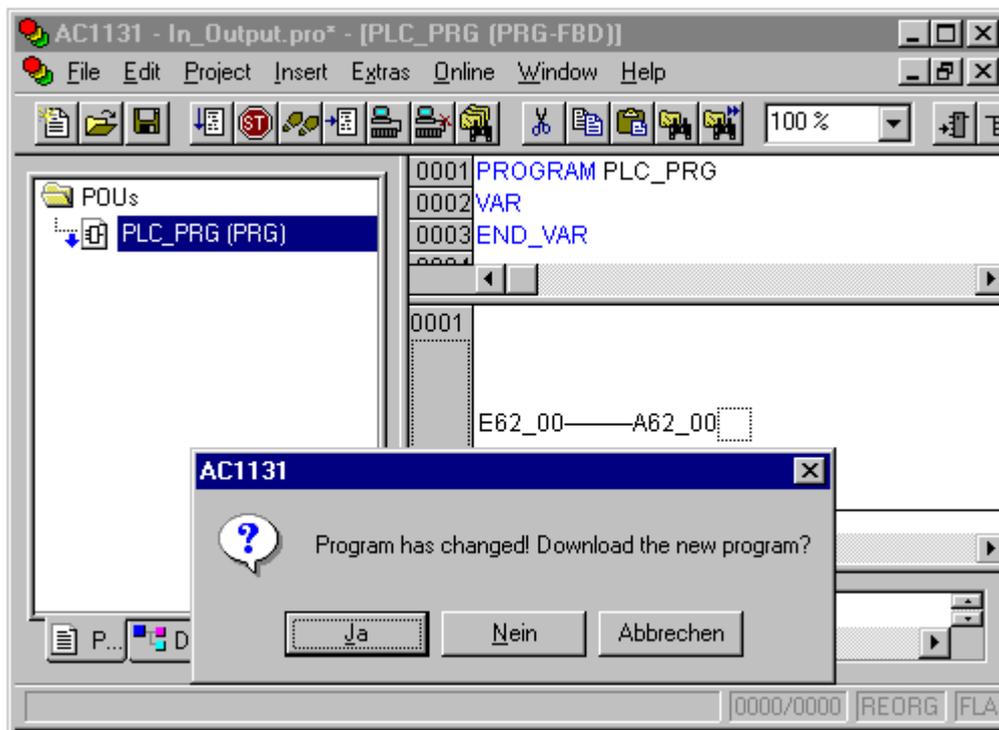
Select "Yes" to load the new program. During the load process a progress display informs about the currently loaded bytes. When starting the load process the PLC changes to the STOP mode.

15. The next step is to store the program in the Flash memory of the PLC to protect it against voltage breakdown. To perform this, please select "Online/Create boot project (flash)":
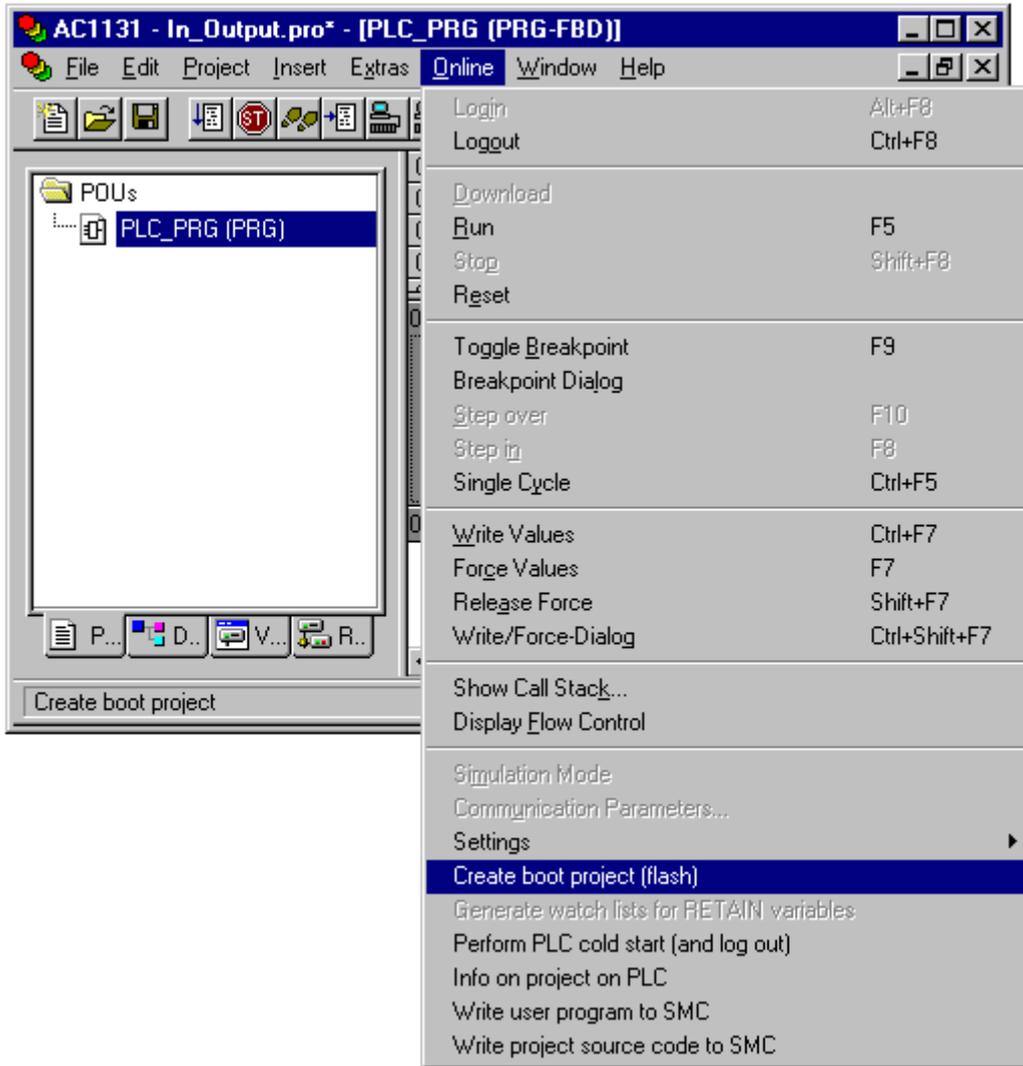
The project is stored to the Flash memory. During the flashing process the entry "FLASH" is displayed in black in the status bar of 907 AC 1131 and the "Batt." LED on the PLC flashes.

16. Now the program has to be started. For this, select "Online/Run":
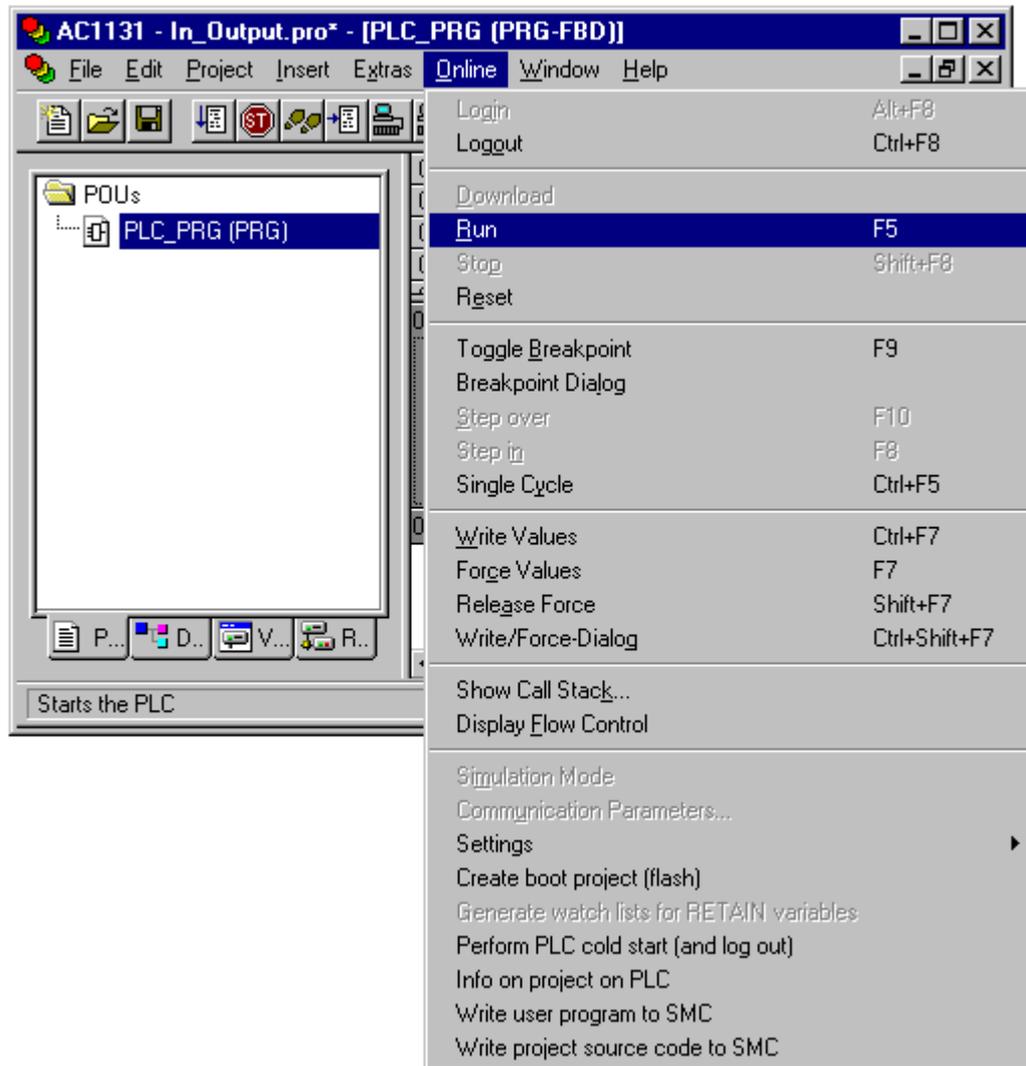
The status bar of 907 AC 1131 displays "RUN" in black color and the "RUN" LED on the PLC lights up.

17. In order to test the program we are going to force input E62_00 to TRUE. To do this, please double click on the icon "E62_00". Next to "E62_00" the entry "TRUE" is displayed. After performing another double click, "FALSE" is displayed. Double clicking once more releases forcing. This way you can choose the desired value for forcing. In our example "TRUE" is used.
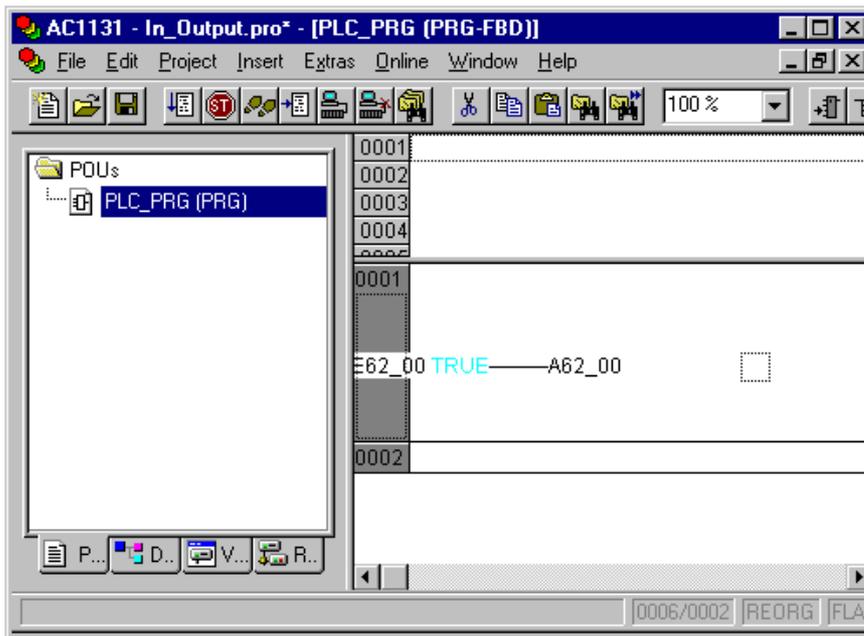
Now the variable E62_00 is prepared for forcing.

Press the <F7> key or select "Online/Force Values" from the menu. This will force the variable. On the PC, A62_00 becomes blue and the output A62_00 of the PLC is connected through. The LED for output A62_00 lights up. The "RUN" LED on the PLC flashes to indicate the process of "Forcing" (as well as "Overwriting", "Single step", "Breakpoint", etc.).
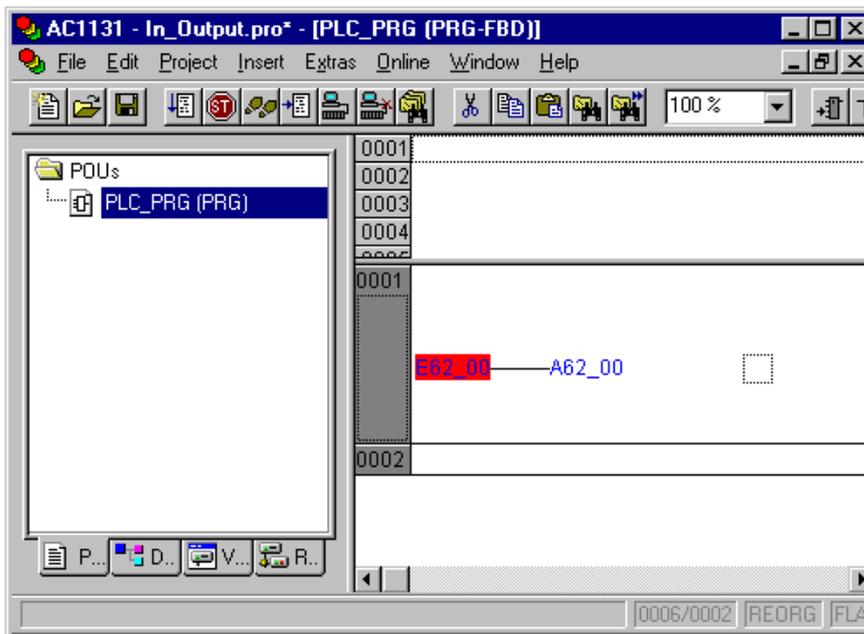
Now you can force the output to FALSE again or "Release Force". Using "Online/Logout" the connection to the PLC is disconnected again. To exit the software 907 AC 1131 use "File/Exit".

2 Special functions of 907 AC 1131

2.1 Entering the PLC into the controller configuration

From version V4.1 of the software 907 AC 1131 on, the PLC type is entered into the controller configuration. This is necessary to adapt the software to the various controllers, e.g.
- for different memory areas of the controllers 07 KT 95..97 and 07 KT 98 or
- to distinguish series 90 and series 40...50.

☞ **Note:**

The controllers 07 KT 95, 07 KT 96 and 07 KT 97 use the same memory areas. When creating projects for those controllers you can enter "07KT95", "07KT96" **or** "07KT97" into the controller configuration. If you want to load the same project into the controller 07 KT 98, you have to modify the controller configuration entry to 07KT98.

When logging into the controller a verification is performed whether the entry in the controller configuration matches the controller. If this is not the case an error message is displayed:
"The connected CPU is not 100% compatible to the CPU used in the controller configuration. It is not possible to go online. Please replace the CPU module!"

If you try to translate a project without any entry in the controller configuration, the following error message is displayed:
"No CPU set. Please enter a CPU module into the controller configuration."
In this case it is also possible that various secondary errors are signalized.

2.1.1 How to enter the PLC into the controller configuration

There are several possibilities to enter the PLC into the controller configuration:

1. **Entering the PLC into a new project**

When creating a new project (see section "Creating a new project for the 07 KT 97") you are requested for the PLC after selecting "File/New". The selected PLC is entered into the "PLC Configuration" folder in the "Resources" tab.
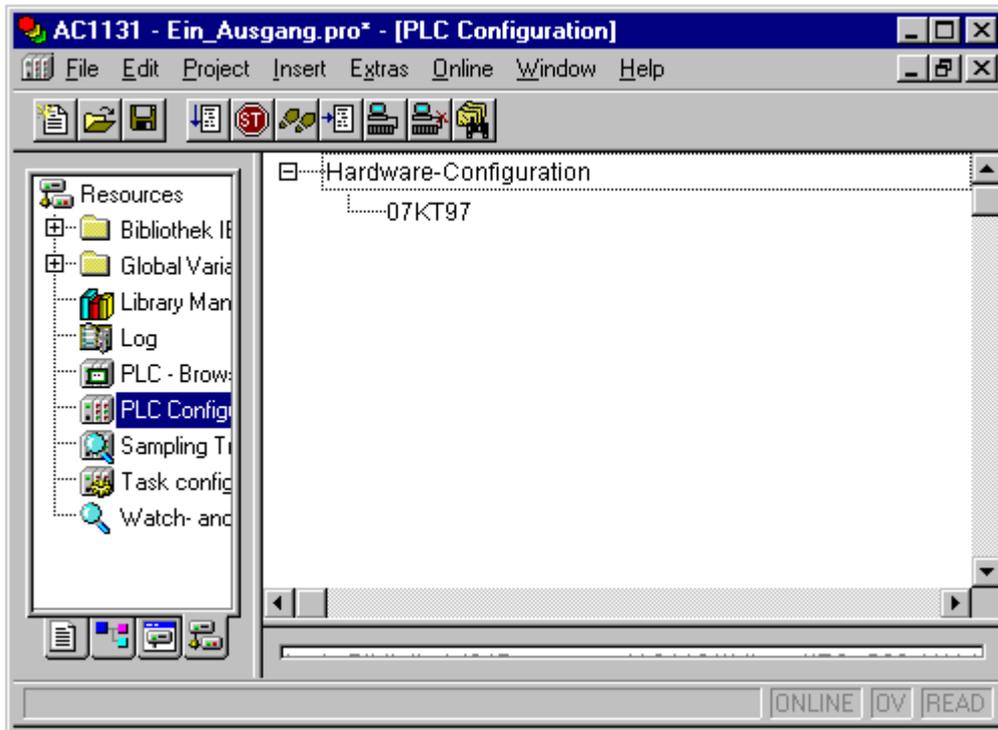
2. **Entering the PLC into an existing project (e.g. created in V4.0)**

Proceed as follows to enter the controller into an existing project:
- Switch to the "Resources" tab.
- Double click on the "PLC Configuration" folder to open it.
- In the window on the right, right click on "Hardware Configuration". From the appearing context menu select "Append Subelement" and then the desired PLC. Confirm the subsequently appearing window with "OK".
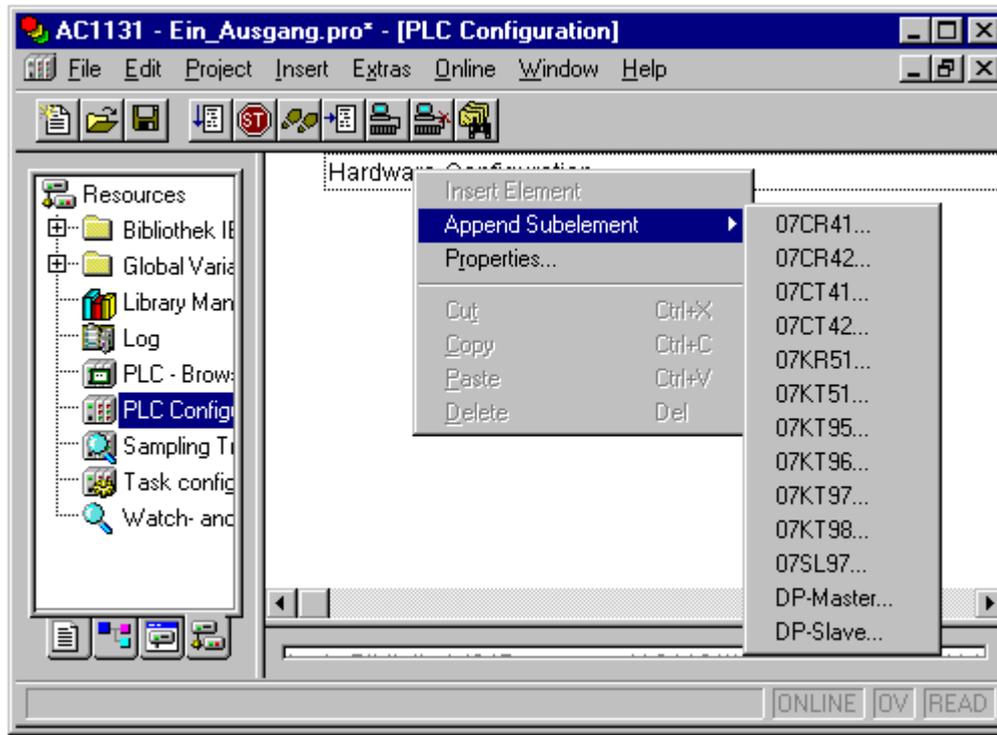
2.1.2 Changing the PLC for a project

If you want to load a project which was created for the controllers 07 KT 95..97 into the controller 07 KT 98 (or vice versa) you have to modify the entry in the controller configuration accordingly.

For this, the following steps are required:

- Switch to the "Resources" tab.
- Double click on the "PLC configuration" folder to open it.
- If the "Hardware Configuration" entry is preceded by a plus sign, open it by clicking on the plus sign to make the subelements visible.
- Select the controller (e.g. 07 KT 97) and press the <Del> key (or <Entf>) or open the context menu by clicking with the right mouse button and select "Delete".
- Now the controller configuration is empty again. Open the context menu once again by clicking with the right mouse button. Select "Append Subelement" and then the desired PLC. Confirm the subsequently appearing window with "OK".

2.1.3 Configuration files for the controllers

The configuration files for the controllers are located in the subdirectory "Library/PLCCONF". These files are the GSD files for the PROFIBUS configurator as well as the *.con files for the individual controllers.

Controller	Series	Configuration file
07 KT 95	S90	CPUKT95.con
07 KT 96	S90	CPUKT96.con
07 KT 97	S90	CPUKT97.con
07 KT 98	S90	CPUKT98.con
07 SL 97	S90	CPUSL97.con
07 CR 41	S40..50	CPUCR41.con
07 CR 42	S40..50	CPUCR42.con
07 CT 41	S40..50	CPUCT41.con
07 CT 42	S40..50	CPUCT42.con
07 KR 51	S40..50	CPUKR51.con
07 KT 51	S40..50	CPUKT51.con

The file CPUABB.SEL contains all selectable controllers.
Please do not modify this file!!!

The **configuration files for the controllers 07 KT 95..97** contain the following parameters and values:

```
; Basic address %M area
Param1=BaseMerker;DWORD:=16#1b8000;16#1b8000;16#1b8000;

; Basic address %I area
Param2=BaseInputs;DWORD:=16#2f8000;16#2f8000;16#2f8000;

; Basic address %Q area
Param3=BaseOutputs;DWORD:=16#2fc000;16#2fc000;16#2fc000;

; Basic address RETAIN variable
Param4=BaseRetains;DWORD:=16#1C0000;16#1C0000;16#1C0000;

; Size of user program (480 kB-16 kB I/O description = 464 KB)
Param5=CodeSize;DWORD:=16#74000;16#74000;16#74000;

; Size of user program without RETAIN (240kB)
Param6=DataSize;DWORD:=16#3c000;16#3c000;16#3c000;

; Size of RETAIN variable (16kB)
Param7=RetainSize;DWORD:=16#4000;16#4000;16#4000;

; Segment size user data (32kB)
Param8=DataSegmentSize;DWORD:=16#8000;16#8000;16#8000;

; Maximum Number of POUs (1024)
Param9=MaxNumofPOUs;DWORD:=1024;1024;1024
```

Please do not modify these files!!!

The **configuration file for the controller 07 KT 98** contains the following parameters and values:

```
; Basic address %M area
Param1=BaseMerker;DWORD:=16#02800000;16#02800000;16#02800000;

; Basic address %I area
Param2=BaseInputs;DWORD:=16#710000;16#710000;16#710000;

; Basic address %Q area
Param3=BaseOutputs;DWORD:=16#714000;16#714000;16#714000;

; Basic address RETAIN variable
Param4=BaseRetains;DWORD:=16#02810000;16#02810000;16#02810000;

; Size of user program (1 MB := 1024 kB-64 kB I/O description = 960 kB)
Param5=CodeSize;DWORD:=16#0F0000;16#0F0000;16#0F0000;

; Size of user data without RETAIN
; Standard:   1MB:=1024kB:=16#100000
Param6=DataSize;DWORD:=16#100000;16#100000;16#100000;

; Size of RETAIN variable (256kB)
Param7=RetainSize;DWORD:=16#40000;16#40000;16#40000;

; Segment size user data
; max.:        256 kB := 16#40000
; possible:     64 kB := 16#10000
;             128 kB := 16#20000
; standard:     32 kB :=16#8000
Param8=DataSegmentSize;DWORD:=16#8000;16#8000;16#8000;

; Maximum Number of POUs (4096 from V5.0 onwards)
Param9=MaxNumofPOUs;DWORD:=1024;1024;1024
```

For the controller 07 KT 98 the parameter "Param8" can be modified if necessary (if a POU in the project requires more than 32 kB of data). In this case the modification applies to all projects.

2.1.4 Displayed code and data size

Maximum code size of the controllers:

For the controllers **07 KT 95..97** the maximum program size is 464 kB = 475136 bytes = 16#74000 bytes. The I/O description takes additional 16 kB which results in a maximum user program size of 480 kB.

For the controller **07 KT 98** the maximum program size is 960 kB = 983040 bytes = 16#0F0000 bytes. The I/O description takes additional 64 kB which results in a maximum user program size of 1 MB = 1024 kB.

Maximum data size of the controllers:

The data segment size is determined in the configuration file by the parameter **Param8** and is set to **32 kB := 32768 bytes = 16#8000 bytes**.
The number of data segments is set in the 907 AC 1131 software for each project using "Project"=>"Options"=>"Build"=>"Number of Data Segments". The default setting is 2 which means that at least two data segments have to be defined.

The variables of the RETAIN segment, the inputs / outputs (%I-, %Q-) and the variables from the addressable flag area (%M-) are not contained in the displayed data size.

For the controllers **07 KT 95..97** the size of the variable memory (refer to parameter Param6 in the configuration file) is 240 kB = 245760 bytes. Subtracting 12432 bytes of internal extra code results in 233328 bytes. These are approx. 7 data segments of 32768 bytes each. Following this, you can set the number of data segments to a maximum of 8 (7 data segments + 1 RETAIN segment).

For the controller **07 KT 98** the size of the variable memory (refer to parameter Param6 in the configuration file) is 1 MB := 1024 kB := 1048576 bytes. Subtracting 12432 bytes of internal extra code results in 1036144 bytes. These are approx. 31 data segments of 32768 bytes each. Following this, you can set the number of data segments to a maximum of 32 (31 data segments + 1 RETAIN segment). If you change the size of the data segments, the maximum number of possible data segments changes accordingly.

Displayed code and data size for version V4.x of 907 AC 1131:

When translating a project using the function key <F11> or "Project/Rebuild All" in one of the versions V4.x of the 907 AC 1131 software, the code and data size information is displayed in the message window in the next to last line.

For our example project "Ein_Ausgang.pro":
Code Size 1065, Data Size 45200

Code size - program size:

The displayed code size corresponds exactly to the size of the program code, but without I/O description (inputs and outputs for the tasks).

Data size - amount of data:

For the data size the size of the required data segments is displayed. Additionally 12432 bytes are required for internal extra code (tables for POU start addresses and lengths, auxiliary functions for online change, debugging, etc.).

Following this, a displayed data size of "Data Size 45200" means:
- 1 data segment with 32768 bytes: 1 *   32768 bytes
- internal extra code   12432 bytes

_____

- data size   45200 byte

The actually used data in the data segment are not displayed. The following error message is displayed if you enter more data than a data segment can accommodate: "The data occupy too much memory. Increase the number of data segments."

Displayed code and data size for 907 AC 1131 version V5.0 onwards:

From 907 AC 1131 version V5.0 on, the code and data size is not known until all used POUs have been built and linked and the initialization code has been created. This is not done before you have answered the inquiry "Do you want to load ... ?" in online mode with <Yes>.

This is why the code and data size is displayed in the message window after loading the project to the controller.

For example, the following example message is displayed for an "empty" project and 2 data segments:
Code Size: 993 Data Size: 12470
Used global data 38 of 32768

Code size - program size:

The displayed code size corresponds exactly to the size of the program code, but without I/O description (inputs and outputs for the tasks).

Data size - amount of data:

For the data size, the basic need of 12470 bytes (12432 bytes for internal extra code + 38 bytes in the 2nd segment for tables of POU start addresses and lengths, auxiliary functions for online change, debugging, etc.) + full data segments + filled part of the data segment used last is displayed.

Example:
Code Size: 2581 Data Size: 14297
Used global data 1865 of 32768

Data Size: 14297 in the example means: 12432 + 38 + 1827 = 14297
Used global data 1865 of 32768 means: 38 + 1827 bytes for project variables and two data segments are set with 32 kB=32768 bytes. Consequently, 1827 bytes of variables are declared.

Therefore, the number of implemented variables can be calculated as follows:
Planned variables = Data Size – 12470 [bytes]

"Used global data %d of %d" is only displayed if the last data segment contains data. It is not displayed if the number of data segments is increased to 3 in the example.

Full segments are always calculated with their full size, even if they still may have some bytes free.

The improvement compared to version V4.x is that unused data segments are no longer calculated and that only the actually used bytes of the data segment last used are calculated. A disadvantage is that due to the new compile process the size is not known until the offline mode is activated.

2.2   Special functions in the "Online" main menu

2.2.1   Overview of functions

The following menus contained in the main menu **"Online"** are not described in the operating instructions of the 907 AC 1131 software:

- Menu "Create boot project (flash)" (refer to chapter 2.2.2)
- Menu "Settings"=>"Settings for force" (refer to chapter 2.2.3)
- Menu "Perform PLC cold start (and log out)" (refer to chapter 2.2.4)
- Menu "Info on project on PLC" (refer to chapter 2.2.5)
- Menu "Write user program to SMC" (refer to chapter 2.2.6)
- Source download to a SmartMedia card (refer to chapter 4.3)

2.2.2   Menu "Create boot project (flash)"

This menu is only available in the logged in state. In the offline state the menu is grayed out. After selecting this menu item the program stored in the RAM memory of the PLC is written to the Flash memory of the PLC resulting in a voltage breakdown-safe storage. During the flash process the indication "FLASH" is displayed in black in the status bar of the 907 AC 1131 software (otherwise the indication is displayed in gray). The LED "Batt." on the PLC flashes.

☞   **Note:**
If the controller is in RUN mode, this operation can take some minutes.

Together with the user program the areas of the constants %MW3000.0..%MW3089.15 and the double word constants %MD4000.0..%MD4023.15 as well as the PROFIBUS configuration data and, if necessary, the force list are stored in the flash memory.

2.2.3   Menu "Settings"=>"Settings for force"

Using the menu item "Settings"=>"Settings for force" the forcing parameters "Do not release force when logging out" and "Save force list in boot project (Flash/SMC)" are set. This is described in more detail in section 2.3 "Peculiarities for forcing and overwriting".

2.2.4   Menu "Perform PLC cold start (and log out)"

The menu item "Perform PLC cold start (and log out)" is only available in the logged in state (otherwise it is grayed out). This menu item initiates a cold start on the PLC. A warm start on the controller is initiated using the menu item "Online/Reset". For both menu items a logout is performed. The cold and the warm start of the controller are described in detail in volume 3 - System technology.

2.2.5   Menu "Info on project on PLC"

The menu item "Info on project on PLC" is only available in the logged in state (otherwise it is grayed out). Selecting this menu item displays the same information about the runtime system of the controller and the project on the PLC as displayed when logging in to the controller.

2.2.6 Menu "Write user program to SMC"

The menu item "Write user program to SMC" is only available in the logged in state (otherwise it is grayed out). After selecting this menu item the program stored in the Flash memory of the PLC is written to an inserted SmartMedia Card.

Caution:
The PLC writing the user program to the SmartMedia card must have the same firmware version as the PLC the user program is loaded to. If the firmware versions differ, the SmartMedia card cannot be read and the FK3 error 130 is output.

Writing a user program to a SmartMedia card:

Proceed as follows to write a user program to the SmartMedia card:

1. Boot the PLC without a SmartMedia card inserted.

2. Load the user program to the controller using 907 AC 1131.

3. Use "Online/Create boot project (flash)" to write the program to the Flash memory.

4. Insert the SmartMedia card into the controller (chip points to the top, refer to hardware documentation).

5. Select the command "Online/Write user program to SMC". (The program is loaded from the Flash memory and written to the SmartMedia card.)

Caution:
The SmartMedia Card is formatted prior to writing data to it. Data possibly stored on the card are deleted without any inquiry.

If both parameters under "Online/Settings for force" are checked ("yes"):
- "Do not release force when logging out" - checked
- "Save force list in boot project (Flash/SMC)" - checked
you are asked whether you really want to write the force values to the SmartMedia Card. The menu is aborted if you answer with no.
(Refer to chapter "Peculiarities for forcing and overwriting from version V4.1 onwards").

The following error is displayed if no SmartMedia Card is inserted in the controller:
"The controller has performed the last online service incorrect".

Loading a user program from the SmartMedia card to the PLC:

Proceed as follows to load a user program from the SmartMedia card to the PLC:

1. Switch off the PLC (control voltage OFF).

2. Insert the SmartMedia card into the controller (chip points to the top, refer to hardware documentation).

3. Switch on the control voltage. During the boot process of the PLC, the user program is read from the SmartMedia card and written to the Flash memory of the PLC. Then it is loaded from the Flash to the RAM memory and started.

2.3 Peculiarities for forcing and overwriting

In version V4.0 of the software 907 AC 1131 only one buffer was available for the functions forcing and overwriting. This is why it was only possible to either force **or** overwrite.

From version V4.1 of the software 907 AC 1131 onwards, separate buffers are used for forcing and overwriting. Furthermore it is now possible that the force values are not deleted when logging out or that the force values are even written to the Flash memory or the SMC to become voltage breakdown-safe.

⚠️ **Caution:**

The new features for forcing and overwriting are only available on controllers with a runtime system from version V4.1 onwards. The force parameters described below are without any effects on controllers with runtime system version V4.0x.

Using **"Write Values"** values are written once. After this, the variables can immediately change their value again initiated by the program.

Using **"Force Values"** the values are written at each beginning and each end of a cycle until this process is cancelled with "Release Force". For example when you force a digital input to TRUE, the I/O image of the tasks works with the forced TRUE, independently from the actual status of the input. But, for instance, if you force an INT variable to 0 and increment this variable in the program, the variable's value inside the program is set to 1 (and not 0). This applies to all variables and also to outputs.

The "Online" menu contains the menu item "Settings for Force". If you select this menu item a window containing the force parameters appears.
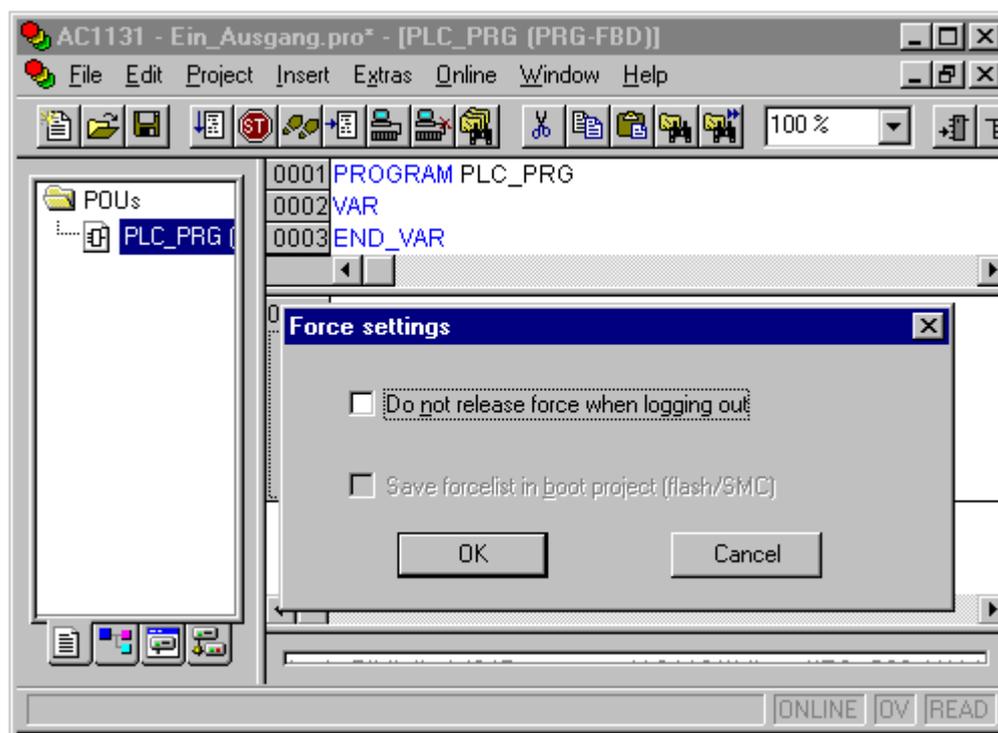
When you click on "Do not release force when logging out" the entry is marked by a checkmark ✓ (indicating that the function is selected) and the entry for the parameter "Save force list in the boot project (Flash/SMC)" is displayed in black and can be selected now.
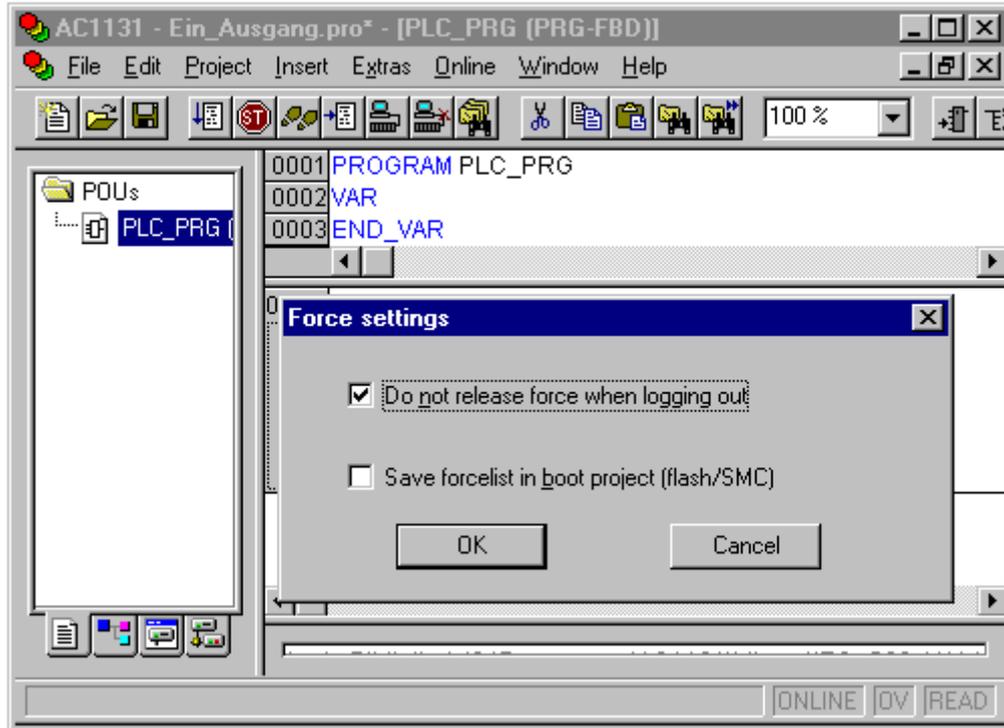
The following behavior can be set using these two parameters:

1. - "Do not release force when logging out" - not checked
   - "Save force list in boot project (Flash/SMC)" - not checked

   This is the **default setting**. The behavior is as for version V4.0 which means that
   - the force values are deleted when logging out and
   - the force values are not stored in the Flash memory or on the SmartMedia Card.
   The LED RUN flashes as long as the logged in state is kept and the values are forced. After logging out the LED lights as usual.

2. - "Do not release force when logging out" - checked
   - "Save force list in boot project (Flash/SMC)" - not checked

   Using this setting the force values are kept in the RAM memory after logging out. When logging out you are asked whether the force values should be kept. If you answer with "No" the force values are deleted. If you answer with "Yes" the force values are kept. In addition, the RUN LED flashes. When logging in, the force list is read from the RAM memory of the PLC. The status bar of 907 AC 1131 displays "FORCE" in black. The forced values are marked in color.
   The force list is deleted on a cold or warm start or when switching the control voltage off/on. (The behavior is as for the 07 KT 94 with EBS operating system). The RUN LED is lighting again.

3. - "Do not release force when logging out" - yes (checked)
   - "Save force list in boot project (Flash/SMC)" - yes (checked)

   You must store the user program to the Flash memory using "Online/Create boot project (flash)" before logging out. Prior to flashing the user program you are again reminded that force values are present and that they are also written to the Flash memory. After this, the force values are saved in the RAM and in the Flash memory. The RUN LED flashes. If you store the user program to a SMC the force values are also stored. You are once again reminded to this fact.
   When performing a cold or warm start or when switching the control voltage off/on the force list is loaded from the Flash memory together with the user program and then activated. The RUN LED flashes.
   The force list can only be deleted using the software 907 AC 1131. When logging in, the force list is read from the RAM memory of the PLC.

4. - "Do not release force when logging out" - not checked
   - "Save force list in boot project (Flash/SMC)" - checked

   This setting is not possible!

2.4 Peculiarities of batch operation

Batch operation with the software 907 AC 1131 is described in appendix F "Command line and command file commands" of volume 10 - Operating instructions S.90/40/50.
In order to enable loading the same project to several controllers using a command file, the batch operation was expanded by **setting the communication parameters**.

Caution:

To obtain a correct function of the batch operation for **write protected projects**, the parameter Autoload in the file **AC1131.INI** must be set to off (**Autoload=No**). By default Autoload is set to on, i.e. Autoload=Yes.
For the 907 AC 1131 versions V4.x, the file AC1131.INI is located in the WINNT folder (for WinNT) or the WINDOWS folder (for Win98 SE). From version V5.0 onwards, the file is located in the main directory of the programming software (e.g. C:\Programs\AC1131).

2.4.1 Example of a command file

Below an example project is introduced together with the corresponding command file. The lines beginning with a semicolon are comment lines. Comments written in command lines are also preceded by a semicolon.

The content of the command file **ABB.CMD** shall be as follows:

; the following line can be omitted if the project has no access passwords
; insert "query off ok" already at this point if the project file is write protected

query off cancel ; ignore the password request - no writing permission

;file open downloadbatch.pro ; **project name** => enter your own project here
file open c:\programme\ac1131\projekte\downloadbatch.pro ; or enter file with path
query off ok ; all following dialogs with OK

online sim off ; switch off the simulation

gateway local ; local access to gateway (shared memory)

; Example for **serial communication**
;device guid {83BC99E0-3455-11d3-A2A8-00104BB37AA1} ; ABB RS232
;device parameter 1000 COM2 ; COM port 2
;device parameter 1001 19200 ; baud rate

; Select **Arcnet driver**
device guid {F41F2C40-346C-11d3-A2A8-00104BB37AA1}

; enter target address of the controller
; address of first controller:
device parameter 1101 35 ; **Target node number 35** (arcnet address)

```
online login ; login to controller
delay 15000 ; wait time for download (15 seconds, long enough?? => change if necessary)
online createbootproject ; start flashing
delay 10000 ; 10 sec for flashing process
online run ; start program
delay 2000
online logout ; logout from the controller
delay 7000 ; 7 sec. for logout


; address of second controller:
device parameter 1101 2 ; target node number 2 (arcnet address)

online login
delay 15000 ; for download (15 seconds, long enough?? => change if necessary)
online createbootproject ; start flashing
delay 10000 ; 10 sec for flashing process
online run
delay 2000
online logout
delay 7000 ; 7 sec. for logout

; repeat from target address for further controllers

file close
file quit
```

In our example the write protected project download.pro is sent via ARCNET to the ARCNET subscribers 35 and 2. The project is flashed in each controller and then started. If necessary, the set wait times must be modified according to the program length.

To **call** the command file:

• Select "Start" => "Run"

• Enter:
  c:\programme\ac1131\ac1131.exe /cmd "c:\programme\ac1131\projekte\abb.cmd"

In this case the command file abb.cmd and the project downloadbatch.pro are located in the directory
c:\programme\ac1131\projekte.

2.4.2 Driver settings for batch operation

In the command file the drivers are selected using the device guid command. The parameters of the drivers are entered using device parameter. Below, the settings for the communication drivers of 907 AC 1131 are described.

☝ **Note:**

The individual drivers are described in volume 15 - System Technology Series 90, chapter "System technology of the basic units / Programming and testing".

Serial driver ABB RS232:

device guid {83BC99E0-3455-11d3-A2A8-00104BB37AA1}; ABB RS232
device parameter 1000 COMx; COM port of the PC with x = COM number (COM1, COM2,..)
device parameter 1001 19200; baud rate 19200 baud

Serial routing driver ABB RS232 Route:

device guid {74CE214D-73BF-49fe-B50F-EBCB7C922529}; ABB RS232 Route
device parameter 1000 COMx; COM port of the PC with x = COM number (COM1, COM2,..)
device parameter 1001 19200; baud rate 19200 baud
device parameter 1002 No; no parity
device parameter 1003 1; 1 stop bit
device parameter 1206 0; routing level (0..2)
device parameter 1207 0; coupler for level 1 (0, internal 1..4, external)
device parameter 1208 0; channel on coupler stage 1 (0..19)
device parameter 1209 0,0,0,0,0; address in target coupler stage 1 (5 bytes max.)
device parameter 1210 0; coupler for level 2 (0, internal 1..4, external)
device parameter 1211 0; channel on coupler stage 2 (0..19)
device parameter 1212 0,0,0,0,0; address in target coupler stage 2 (5 bytes max.)

Serial modem driver Serial (Modem)

device guid {F5DBCE81-D15B-11d2-8D9C-00104BA6F685}; Serial (Modem)
device parameter 1000 COMx; COM port of the PC with x = COM number (COM1, COM2,..)
device parameter 1001 19200; baud rate 19200 baud
device parameter 1002 No; no parity
device parameter 1003 1; 1 stop bit
device parameter 1100 AT&F0; Modem initialization (Init)
device parameter 1101 ATDT03091772108; phone number (Dial)
device parameter 1102 ATH; hangup
device parameter 1103 +++; break
device parameter 1104 60; timeout in [sec]

ARCNET driver ABB Arcnet:

    device guid {F41F2C40-346C-11d3-A2A8-00104BB37AA1}; ABB Arcnet
    device parameter 1100 254; sender node (PC node)
    device parameter 1101 1; target node (PLC node)
    device parameter 1102 2000; receive timeout in [ms]

ARCNET routing driver ABB Arcnet Route:

    device guid {7A58B3F0-110C-11d4-A3CE-00104BB37AA1}; ABB Arcnet Route
    device parameter 1100 254; sender node (PC node)
    device parameter 1101 1; Target node (PLC node)
    device parameter 1102 2000; receive timeout in [ms]
    device parameter 1106 0; routing level (0..2)
    device parameter 1107 0; coupler for level 1 (0, internal 1..4, external)
    device parameter 1108 0; channel on coupler stage 1 (0..19)
    device parameter 1109 0,0,0,0,0; address in target coupler stage 1 (5 bytes max.)
    device parameter 1110 0; coupler for level 2 (0, internal 1..4, external)
    device parameter 1111 0; channel on coupler stage 2 (0..19)
    device parameter 1112 0,0,0,0,0; address in target coupler stage 2 (5 bytes max.)

ARCNET routing driver with adjustable package size ABB Arcnet Route fast:

    device guid {DF13B5A0-B507-11d6-A6B0-00104B548006}; ABB Arcnet Route fast
    device parameter 1100 254; sender node (PC node)
    device parameter 1101 1; target node (PLC node)
    device parameter 1102 2000; receive timeout in [ms]
    device parameter 1106 0; routing level (0..2)
    device parameter 1107 0; coupler for level 1 (0, internal 1..4, external)
    device parameter 1108 0; channel on coupler stage 1 (0..19)
    device parameter 1109 0,0,0,0,0; address in target coupler stage 1 (5 bytes max.)
    device parameter 1110 0; coupler for level 2 (0, internal 1..4, external)
    device parameter 1111 0; channel on coupler stage 2 (0..19)
    device parameter 1112 0,0,0,0,0; address in target coupler stage 2 (5 bytes max.)
    device parameter 1113 128; data package size (480 bytes max.)

PCI driver for 07 SL 97 ABB SL97:

        device guid {27A33940-3190-11d4-A3ED-00104BB37AA1}; ABB SL97
        device parameter 1100 0; PCI card number
        device parameter 1101 2000; timeout in [ms]
        device parameter 1102 0; routing level (0..2)
        device parameter 1103 0; coupler for level 1 (0, internal 1..4, external)
        device parameter 1104 0; channel on coupler stage 1 (0..19)
        device parameter 1105 0,0,0,0,0; address in target coupler stage 1 (5 bytes max.)
        device parameter 1106 0; coupler for level 2 (0, internal 1..4, external)
        device parameter 1107 0; channel on coupler stage 2 (0..19)
        device parameter 1108 0,0,0,0,0; address in target coupler stage 2 (5 bytes max.)

PCI driver with adjustable data package size for 07 SL 97 ABB SL97 fast:

        device guid {B62D8BE5-A2B0-4095-B20F-2C5AF8DFB822}; ABB SL97 fast
        device parameter 1100 0; PCI card number
        device parameter 1101 2000; timeout in [ms]
        device parameter 1102 0; routing level (0..2)
        device parameter 1103 0; coupler for level 1 (0, internal 1..4, external)
        device parameter 1104 0; channel on coupler stage 1 (0..19)
        device parameter 1105 0,0,0,0,0; address in target coupler stage 1 (5 bytes max.)
        device parameter 1106 0; coupler for level 2 (0, internal 1..4, external)
        device parameter 1107 0; channel on coupler stage 2 (0..19)
        device parameter 1108 0,0,0,0,0; address in target coupler stage 2 (5 bytes max.)
        device parameter 1109 128; data package size (480 bytes max.)

Ethernet driver ABB Tcp/Ip Level 2:

        device guid {0379CE33-B643-40ba-8F61-2DDE4C147BC2}; ABB Tcp/Ip Level 2
        device parameter 1100 169.254.24.28; IP address of the PLC
        device parameter 1101 1200; IP port
        device parameter 1102 2000; timeout in [ms]
        device parameter 1106 0; routing level (0..2)
        device parameter 1107 0; coupler for level 1 (0, internal 1..4, external)
        device parameter 1108 0; channel on coupler stage 1 (0..19)
        device parameter 1109 0,0,0,0,0; address in target coupler stage 1 (5 bytes max.)
        device parameter 1110 0; coupler for level 2 (0, internal 1..4, external)
        device parameter 1111 0; channel on coupler stage 2 (0..19)
        device parameter 1112 0,0,0,0,0; address in target coupler stage 2 (5 bytes max.)
        device parameter 1113 128; data package size (1430 bytes max.)

2.5   Notes concerning frequently asked questions

2.5.1   Project protection for logging in to the controller

The following procedure is recommended to ensure that a project cannot be modified unintentionally and that a login to the controller can be performed again and again without producing the message "Controller contains other project..." (and causing the controller to STOP on "OK").

1.   - Translate the project using <F11> or "Project/Rebuild All".
     - Save the project.
     - Set up a password by selecting:
     "Project/Options/Passwords/Write Protection Password"
     (e.g. enter 'AC31' and repeat for confirmation)
     - Exit the 907 AC 1131 software.

2.   - Start the 907 AC 1131 software.
     - Open the project (or automatically opened).
     - **Do not** enter the password. Confirm with <Cancel>.
     - Login to controller 1, "Controller contains other project...", download, bootproject, start, logout.
     - Login to controller 2, ...
     Exit → The message "Project was changed. Save..." appears.

3.   - Start the 907 AC 1131 software.
     - Open the project (or automatically opened).
     - **Do not** enter the password. Confirm with <Cancel>.
     - Login to controller 1 (without message "Controller contains other project...").
     - Login to controller 2, ...

If you want to call the following menus in a write protected project (as described above), you can update the menus using <F11> or "Project/Rebuild All".
- "Project/Show Call Tree ..."
- "Project/Show Cross Reference ..."
- "Project/Show unused Variables ..."
The project will not be changed so that it is again possible to login.
Please **do not** enter the write protection password when opening the project!

2.5.2 Creating and modifying your own library

Creating a library:

Proceed as follows to create a library containing self-made POUs:

1. First a project is created containing the POUs.

2. The PLC_PRG can contain all the POU calls. Consequently the program can be tested.

3. The project is saved for testing.

4. All parts of the project which are not required for this library are deleted:
   - the POU PLC_PRG in the "POUs" tab and
   - the objects in the "Global Variables" folder of the "Resources" tab (such as "Auslastung", "Fehlermeldungen", "Systemkonstanten", "KT97_Onboard", etc.).

5. For the number of data segments (which can be found under "Project/Options/Build/Number of data segments") "2" must be entered.

6. The "PLC Configuration" in the "Resources" tab must contain a PLC of series 90 (e.g. 07KT97).

7. Now the project is built one more time using the <F11> key or "Project/Rebuild All". When using one of the versions V4.x, an error message occurs:
   "The project does not contain a block PLC_PRG (vector function)".

8. To protect the library with a password, select "Project/Options/Passwords" and enter the desired password in the fields "Password" and "Confirm Password".

9. To save the library, use
   "File/Save as...".
   Select "File type - Internal library (*.lib)" in the appearing dialog. The desired name for the library is entered in the "File name" field. The file name extension LIB is automatically attached.

Modifying a library:

To modify a library which was created this way, use "File/Open", select the file type "Library (*.lib)" and enter the file name to open the library.

Now existing POUs can be modified or deleted and/or new POUs can be added. To save the library perform the steps 5, 6, 7 and 9 as described above in section "Creating a library".

Caution:
The manufacturer libraries are password protected and may not be modified by the user.

Library Base_free_S90_V41.LIB:

From 907 AC 1131 version V4.3 onwards, the library Base_free_S90_V41.LIB is provided in addition to the library Base_S90_V41.LIB. This library contains all POUs of the Base_S90_V41.LIB. However, it is not password protected. This enables the user to create own libraries.

To create your own library from the Base_free_S90_V41-LIB perform the following steps:

1. Open the library by selecting:
- "File"=>"Open" with: File type=Library (*.lib) and file name=Base_free_S90_V41.LIB

2. Delete all POUs or folders you do not need:
- Select the POU/folder, press the right mouse button and then select "Delete object". Under "Resources"=>"PLC Configuration", enter a series 90 CPU.

4. Under "Project"=>"Options"=>"Build" enter a 2 as the number of data segments.

5. Translate the project by pressing the <F11> key.
For each POU the warning message "This expression does not contain an assignment. No code generated" is generated.
Version V4.3 still displayed the error (which had to be ignored) "Project does not contain the POU PLC_PRG".

6. Save the library by selecting:
- "File"=>"Save as..." with file type=external library and file name: your_file_name.

7. Close the library by selecting:
- "File"=>"Close".
In version V4.3 the inquiry for "Save changes" has to be answered with "No". In version V5.0 this inquiry does no longer exist.

Note:
It is not possible to insert new POUs (e.g. user POUs) into the library because the library is an external library and the POU code is stored in the runtime system of the PLC. Due to this, the POUs are also not processed in the simulation.

3   The initialization file of 907 AC 1131 - AC1131.ini

3.1   Parameter overview

In the V4.x versions of 907 AC 1131, the initialization file **AC1131.INI** for the 907 AC 1131 software is loaded to the system directory together with the software:
WINNT\System32   for Windows NT (from V4.0 SP4 onwards), 2000 and XP
or
WINDOWS\System   for Windows 98 SE.

From version V5.0 on the initialization file is located in the main directory of 907 AC 1131.

Caution:
In order to keep the capability of undoing faulty modifications always create a copy of the file before modifying this file.

Most of the parameters are automatically updated or entered together with the 907 AC 1131 software. Parameters which are relevant for the user are described below. These parameters are:

- InitZero=No

- DebugCode=Yes

- UseEmuLib=Yes

- MaxNumOfPOUs=1024 (no longer relevant from V4.3 onwards)

- KT94WaitTime=2200

- UpdateRate=200

- DownloadWaittime=22222

- Autoload=Yes

- NestedCommentsByDefault=No

- RetainWatchLists=Yes

- POULanguage=ST

- LibraryDir=c:\Programme\AC1131\Library

- IOConfigDir=C:\Programme\AC1131\Library\PLCCONF\

- ProjectsDir=c:\Programme\AC1131\Projekte

- ImportExportDir=c:\Programme\AC1131\Library\

- DocuframeDir=c:\Programme\AC1131

- TraceDir=c:\Programme\AC1131\Projekte

- WatchDir=c:\Programme\AC1131\Projekte

- Project0=c:\Programme\AC1131\Projekte\Ein_Ausgang.pro

- Project1=c:\Programme\AC1131\Projekte\Bspdt.pro

- Project2=c:\Programme\AC1131\Projekte\Ampel.pro

3.2 Parameter descriptions

3.2.1 Parameters for setting the compiler

Parameter InitZero

If InitZero=Yes, initialization code is created for all variables (including all variables which are initialized by zero or FALSE). In this case the program code becomes considerably larger. If InitZero=No, initialization code is only created for variables which are initialized by TRUE or not equal to zero. When the controller is started the non-buffered memory areas are initialized by zero (refer to volume 15 "System technology basic units, Initialization and buffering of data areas").
Default value: **InitZero=No**

Parameter DebugCode:

Code for which all the online functions such as breakpoint, single step, etc. are possible is only created if the setting is DebugCode=Yes. If the parameter is set to DebugCode=No the program code is becoming smaller but the program cannot be tested any more using all online functions.
Default value: **DebugCode=Yes**

Parameter UseEmuLib:

UseEmuLib=Yes uses the emulation of the floating point functions in the run time system of the PLC. For UseEmuLib=No, the calling of the emulator is substituted by direct function calls. This **considerably accelerates** the processing of floating point functions.
The setting applies to all projects.
Default value: **UseEmuLib=Yes**

Caution:
UseEmuLib=No (fast floating point functions) can be used from the PLC run time system version V4.16 onwards. If UseEmuLib=No is used on earlier run time systems, a message that not all functions can be assigned appears during the program startup.

From 907 AC 1131 version V5.0 onwards, the parameter setting is displayed in the message window during project translation.

```
Compilation was done with UseEmuLib=Yes
Hardware-Configuration
0 Error(s), 0 Warning(s).
```
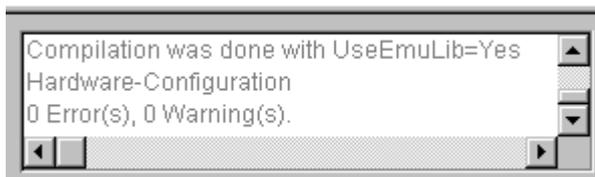
Parameter MaxNumOfPOUs:

The number of POUs contained in a project is composed as follows:
- planned programs - 1 POU each
- planned functions - 1 POU each
- planned function blocks - 2 POUs each
- planned data types - 1 POU each
- functions in libraries - 1 POU each
- function blocks in libraries - 2 POUs each

In the older 907 AC 1131 versions up to V4.2, the maximum number of POUs was set using the parameter MaxNumOfPOUs. According to the runtime system of the AC31 controllers it was fixed to 1024.

From 907 AC 1131 version V4.3 on, the maximum number of POUs is no longer read from the AC1131.INI file, but entered for each controller in the corresponding controller configuration file. (Refer to chapter 2.1.3 "Configuration files for the controllers".)

3.2.2 Parameters for configuring the communication with the PLC

Parameter KT94WaitTime:

The parameter KT94WaitTime is used to set the wait time until a communication error is reported in case of serial communication disturbances.

This parameter is only effective for drivers which do not have an own "Timeout" parameter.

Default value: **KT94WaitTime=2200** (corresponds to approx. 10 sec.)

Parameter UpdateRate:

The parameter UpdateRate is used to set the refresh time in milliseconds for the variables.

Default value: **UpdateRate=200** (smaller values are not possible)

Parameter DownloadWaittime:

The parameter DownloadWaittime is used to set the download timeout interval.
This parameter is available from 907 AC 1131 version **V5.0** onwards.

Default value: **DownloadWaittime=22222** (corresponds to approx. 20 sec.)

3.2.3 Parameters for 907 AC 1131 set-up

Parameter Autoload:

> If Autoload=Yes the project opened last is automatically opened again when the 907 AC 1131 is started. For batch operation on read-only projects this parameter must be set to Autoload=No (refer to chapter 2.4 "Peculiarities of batch operation").
>
> Default value: **Autoload=Yes**

Parametrer NestedCommentsByDefault:

> This parameter is used to set the default setting for nested comments. The default setting is "no nested comments". This setting should not be changed. Opening projects created in version V4.x with the setting "NestedCommentsByDefault=Yes" can cause numerous errors during translation.
> In 907 AC 1131 this parameter can be set for the project by selecting
> "Project"=>"Options"=>"Build".
>
> Default value: **NestedCommentsByDefault=No**

Parameter RetainWatchLists:

> This parameter is used to unlock the menu option "Online"=>"Remanent Variables"=>"Generate watch lists for RETAIN variables" (refer to chapter 4.2.2 "RETAIN variables - note for program change").
>
> Default value: **RetainWatchLists=Yes**

Parameter POULanguage:

> This parameter is used to set the default IEC language when inserting a POU.
>
> Default value: **POULanguage=ST** (Structured Text)

Parameter Autossave:

> The "Autosave" parameter is used to switch the automatic project saving function on or off. The parameter is set with the menu item "Project/Options/Load & Save". Autosave should only be switched on during longer editing sessions (offline).
>
> Default value: **Autosave=No**

3.2.4 Parameters for setting the directories

In 907 AC 1131 the following directories are set using the menu item
"Project"=>"Options"=>"Directories".
- **General:**
libraries, translation files, upload filed.

From Version V5.0 on, the following additional path settings can be set for each project:
- **Project:**
libraries, translation files.

Parameter LibraryDir:

The LibraryDir parameter is used to set the library path. The parameter is set with the menu
item "Project/Options/Directories/Libraries".

Default value: **LibraryDir=c:\Programme\AC1131\Library**

Parameter IOConfigDir:

The parameter IOConfigDir is used to set the path for the PLCCONF directory.

Default value: **IOConfigDir=C:\Programme\AC1131\Library\PLCCONF**

Parameter ProjectsDir:

The ProjectsDir parameter is used to set the project path.

Default value: **ProjectsDir=c:\Programme\AC1131\Projekte**

Parameter ImportExportDir:

The parameter ImportExportDir is used to set the path for *.exp files.

Default value: **ImportExportDir=c:\Programme\AC1131\Library**

Parameter DocuframeDir:

The parameter DocuframeDir is used to set the path for the online help system files. From
907 AC 1131 version V5.0 on, the directories
...\Help\German and ...\Help\English containing the German and English help files are
subdirectories of the path set with the DocuframeDir parameter.
This way, changing the language also calls the language-specific help system.

Default value: **DocuframeDir=c:\Programme\AC1131**

Parameter TraceDir:

The parameter TraceDir is used to set the path for trace files.

Default value: **TraceDir=c:\Programme\AC1131\Projekte**

Parameter WatchDir:

The parameter WatchDir is used to set the path for the watch lists *.WTC.

Default value: **WatchDir=c:\Programme\AC1131\Projekte**

Parameter Project0..Project9:

The file names of the projects opened last and their paths are entered to the parameters Project0 ..Project8. If Autoload=Yes the project in parameter Project0 is loaded when starting the 907 AC 1131 software.
Example: **Project0=c:\Programme\AC1131\Projekte\Ein_Ausgang.pro**

Note:
The parameters Project0.. can be deleted in case a project cannot be opened any more.

4   New functions of 907 AC 1131 from version V5.0 onwards

4.1   Password protection of the PLC with a write protection password

From version V5.0 of the programming software 907 AC 1131 on, the **write protection password** is also transferred to the controller when the program is loaded. The password is then inquired during the login to the controller.

⚠️   **Caution:**
Password protection of the PLC is possible on controllers with a **runtime system of version V5.0** or later. For controllers with a runtime system version V4.x, the password protection is ineffective.

4.1.1   Setting the PLC password

The following steps have to be carried out to set a password for the PLC:

1. Start the 907 AC 1131 software and open the project.

2. Enter the desired password (e.g. ac31) under "Project/Options/Passwords" -> "Write Protection Password", confirm the password by entering the password in the "Confirm Write Protection Password" field and close the dialog with <OK>.

3. Save and close the project.

4. Open the project again. When opening the project, you are asked to enter the write protection password.

    Enter Write Protection Password

    Write Protection Password: [   ]   OK   Cancel

    If you enter the password and confirm it with <OK>, you can modify the project and save the changes.
    If you press <Cancel>, you cannot modify the project.

5. Login to the controller and load the program into the controller and then into the Flash memory. The write protection password is automatically transferred to the controller and written to the Flash memory together with the program.

6. Start the program and logout again.

When you open the project again, the password is asked each time you login to the controller (refer to section 4.1.2 "Login to a password protected PLC").

4.1.2 Login to a password protected PLC

When logging in to a password protected PLC it is distinguished whether a write protection password has been entered when opening the project or not.

Login with projects that have the same write protection password:

The following description applies to the following cases:

- Same project, same write protection password, password was entered when opening the project
- Other project, same write protection password, password was entered when opening the project
- Same project, same write protection password, password inquiry when opening the project was closed with <Cancel>
- Other project, same write protection password, password inquiry when opening the project was closed with <Cancel>

If the write protection password is entered when opening the project, the login to the controller can be performed without any further inquiry.

If the inquiry for the write protection password is closed with <Cancel> when opening the project, the controller password is asked when logging into the controller.

If you enter the correct password and confirm the dialog with <OK>, the login to the PLC is performed.

If you enter an incorrect password or close the dialog with <Cancel>, the following message is displayed:

After you confirm this message with <OK>, another message appears displaying which project is loaded on the controller. Then the login is aborted.
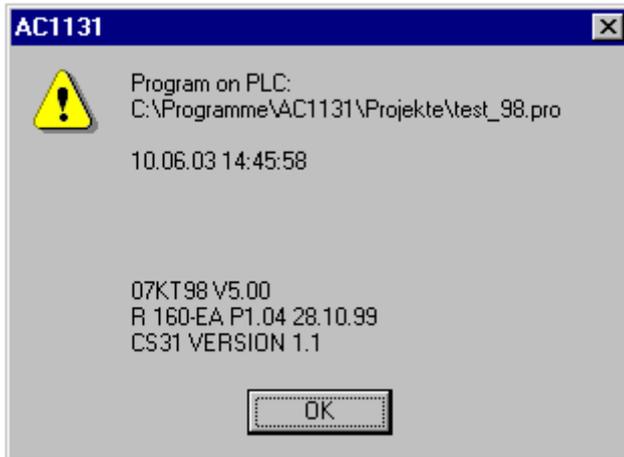
Login with projects that have a different write protection password:

The following description applies to the following cases:

- Same project, different write protection password, password was entered when opening the project
- Other project, different write protection password, password was entered when opening the project
- Same project, different write protection password, password inquiry when opening the project was closed with <Cancel>
- Other project, different write protection password, password inquiry when opening the project was closed with <Cancel>

If you login with a project that has a different write protection password, the PLC password is inquired, no matter whether the write protection password has been entered when opening the project or not.

To perform a successful login, the password stored in the PLC has to be entered.

If it is another project, the message "Other project ..." appears. When the new project is loaded and saved in the Flash memory, the new password is also loaded!
(Refer to section 4.1.3 "Changing the controller password".)

Login with projects that do not have a write protection password:

The following description applies to the following cases:

- Same project without a write protection password.
- Other project without a write protection password.

If you login with a project that does not have a write protection password, the PLC password is inquired.

To perform a successful login, the password stored in the PLC has to be entered.

If it is another project, the message "Other project ..." appears. When the new project is loaded and saved in the Flash memory, the password is deleted! (Refer to "Deleting a password".)

4.1.3 Changing the controller password

The following steps have to be performed to change the controller password:

1. Start the 907 AC 1131 software and open the project.

2. Enter the write protection password and confirm with <OK>.

3. Change the write protection password under "Projects/Options/Passwords". Enter the desired new password (e.g. user) under "Write Protection Password", confirm the password by entering the password in the "Confirm Write Protection Password" field and close the dialog with <OK>.

4. If you have modified the project you can transfer the changed password after the login (refer to step 6, 7) using online change. Otherwise you have to force a download with "Project/Clean All" and "Project/Rebuild All" or <F11>.

5. Save the project by selecting "File/Close".

6. Login to the controller with "Online/Login". The inquiry for the controller password is displayed.

Enter the controller password (i.e. the previous write protection password) and confirm with <OK>.

7. Perform an online change or a download and save the project (incl. the new password) in the Flash memory using "Online/Create boot project (flash)".

Note:
If you load another project with a different write protection password to the controller, the current PLC password is asked during the login and, after this password is entered, the new project with a new password is loaded to the controller. Here it makes no difference whether you have entered the write protection password when opening the project or closed the dialog with <Cancel>.
If you do not know the write protection password of the project, you can't access the PLC afterwards!

4.1.4 Deleting the controller password

Deleting the controller password on the PLC is done by loading a project without a write protection password. For this, the following steps are required:

1. Start the 907 AC 1131 software and open the project.

2. Enter the write protection password and confirm with <OK>.

3. Under "Projects/Options/Passwords" delete the write protection password in the fields "Write Protection Password" and "Confirm Write Protection Password" and close the dialog with <OK>.

4. If you have modified the project you can perform an online change after the login (refer to step 6, 7). Otherwise you have to force a download with "Project/Clean All" and "Project/Rebuild All" or <F11>.

5. Save the project by selecting "File/Close".

6. Login to the controller with "Online/Login". The inquiry for the controller password is displayed.

Enter the controller password (i.e. the previous write protection password) and confirm with <OK>.

7. Perform an online change or a download and save the project (without password) in the Flash memory using "Online/Create boot project (flash)".

Note:
If you load another project without a write protection password to the controller, the current PLC password is asked during the login and, after this password is entered, the new project without password is loaded to the controller.

4.2 Remanent variables

4.2.1 Overview

Remanent variables can keep their values beyond the usual program running time. This includes retain variables and persistent variables which are stored in the SRAM and buffered by a **battery**.

Section 3.3 - "Initialization and buffering of operands" in volume 15 - "System technology S90 / System technology of the basic units" describes the initialization of the variables. There, the peculiarities when declaring the variables and the differences between the variables of the addressable area (%M area) are also mentioned.
Section 4.2.2 "RETAIN variables - note for program change" of this documentation describes how RETAIN variables can keep their values during a download.

From 907 AC 1131 version V5.0 on, persistent variables are available in addition to the retain variables.

> ⚠️ **Caution:**
> Persistent data are possible on controllers with a **runtime system of version V5.0** or later. For controllers with a runtime system version V4.x, VAR PERSISTENT is ineffective.

Retain variables are identified by the keyword **RETAIN**. These variables keep their value after switching the controller normally off and on (or after the "Online / Reset" command is applied) as well as after an uncontrolled stop of the controller. When the program is started again the stored values are processed again. An example for an application of these variables is a parts counter used in a manufacturing installation which has to continue counting even after a voltage breakdown. After a new program download, retain variables are not initialized new, but change their order if the number or order in the program has changed.

Persistent variables are identified by the keyword **PERSISTENT**. In contrast to the retain variables, the persistent variables keep their value after a new download, but not after a stop and subsequent restart of the controller (i.e. not after a "Online / Reset" command) because they are not stored in the "retain area". If persistent variables shall keep their values even after an uncontrolled controller breakdown they have to be additionally declared as VAR RETAINs (VAR RETAIN PERSISTENT). An example for an application of "persistent retain variables" is an hour meter which has to continue counting even after a voltage breakdown.

> ⚠️ **Caution:**
> - If a local variable is declared as RETAIN in a program, exactly this variable is stored in the retain area (like a global retain variable).
>
> - If a local variable is declared as RETAIN in a function block, the complete instance of this function block is stored in the retain area (all data of the POU) but only the declared retain variable is treated as such.
>
> - If a local variable is declared as RETAIN in a function, this is without any effect. The variable is not stored in the retain area! If a local variable is declared as PERSISTENT in a function, this is also without effect.

During a download a description file of the persistent data is loaded to the controller and stored in the SRAM. According to the description file the persistent data are copied. For the 07 KT 97 16 kB are available for the description file and for the 07 KT 98 256 kB are available.

The PLC browser command "tsk" outputs the start address of the description file. For each variable the name and the pointer are stored. The PLC browser command "delpersistent" deletes the description file (refer to 4.4 "PLC browser - new functions").

4.2.2 RETAIN variables - note for program change

The starting behavior and the buffering of variables are described in section "Initialization and buffering of data areas" of volume 15 - System technology.

RETAIN variables only keep their order when the program change is performed using online change. When performing a download or when changing the variables (add, delete, change order) the order of the RETAIN variables is also lost resulting in a loss of their values.

Below a method is described how to keep the RETAIN variables even when changing the program. A small example program is used to simplify the description.

The following small project is created (here in ST):

For our project we create one (or more) watch list(s) containing the RETAIN variables. Note that the initialization value is entered after the variable name (without ;).

☞ **Note:**

From version V4.3 of the programming software 907 AC 1131 on, the creation of the watch lists can be done automatically. For this, the parameter **RetainWatchLists=Yes** has to be entered in the file **AC1131.INI**. Then the watch lists are created using the command "Online"=>"Remanent Data"=>"Create Watch Lists for Retain Variables".

In our example the watch list was saved under the name VAR_RETAIN_1.WTC.
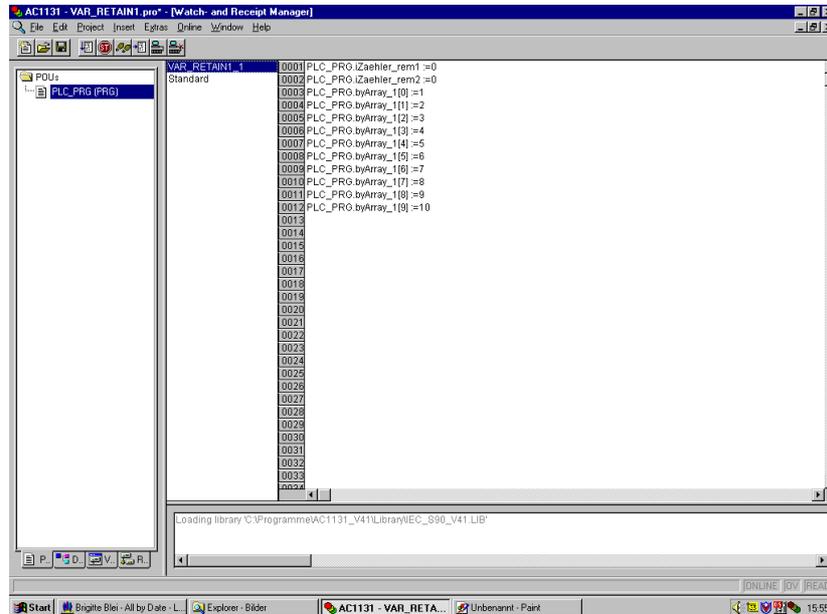
Now the program is saved, provided with write protection, downloaded to the controller and flashed.
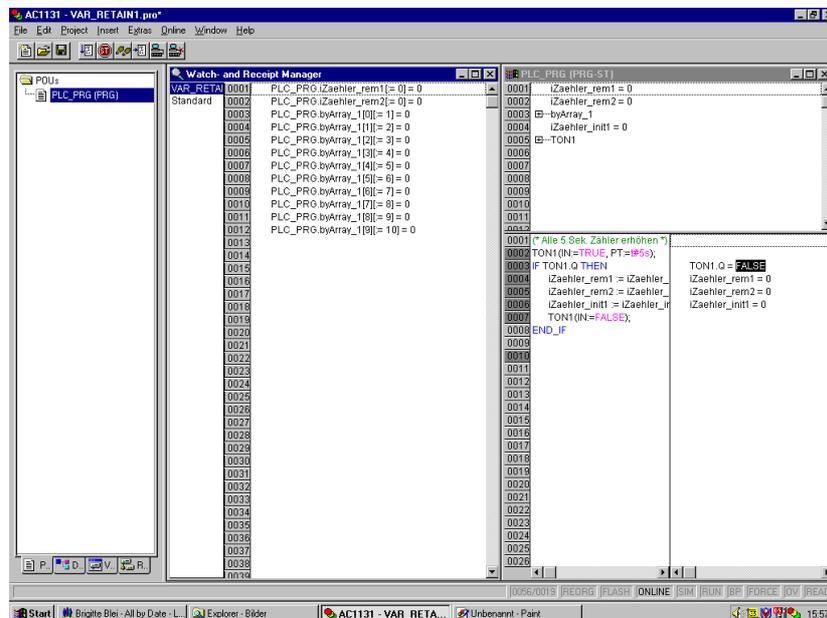
After starting the program the variables change as desired:
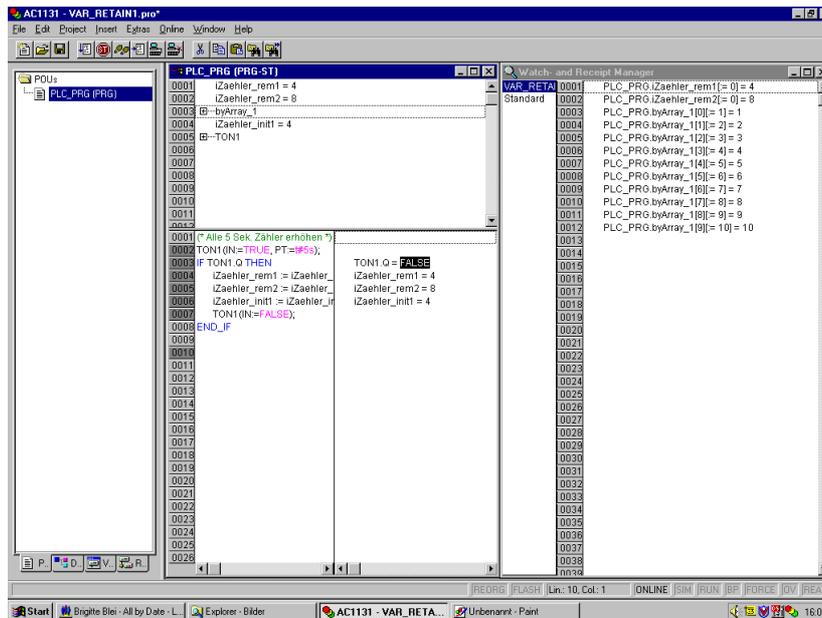
The next step is to add one more counter. Prior to modifying the program we must read the current state of the RETAIN variables. For this, we login to the controller with the project still unchanged. Then we set the controller to STOP mode ("Online/Stop") and in the watch and recipe manager we open a context menu by right clicking with the mouse. Select "Read Recipe". The variables are updated once again.
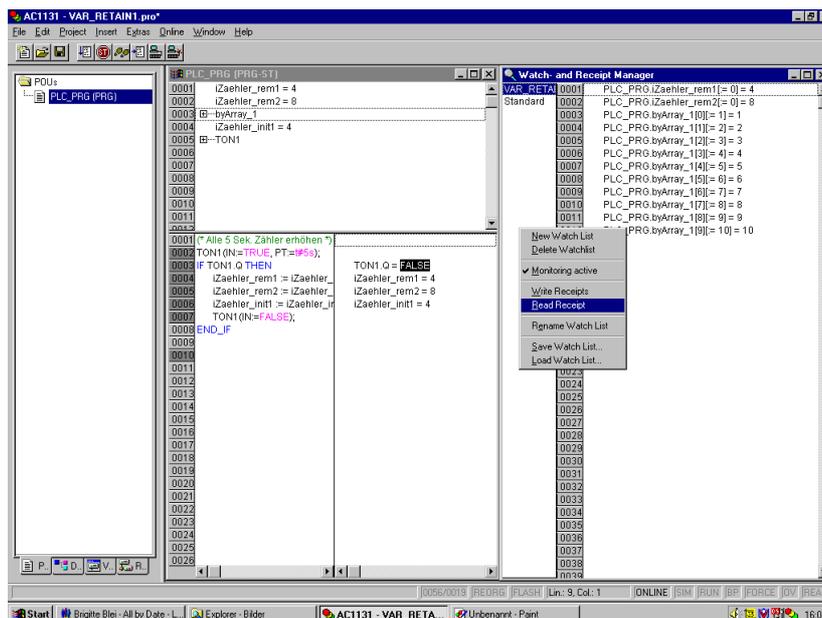
After this we open the context menu again and select "Save Watch List". In our example the name VAR_RETAIN_2.WTC was chosen. Now we logout and delete the watch list (same context menu, "Delete Watch List").

After that we modify the project (or close it and open it again entering the write protection password for modifying). Now we add one more counter to the VAR_RETAIN list which is also incremented by 3 every 5 seconds in the program. After the modifications are completed, translate the project using the <F11> key, save it and download it to the controller.
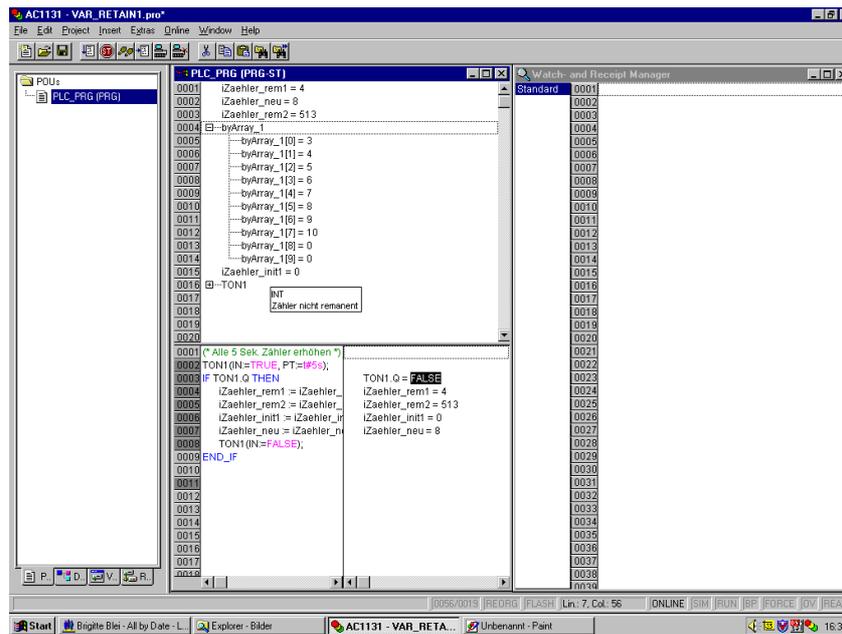
You can see that the values are now incorrect!

This is the reason why we have to log out again and load the watch list containing the stored RETAIN variables (in our example VAR_RETAIN_2.WTC). The name inquiry was answered with VAR_RETAIN_3. Now the watch list is supplemented by the new variable iZaehler_neu and saved again. After logging in again the watch list contains the values currently read from the controller and the correct values saved before.
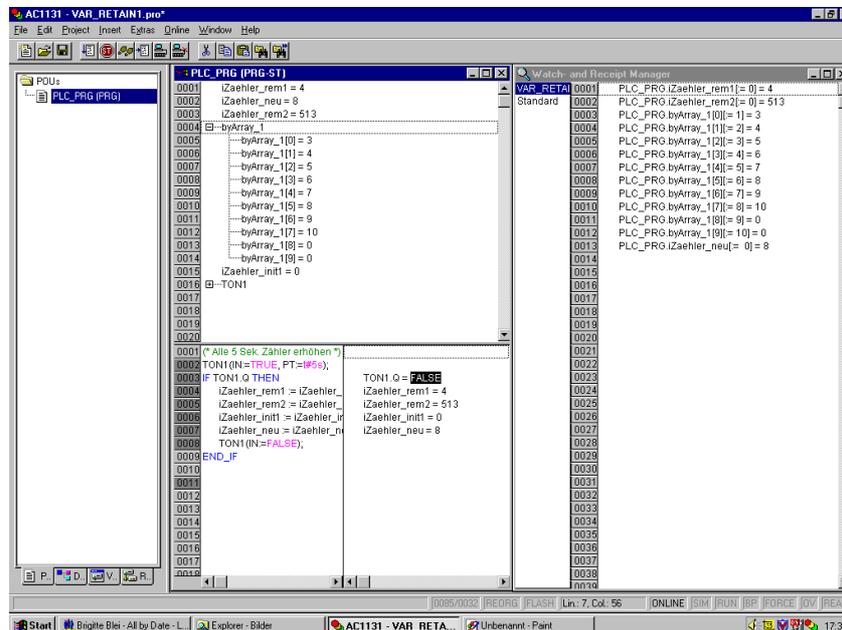
To write the correct values back to the controller, you must open the context menu again using the right mouse button (or menu "Extras") and then select the command "Write Recipe". Answer the question "Do you really want to write the current recipe" with "Yes". The "old" values are written and the new counter is initialized by zero.
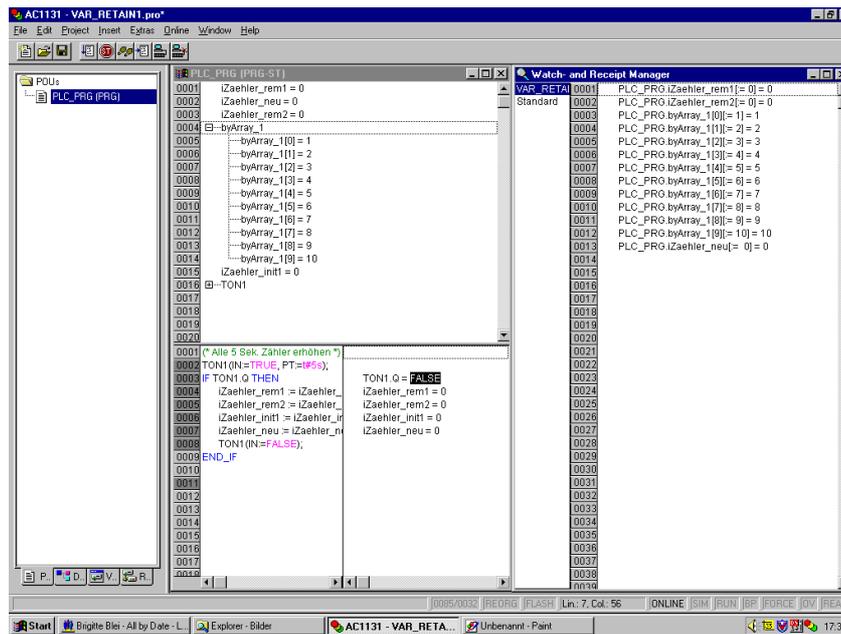
If the program is now started the values go on correctly. The new counter begins to count correctly. After this log out again.

4.3   Source download to a SmartMedia card

When loading a project to the controller, the translated code (i.e. code which can be directly processed by the processor) is loaded and stored in the Flash memory using the command "Online / Create boot project (flash)". Using the command "Online / Write user program to SMC" the translated code can be stored to a SmartMedia card (refer to chapter 2.2.6, Menu "Write user program to SMC").

Because the original project cannot be built from the translated code, this code can also not be read out from the controller.

From 907 AC 1131 version V5.0 on it is possible to write the source program to a SmartMedia card which is specially initialized for this purpose.

The following SmartMedia cards are available for a source download:
- 07 MC 90 R0101 with 2 MB and
- 07 MC 90 R0201 with 8 MB for controllers 07 KT 9x R02xx and runtime system V5.0 onwards.

Caution:
Source download to SmartMedia card is possible on controllers with a **runtime system of version V5.0** or later. For controllers with a runtime system version V4.x, the source download is ineffective.

Proceed as follows to write the source program to a SmartMedia card:

1. Initialize the SmartMedia card for the source download.
2. Set the options for the source download in 907 AC 1131.
3. Load the project to SmartMedia card.
4. If necessary, read out the project from SmartMedia card.

4.3.1   Initializing the SmartMedia card for the source download

Initialization of the SmartMedia card is performed using the PLC browser (refer to chapter 4.4 "PLC browser - new functions"). Proceed as follows:

1. Boot the PLC without a SmartMedia card inserted.

2. Login to the controller (with any project).

3. If necessary, stop the user program
   by toggling the RUN/STOP switch to the STOP position
   or using the command "Online / Stop".

4. Insert the SmartMedia card into the controller
   (chip points to the top, refer to hardware documentation).

5. Launch the PLC browser under "Resources / PLC Browser".

6. Enter the command **initsmc**.
   SmartMedia card is going to be initialized. This can take several minutes.

   Caution:
   Data possibly stored on the card are deleted without any inquiry.

   The **Batt.** LED on the PLC flashes during the initialization process.

4.3.2 Setting the options for the source download in 907 AC 1131

Setting the source download options is done in the **"Source download"** category in the menu **"Project / Options"**. After selecting this category the following dialog appears:
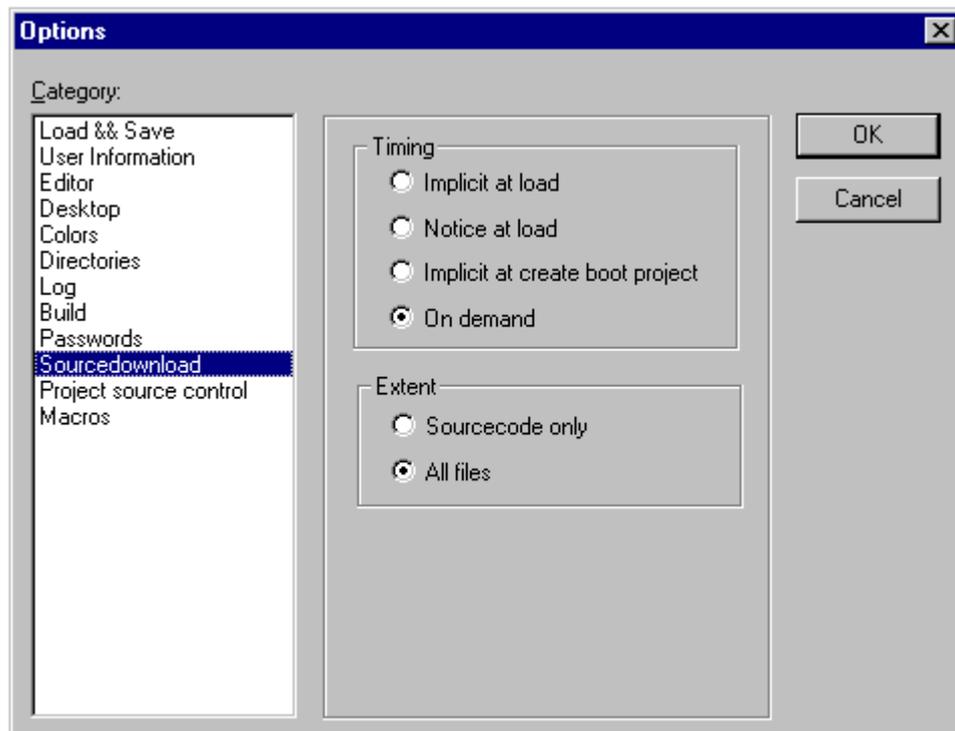
Figure 4-1: Dialog for setting the options for the source download

You can choose the **Extent** and **Timing** for storing the project source code to the SmartMedia card. For this, the data are packed.

The option **Sourcecode only** is only relevant for the 907 AC 1131 project file (extension **.pro**).

The option **All files** additionally includes files such as the assigned libraries, visualization bitmaps, configuration files, etc.

If the option **Implicit at load** is selected for the download, the selected scope of data is automatically loaded to the controller after loading the translated code is finished.

If the option **Notice at load** is selected, the inquiry "Write source code to controller?" appears during the download. Confirm with <YES> if you want the selected scope of data to be automatically loaded to the controller. Otherwise close the dialog with <NO>.

The option **Implicit at create boot project** is without effect.

If the option **On demand** is selected, the command "Online / Write project source code to SMC" has to be used explicitly to load the selected scope of data to the controller.

4.3.3 Loading the project to SmartMedia card

When loading the source program to the SmartMedia card, the following possibilities exist:

Case 1: Loading the project completely new,
i.e. downloading the translated code, storing it in the Flash memory and loading the source code to the SmartMedia card.

Case 2: Loading the project changes,
i.e. online change, storing in the Flash memory and loading to the SmartMedia card.

Case 3: Loading only the project source code to SmartMedia card.

Caution:

Packing the source program depends on the computing capacity of the PC. Depending on the program size, loading and storing the source program to the SMC card can take several minutes. In PLC run mode, particularly for projects with a cyclic task the loading process can last very long. This is why the source download should generally be done in PLC stop mode.

Case 1: Loading the project completely new:

Proceed as follows to load the complete project to the controller and the source code to the SmartMedia card:

1. Boot the PLC without a SmartMedia card inserted.

2. Insert the SmartMedia card initialized for the source download into the controller
   (refer to chapter 4.3.1 "Initializing the SmartMedia card for the source download")
   (chip points to the top, refer to hardware documentation).

3. Start the 907 AC 1131 software and open, translate and save the desired project. If you didn't perform any modifications, you can force a download with "Project/Clean All" and "Project/Rebuild All".

4. Login to the controller with the project to be loaded and perform a download. If you have selected one of the timing options **"Implicit at load"** or **"Notice at load"** for loading the source program (refer to chapter 4.3.2 "Setting the options for the source download in 907 AC 1131"), loading the source program is automatically performed after loading the translated code either without or with an inquiry.

5. Save the project in the Flash memory using "Online / Create boot project (flash)".

6. If you have selected the timing option **"On demand"** for loading the source program, you have to start loading the source program by selecting the menu item "Online / Write project source code to SMC".

7. Start the user program.

Case 2: Loading the project changes:

Proceed as follows if you only want to load the project changes (i.e. perform an online change), store them in the Flash memory and load the source code to the SmartMedia card.

1. Insert the SmartMedia card initialized for the source download into the (running) controller (refer to chapter 4.3.1 "Initializing the SmartMedia card for the source download") (chip points to the top, refer to hardware documentation).

2. Start the 907 AC 1131 software and open, modify and save the desired project.

3. Login to the controller with the project to be loaded and perform an Online Change. If you have selected one of the timing options **"Implicit at load"** or **"Notice at load"** for loading the source program (refer to chapter 4.3.2 "Setting the options for the source download in 907 AC 1131"), loading the source program is automatically performed after loading the translated code either without or with an inquiry.

4. Save the project in the Flash memory using "Online / Create boot project (flash)".

    ⚠️ **Caution:**

    In PLC run mode, particularly for projects with a cyclic task the flashing process can last very long. Flashing is performed considerably faster if the PLC is in stop mode.

5. If you have selected the timing option **"On demand"** for loading the source program, you have to start loading the source program by selecting the menu item "Online / Write project source code to SMC".

    ⚠️ **Caution:**

    In PLC run mode, particularly for projects with a cyclic task the loading process can last very long. Loading the source program is performed considerably faster if the PLC is in stop mode.

Case 3: Loading only the project source code to SmartMedia card:

Proceed as follows to load only the source code of the project to the SmartMedia card:

1. Insert the SmartMedia card initialized for the source download into the (running) controller (refer to chapter 4.3.1 "Initializing the SmartMedia card for the source download") (chip points to the top, refer to hardware documentation).

2. Start the 907 AC 1131 software and open the desired project.

3. Select the timing option **"On demand"** for loading the source program.

4. Login to the controller.

5. If necessary, stop the user program
by toggling the RUN/STOP switch to the STOP position
or using the command "Online / Stop".

⚠️ **Caution:**

In PLC run mode, particularly for projects with a cyclic task the loading process can last very long. Loading the source program is performed considerably faster if the PLC is in stop mode.

6. Start loading the source program by calling the menu item "Online / Write project source code to SMC".

4.3.4 Reading out the project from SmartMedia card

Proceed as follows to read the source program from the SmartMedia card into the PLC:

1. If necessary, insert the SmartMedia card containing the source code into the running controller.

2. Start the 907 AC 1131 software.

3. Under "Project / Options" set the directory for the project to be read out in the "Directories" category under "General / Upload files".

⚠️ **Caution:**

The specified directory must exist. Otherwise an error message appears when reading out the data.

4. Select **"File / Open"** with the option **"Open project from PLC"**.

5. First the "Communication Parameters" dialog appears. Select the suitable gateway channel for your controller.

6. The login to the controller is performed and the data are read from the SmartMedia card and stored in the directory specified in step 3. During this process the data will be unpacked again.

4.4 PLC browser - new functions

The PLC browser is described in volume 10 - Operating instructions S.90/40/50 in section "Resources".

From 907 AC 1131 version V5.0 on, the following new PLC browser commands exist:

- delflash - Deletion of the user program in the Flash memory.

- initsmc - Initialization of a SmartMedia card for the source download.

- delpersistent - Deletion of the description file for the persistent data.

⚠️ **Caution:**
The new PLC browser commands are possible on controllers with a **runtime system of version V5.0** or later. For controllers with a runtime system version V4.x, the commands are ineffective.

From 907 AC 1131 version V5.0 on, the file **BrowserHelp.INI** has to be located in the directory of the PLCCONF configuration files, e.g.
C:\Programs\AC1131\Library\PLCCONF\

4.4.1 Deleting the user program in the Flash memory

Using the PLC browser command **delflash** the user program is deleted from the Flash memory without an inquiry. (This command corresponds to the 'deep' command of the EBS operating system.)

When logging in after a reboot of the PLC, the controller outputs the message "No program available in the controller ...".

4.4.2 Initializing a SmartMedia card for the source download

The PLC browser command **initsmc** initializes an inserted SMC card for the source download without an inquiry (refer to chapter 4.3.1 "Initializing the SmartMedia card for the source download").

⚠️ **Caution:**
Data possibly stored on the SmartMedia card (firmware, user program or user data) are deleted without an inquiry!

4.4.3 Deleting the description file for the persistent data

Using the PLC browser command **delpersistent** the description file for the persistent data is deleted without an inquiry.

5   New visualization elements of 907 AC 1131

5.1   Overview of new visualization elements

The visualization in 907 AC 1131 has been expanded by the following visualization elements:

- Circle segment (refer to chapter 5.2)
- Table (refer to chapter 5.3)
- Meter (refer to chapter 5.4)
- Bar display (refer to chapter 5.5)
- Histogram (refer to chapter 5.6)
- Trend element (refer to chapter 5.7)
- Alarm table (refer to chapter 6.2 "Visualization of alarms using the alarm table")

Overview of categories for the configuration of visualization elements

Depending on the selected visualization element different categories are available:

Category:	Available for:
Shape	Rectangle, rounded rectangle, ellipse
	All
Line width	Rectangle, rounded rectangle, ellipse, polygon, line, curve, bitmap, visualization, circle segment
Color	Rectangle, rounded rectangle, ellipse, polygon, line, curve. Only frame color: Bitmap, visualization, circle segment.
Motion absolute	All, except for table *
Motion relative	Rectangle, rounded rectangle, ellipse, bitmap, WMF file, visualization
Variable	All, except for table *
Input	All, except for table *
Angle	Circle segment
Text for Tooltip	All, except for table *
Bitmap	Bitmap, button
	All, except for table *
	Please note: If visualization element properties which are defined via "normal" project variables are additionally programmed via a structure variable, the values of the normal variable are first evaluated in online mode!

Note: (*)
Meter, bar display and histogram have to be grouped newly beforehand.

5.2 Circle segment

Inserting a circle segment:

Menu item "Insert / Circle Segment" or icon:

Using this command you can insert a circle segment as an element into your current visualization.

With the left mouse button pressed, drag an area of the desired size. An oval-shaped element with a radius line is displayed the shape and size of which changes when moving the mouse. The small black square outside of the element indicates the corner of a virtual rectangle surrounding the element.

In order to determine the start and end angle of the circle segment, click on the end of the radius line on the circular arc after inserting the element. If you move the mouse pointer while pressing the mouse button, two small black squares appear showing the two angular positions. From now on, these can also be selected and moved separately. If you want to define the angles dynamically, open the configuration dialog of the 'Angle' category to enter the corresponding variables (refer to chapter 5, "Angle").

To modify the shape and size of the element again, either click on the center point so that the cursor is displayed as two diagonally crossed arrows, and then move the mouse pointer using the mouse or the arrow buttons of the keyboard while keeping the right mouse button pressed. Or you select and move the square indicating the corner position outside the element. In order to move the element, click on the free area inside the element so that the cursor changes to vertically crossed arrows.

Configuring a circle segment

The "Configure circle segment" dialog contains the following configuration categories which are also known for the other elements:

- Text
- Text variables
- Line width
- Colors
- Color variables
- Motion absolute
- Variables
- Input
- Text for Tooltip
- Security
- Programmability

Furthermore it contains the "Angle" category which is only available for the circle segment.

Angle:

In the **Angle** category of the 'Configure circle segment' dialog you can enter one value or project variable defining the **start angle** and the **end angle** of the circle segment element in degrees of angle.
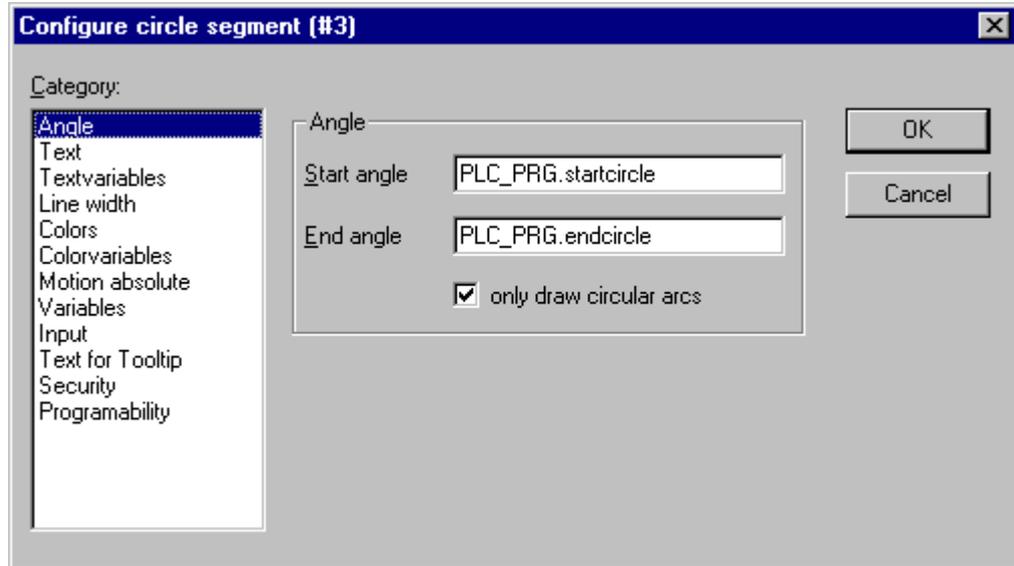
Figure 5-1: Dialog for configuring visualization elements

Example:
Input for start angle: "90", end angle "180"
If the option **only draw circular arcs** is activated, only the line of the circular arc is displayed. Otherwise the circle segment is filled.

Figure 5-2: Online representation of a circle segment

5.3   Table

Inserting a table:

Menu item "Insert / Table" or icon: 🖿

Using this command you can insert a table used for the representation of the elements of an array into your current visualization.

With the left mouse button pressed, drag the element to the desired size. Before the element is represented with its final appearance, the configuration dialog 'Configure Table' is displayed.

Configuring a table:

The 'Configure table' dialog is displayed as soon as a table for the representation of an array is inserted into the visualization. Apart from the categories for the Tooltip and Security configuration which are also known for other elements, the categories 'Table', 'Columns', 'Rows' and 'Selection' for defining the appearance and content of the table are available.

Category 'Table':
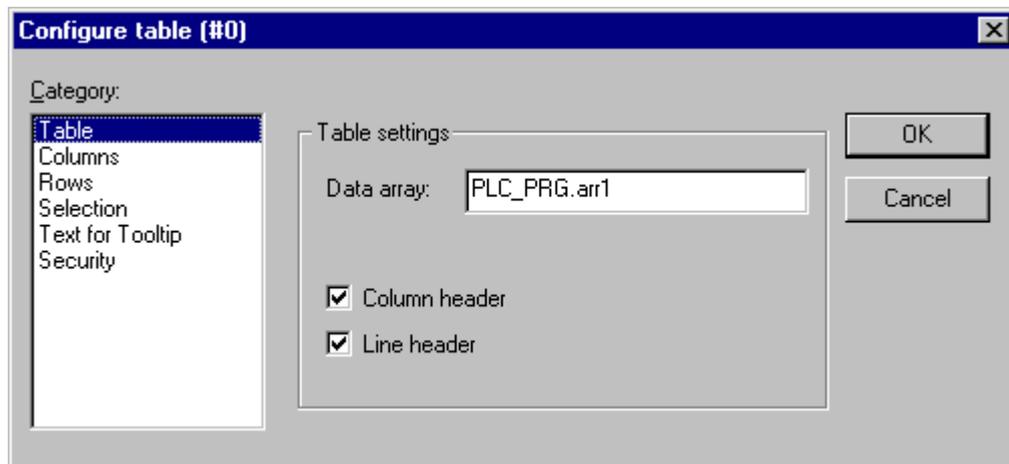
Figure 5-3: Dialog for configuring a table, category 'Table'

In this dialog, enter the following table settings:

Data array:
Enter an array of your project the fields of which shall be represented in the table. Use the input aid (<F2>) or the Intellisense function.

Column header, Line header:
Activate these options if you want the column and/or row headers to be displayed. The row header (on the left-hand border of the table) corresponds to the array index. The column header can be defined in the 'Columns' category.

Category 'Columns':
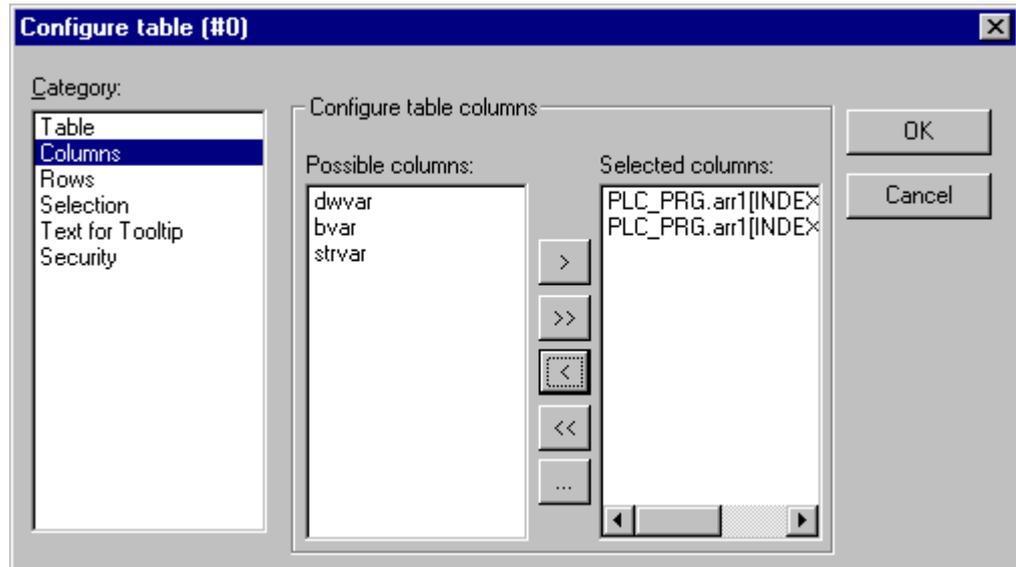
Figure 5-4: Dialog for configuring a table, category 'Table'

Configuring table columns:
Here you define which array elements shall be represented in the table columns. You can also define the column headers and one set of configuration settings per column. The left-hand area of the window (**Possible columns**) shows the elements available for each index of the array. For example, all structure elements for the array of a structure.

Using the arrow buttons (**>**, **<**) between the window areas you can select individual elements and move them into the right-hand area of the window (**Selected columns**). Using the button **>>** you can move all elements at once. To modify the pre-defined representation of an element in the table column, double click on the entry or click on the '**...**' button. The **Configure columns** dialog appears.
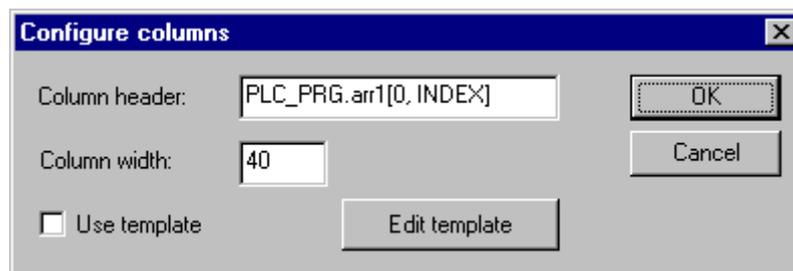
Figure 5-5: Dialog for configuring a table, category 'Columns', column properties

- To modify the column header and column width:
  The text field **Column header** first displays the automatically assigned title (for the array of a structure e.g. "PLC_PRG.arr1[INDEX].iNo" for the column which will represent the structure element "iNo"). This title can be edited. You can furthermore define the **Column width** (number of characters).

- To change additional configuration parameters for all elements inside of one column:
By default, the table fields are represented as simple rectangles. The entries cannot be edited. However, by activating the **Use template** checkbox you can use a pre-defined set of settings, e.g. a particular line width definition, a possibility for entering text, etc. For this, the known configuration dialogs for rectangle, bitmap and button are available which can be opened by clicking on the **Edit template** button.

In order to define online actions for all entries of a column (i.e. for all array elements) use the **place holder [INDEX]** after the array definition when entering into the configuration dialogs (e.g. as it is also used in the default column header).

Example:
You want to visualize an array "arr1 [0..2] of BOOL" (table with 1 column). When a table cell is clicked in online mode you would like that the cell is displayed red and that the corresponding array element can be toggled. For this purpose, you use 'Edit template' when configuring the column and define the template as follows: Category 'Input', action 'toggle variable': "PLC_PRG.arr1[INDEX]". Category 'Colors': Alarm color red. Category 'Variables', action 'change color': "PLC_PRG.arr1[INDEX]".

Category 'Rows':
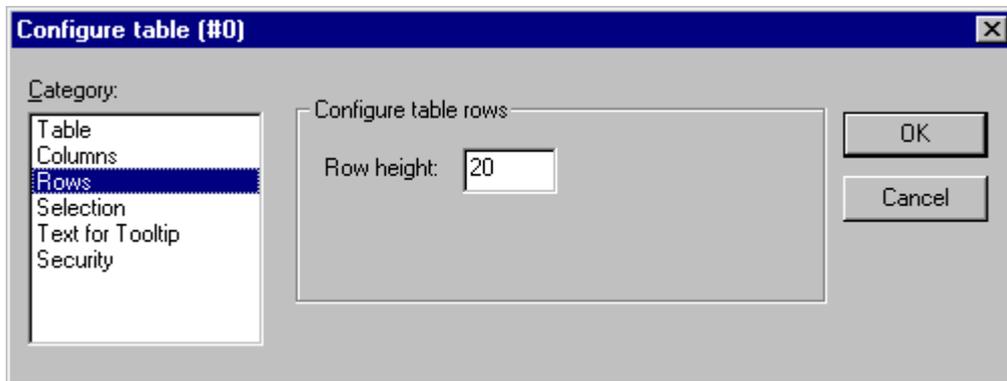
Figure 5-6: Dialog for configuring a table, category 'Rows'

Row height:
Enter the desired height of the table rows in pixels.

Category 'Selection':

In this dialog the following settings regarding the method of selection in the table can be made:
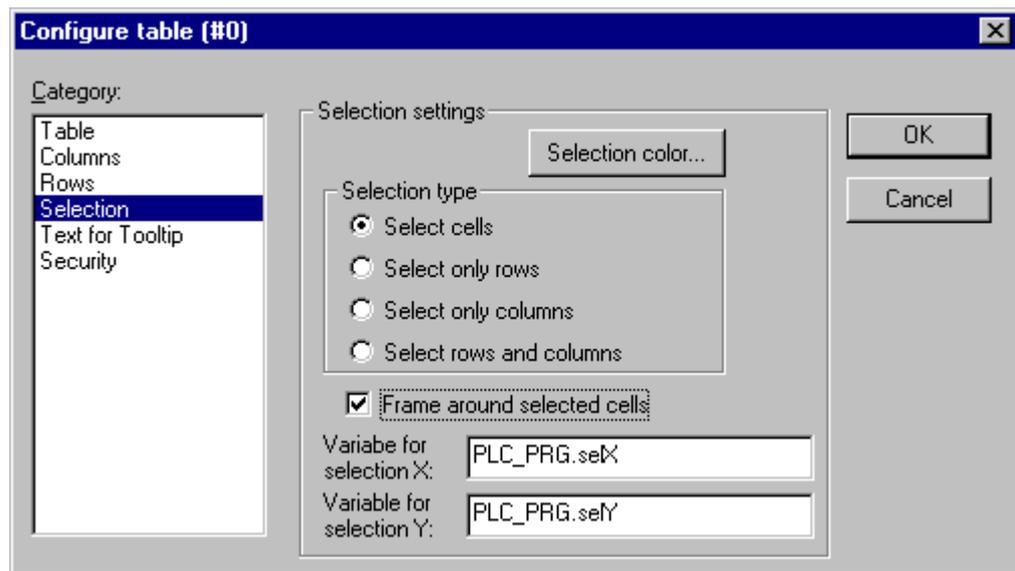
Figure 5-7: Dialog for configuring a table, category 'Selection'

Selection color:
This button opens the standard color selection dialog where you can define a color to highlight (fill) selected table cells.

Frame around selected cells:
A selected cell is surrounded by a frame.

Variable for selection X, Variable for selection Y:
Here you can enter a project variable to display the X or Y index of the selected table cell.

Example:
Table for displaying the array of a structure:

Create the following structure in your project:

TYPE strucTab :
STRUCT
    iNo: INT;
    bDigi : BOOL;
    sText:STRING;
    byDummy: BYTE;
END_STRUCT
END_TYPE

Define the following array and variable in PLC_PRG:

arr1:ARRAY [1..5] OF strucTab;
selX:INT;
selY:INT;

Insert a table into the visualization and configure the table as follows:

Category 'Table':
Data array: "PLC_PRG.arr1"

Category 'Columns':
(Answer the displayed question dialog with YES)
- Move the elements iNo, bDigi and sText from the 'Possible columns' area to the right-hand 'Selected columns' area.
- There, double click on the first entry (PLC_PRG.arr1[INDEX].iNo) and then change the column header to "Number" in the appearing dialog. Confirm with 'OK' and also define new column headers for the other two entries (e.g. "Value" and "Text").
- In the category 'Selection', enter "PLC_PRG.selX" for the 'Variable selection X' and "PLC_PRG.selY" for the 'Variable selection Y'.
- Activate the option 'Frame around selected cells'. Click on the 'Selection color' button and select a color. Close the configuration dialog with 'OK'.
- The table element should now look as follows.
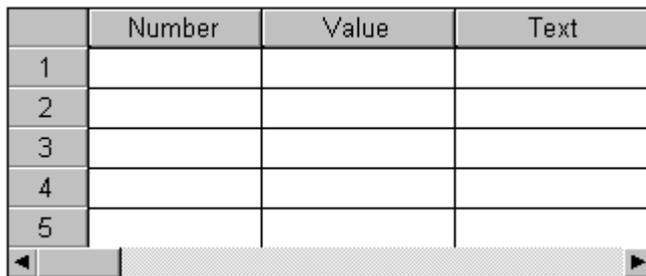
	Number	Value	Text
1			
2			
3			
4			
5			

Figure 5-8: Example of a table in offline mode

Array index numbering on the left-hand border; the column titles of the three structure elements selected for display on the top. You can change the column width by positioning the mouse cursor on the separation lines and then, after the mouse cursor has changed to a horizontal arrow with two heads, dragging the separation line to the left or to the right.

In online mode the table cells display the current values of the array elements. When you click on a table cell, it is surrounded by a frame and displayed in the selected color. Example:
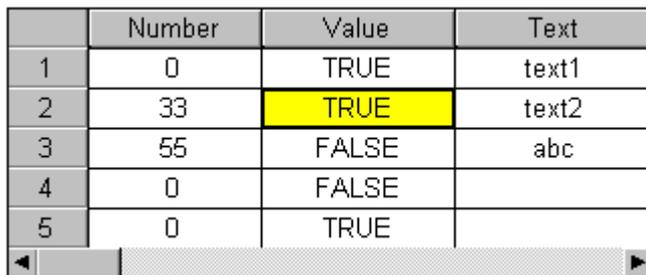
	Number	Value	Text
1	0	TRUE	text1
2	33	TRUE	text2
3	55	FALSE	abc
4	0	FALSE	
5	0	TRUE	

Figure 5-9: Example of a table in online mode

5.4 Meter

Inserting a meter:

Menu item "Insert / Meter" or icon:

Using this command you can insert a meter into your current visualization. The meter provides a scale on a definable segment of a circular arc and a pointer which is "pivoted" in the virtual center of the circle.

With the left mouse button pressed, drag the element to the desired size. Before the element is represented with its final appearance, the configuration dialog 'Configure meter' is displayed. Here you can define various parameters and view their effects to the representation in a preview before finally inserting the element by clicking on 'OK'.

Configuring a meter:

The 'Configure meter' dialog automatically appears after a meter element is inserted into the visualization. A **preview** displays the appearance of the element depending on the parameter settings.
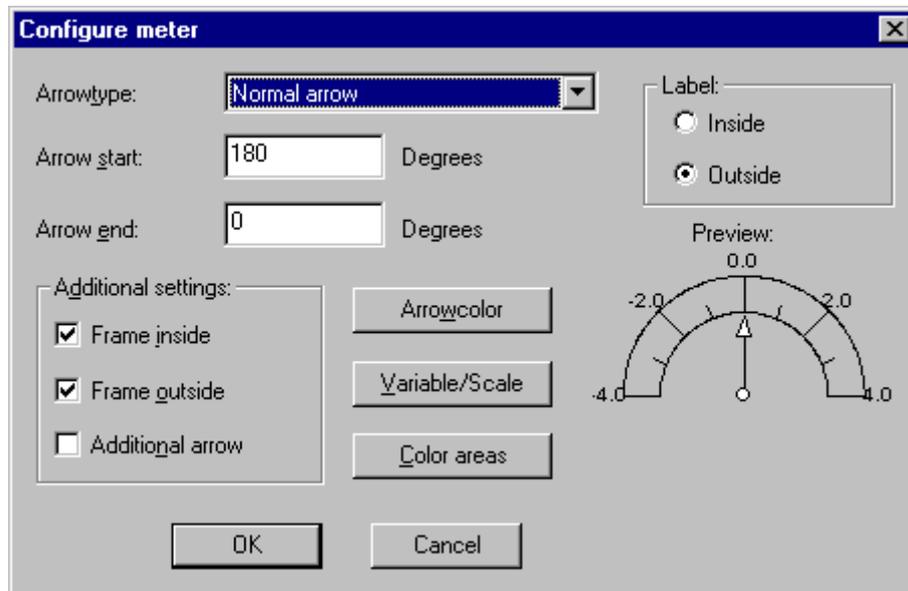
Figure 5-10: Dialog for configuring a meter

Arrow type:
Select whether the current value shall be indicated on the scale by a "normal arrow", a "thin arrow", a "thick arrow" or a "thin needle".

Arrow start, Arrow end:
In these fields, enter the positions (in degrees of angle) for the beginning and the end of the scale on a circular arc (for example, a start value of 180° and an end value of 0° display the scale as a horizontal semicircle bend to the top).

Arrow color:
This button opens a standard dialog where you can select a pointer color.

Variable/Scale:
This button opens the **Configure scale and variable** dialog:
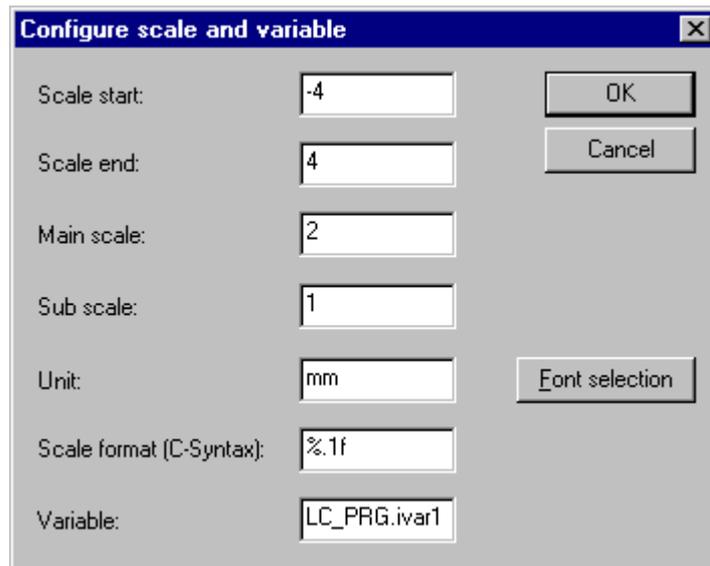
Figure 5-11: Dialog for configuring the scale and the assigned variable of a meter

Scale start, Scale end:
Lowest and highest value of the scale, e.g. "-4" and "4".

Main scale:
Defines the scale intervals to be displayed between the minimum and the maximum scale value (i.e. labeling + scale mark). For example, a value of "2" will display every second integer scale value on the scale.

Sub scale:
Here you can define a sub scale to be displayed in addition to the main scale (short scale marks without labeling displayed between the labeled long scale marks of the main scale).

Unit:
Here you can enter the unit of the scale, e.g. "cm" or "sec". The unit is displayed near the pivot point of the pointer.

Scale format (C syntax):
Here you can define the display format of the scale labeling according to the C syntax. For further information please refer to the category 'Text' in this chapter. For example a definition of %1.1f will display the scale values with one decimal place (e.g. "12.0").

Variable:
The variable in this field defines the position of the pointer (e.g. "PLC_PRG.posvar").

Font selection:
This button opens a standard dialog where you can define the font of the meter labeling.

Color areas:
This button opens the **Configure color areas** dialog: In this dialog you can define a color for each area of the scale.
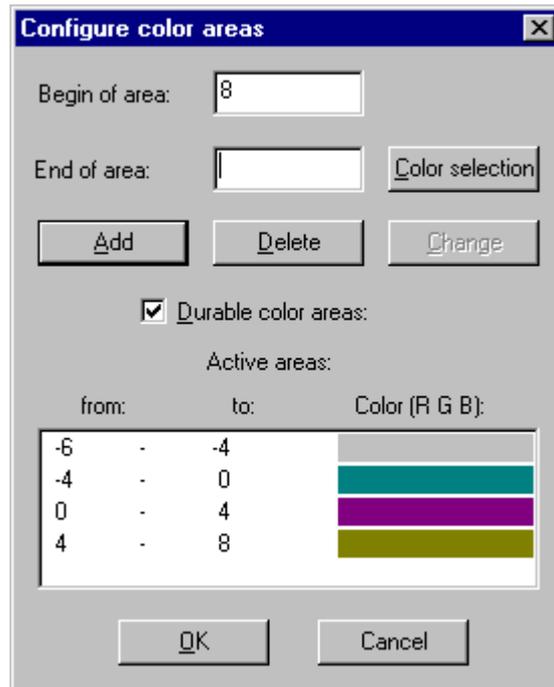
Figure 5-12: Dialog for configuring a color area

Begin of area, End of area:
Enter the start and end value of the scale range you want to assign the color to be defined as follows.

Color selection:
This button opens a standard dialog where you can select a color. Confirm your color selection with 'OK'. The dialog is closed. Then click on the **Add** button which will display the color and the selected scale range in the 'active areas' window. To remove a color area, highlight it in this window and then click on the **Delete** button.

If the option **Durable color areas** is marked with a checkmark, the defined color areas are always displayed. Otherwise, in online mode only the color of the scale range containing the current variable value is displayed.

Label:
Depending on the selected option "Inside" or "Outside", the scale labeling is displayed on the inner or outer side of the scale curve.

Further settings:

Frame inside, Frame outside:
Selecting one of these options (or both) will provide an inner and/or outer boundary line for the scale.

Additional arrow:
In addition to the pointer a small arrow is displayed directly on the scale.

5.5   Bar display

Inserting a bar display

Menu item "Insert / Bar display" or icon:

Using this command you can insert a bar display into your current visualization. The bar display is used to display variable values in the form of a bar moving along a horizontal scale.

With the left mouse button pressed, drag the element to the desired size. Before the element is represented with its final appearance, the configuration dialog 'Configure Bar Display' is displayed. Here you can define various parameters and view their effects to the representation in a preview before finally inserting the element by clicking on 'OK'.

Configuring a bar display

The 'Configure Bar Display' dialog automatically appears after a bar display element is inserted into the visualization in order to display a variable value by the length of a bar along a horizontal or vertical scale. A preview displays the appearance of the element depending on the parameter settings.
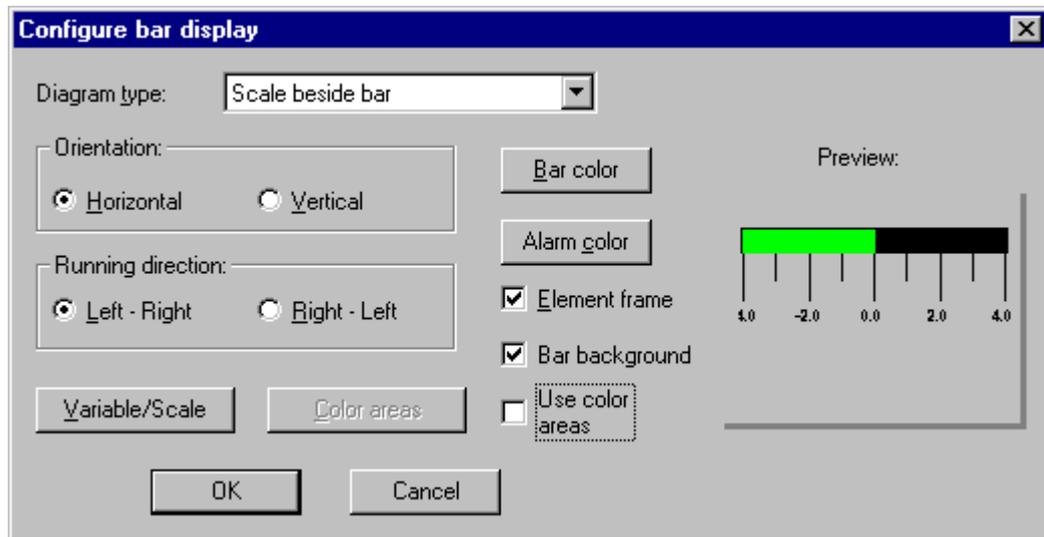
Figure 5-13: Dialog for configuring a bar display

Diagram type:
Select between 'Scale beside bar', 'Scale inside bar' or 'Bar inside scale'.

Orientation:
Select whether a **horizontal** bar or a **vertical** bar shall be displayed.

Bar color:
This button opens a standard dialog where you can select a color for the bar when it is not in alarm state (refer to "Alarm color"). If the option 'Use color areas' (see below) is selected, this option is deactivated and no inputs are possible.

Alarm color:
This button opens the **Configure Alarm** dialog where you can define the alarm color and the threshold value for the visualized variable from which on the bar changes to the alarm color. For this purpose, enter the desired threshold value into the **text input field** and select one of the conditions **greater than** or **less than** to define whether the alarm state shall be triggered by values above or below this value. Click on the **Alarm color** button to select the alarm color in the standard color selection dialog. Close both dialogs with 'OK' to confirm your inputs and to return to the main configuration dialog for the bar display. If the option 'Use color areas' (see below) is selected, this option is deactivated and no inputs are possible.

Variable/Scale:
This button opens the same **Configure scale and variable** dialog as it is displayed for the meter (refer to chapter 5.4 "Meter").

Element frame:
If this option is selected, the bar display is surrounded by a frame.

Bar background:
If this option is selected, the entire display area defined is displayed as a black bar behind the current one. Otherwise only the bar visualizing the present value is displayed.

Use color areas:
If this option is selected, the color area settings defined using the 'Color areas' button are used rather than the color settings defined under 'bar color' and 'alarm color'. As already described for the meter element, the 'Configure color areas' dialog is opened for this purpose. See also the descriptions for the meter settings in the previous chapter.

5.6 Histogram

Inserting a histogram:

Menu item "Insert / Histogram" or icon:

Using this command you can insert a histogram into your current visualization. Using this element all elements of an array can be displayed side by side in the form of bars the height of which varies according to the variable value.

With the left mouse button pressed, drag the element to the desired size. Before the element is represented with its final appearance, the configuration dialog 'Configure Histogram' is displayed. Here you can define various parameters and view their effects to the representation in a preview before finally inserting the element by clicking on 'OK'.

Configuring a histogram:

The histogram element can be used to visualize an array. The values of the array elements can be displayed side by side as vertical bars or lines.
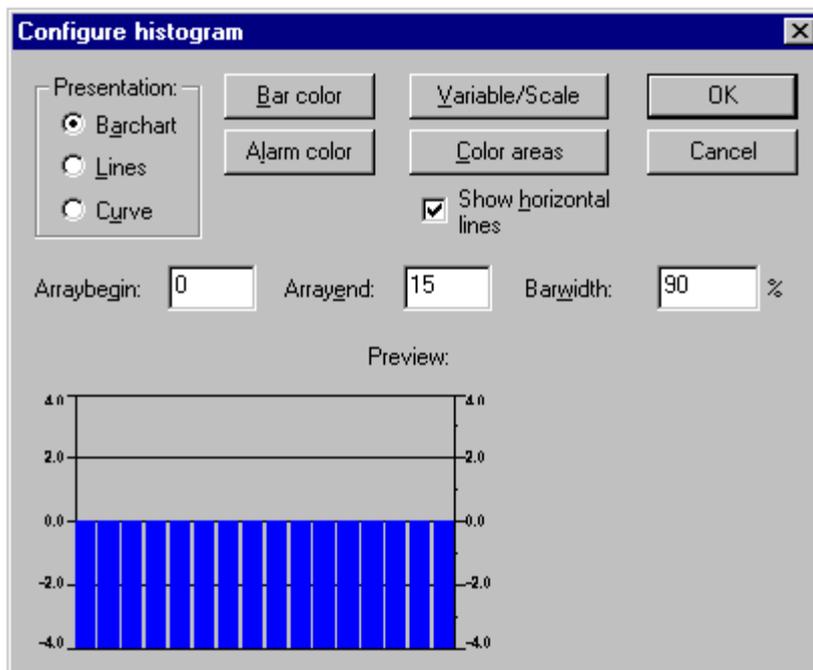
Figure 5-14: Dialog for configuring a histogram

The 'Configure Histogram' dialog automatically appears after a histogram element is inserted into the visualization. A preview displays the appearance of the element depending on the parameter settings.

Presentation:
Select one of the option **Barchart, Lines** or **Curve**.

Show horizontal lines:
If this option is selected, horizontal lines corresponding to the scale division are provided between the left- and right-hand scales.

Bar color:
This button opens a standard dialog where you can select a color for the bar when it is not in alarm state (refer to "Alarm color"). If the option 'Use color areas' (see below) is selected, this option is deactivated and no inputs are possible.

Alarm color:
This button opens the **Configure Alarm** dialog where you can define the alarm color and the threshold value for the visualized variable from which on the bar changes to the alarm color. For this purpose, enter the desired threshold value into the **text input field** and select one of the conditions **greater than** or **less than** to define whether the alarm state shall be triggered by values above or below this value. Click on the **Alarm color** button to select the alarm color in the standard color selection dialog. Close both dialogs with 'OK' to confirm your inputs and to return to the main configuration dialog for the bar display. If the option 'Use color areas' (see below) is selected, this option is deactivated and no inputs are possible.

Variable/Scale:
This button opens the **Configure scale** dialog: See also the descriptions in chapter 5.4 "Meter".

Color areas:
This button opens the **Configure color areas** dialog: As already described for the meter element (refer to chapter 5.4 "Meter"), in this dialog you can define a color for each area of the scale.

Arraybegin / Arrayend:
In this fields, enter the first and last index in array.

Barwidth:
In this field, enter the width of the bars as a percentage of the total width available for the bars.

Example:

The following figure shows an example for an online representation of a histogram displayed as 'lines' and as 'bar chart' for an Array [0..4] of INT.
For this purpose, the following settings have been made: Arraybegin "0", Arrayend "4", scale start "-4", scale end "24", main scale "2", sub scale "1". For the scale range from 0 to 8 another background color has been selected than for the rest of the scale. Furthermore, the bars shall have a blue alarm color if the variable value becomes higher than 8.
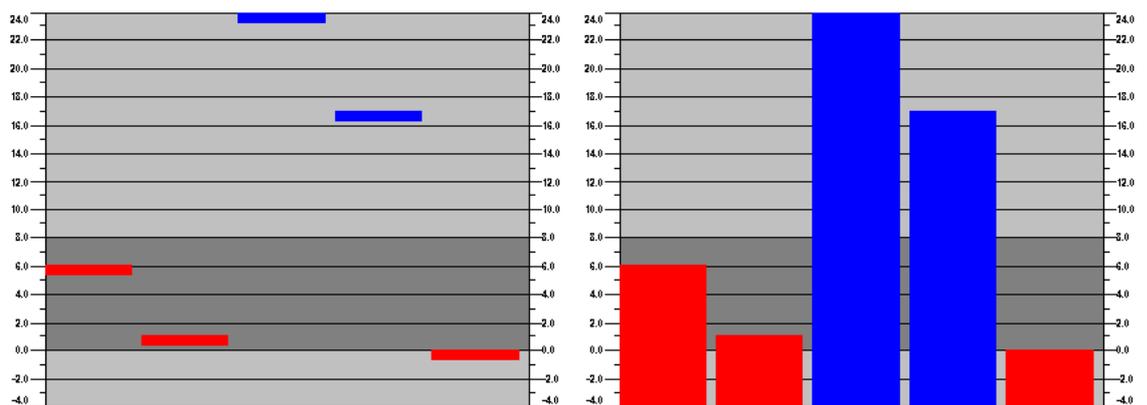
Figure 5-15: Example online representation of a histogram

5.7 Trend element

Inserting a trend element:

Menu item "Insert / Trend" or icon:

Using this command you can insert a trend element into your current visualization. With the left mouse button pressed, drag the element to the desired size. The configuration of the representation (axes, variables, history) is performed in the "Trend" category of the configuration dialog.
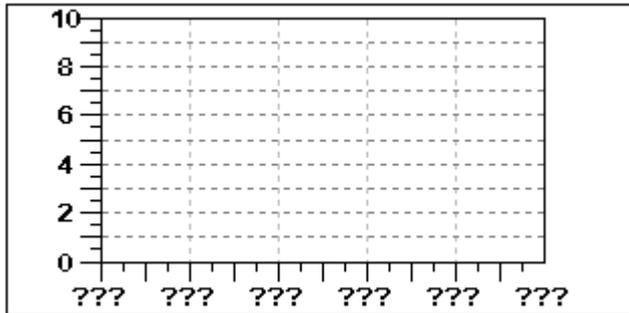
Figure 5-16: Inserting a trend element

The trend or oscilloscope element is used to represent variables over a specific period of time. The trend element stores the data on the client side and represents them as graphs. As soon as a value changes within the graphs, a new entry containing the date/time and the new variable values is written to the file. The trend element is drawn transparently which allows to use any background (color, bitmap).

Configuring a trend:

The trend function is used to record the variation of variable values with time during online mode. It can be compared to the trace functionality. The online representation is displayed as a diagram. During their recording in a text file, the measurement values are listed in lines one below the other.

In order to configure the trend element, select the "Trend" category, press the right mouse button and then select "Configure" from the context menu. The 'Configure Element' configuration dialog appears. Aside from the categories 'Tooltip' and 'Security' which are also available for other elements, the 'Trend' category is additionally available for defining the content and representation of the trend.

Category 'Trend':
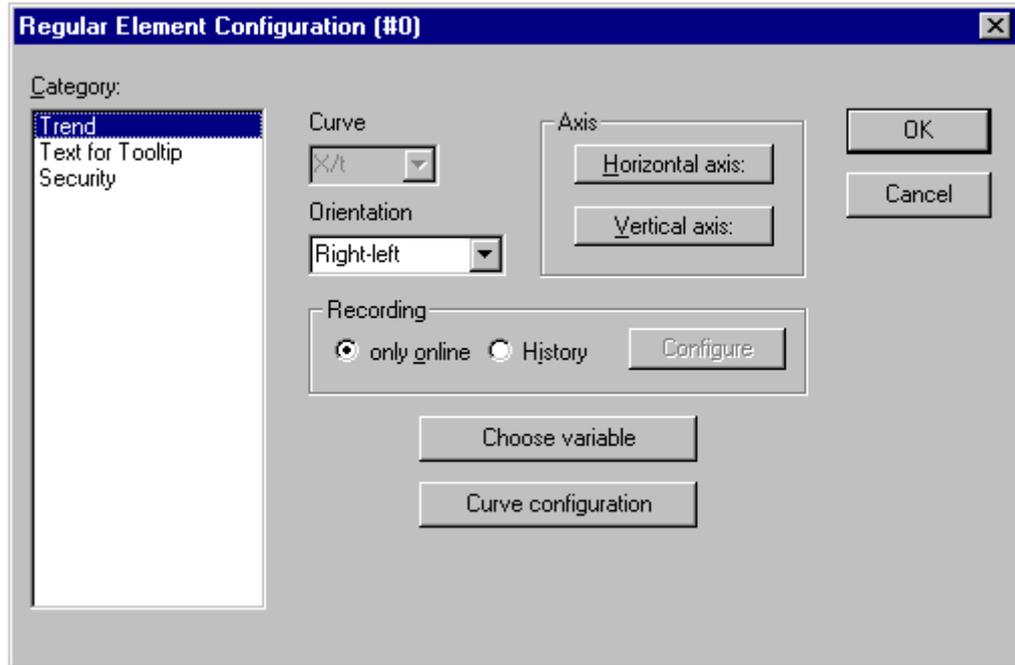
Figure 5-17: Dialog for configuring a trend, category 'Trend'

The following representation settings can be performed in the **Trend** category:

Curve:
Currently only X/t is possible.
Horizontal axis = time axis; vertical axis = value scale

Orientation:
Select between **right-left** and **left-right**.
The current value is represented on the left- or on the right-hand side.

Axes:

Horizontal axis:
This button opens the configuration dialog for the horizontal axis:
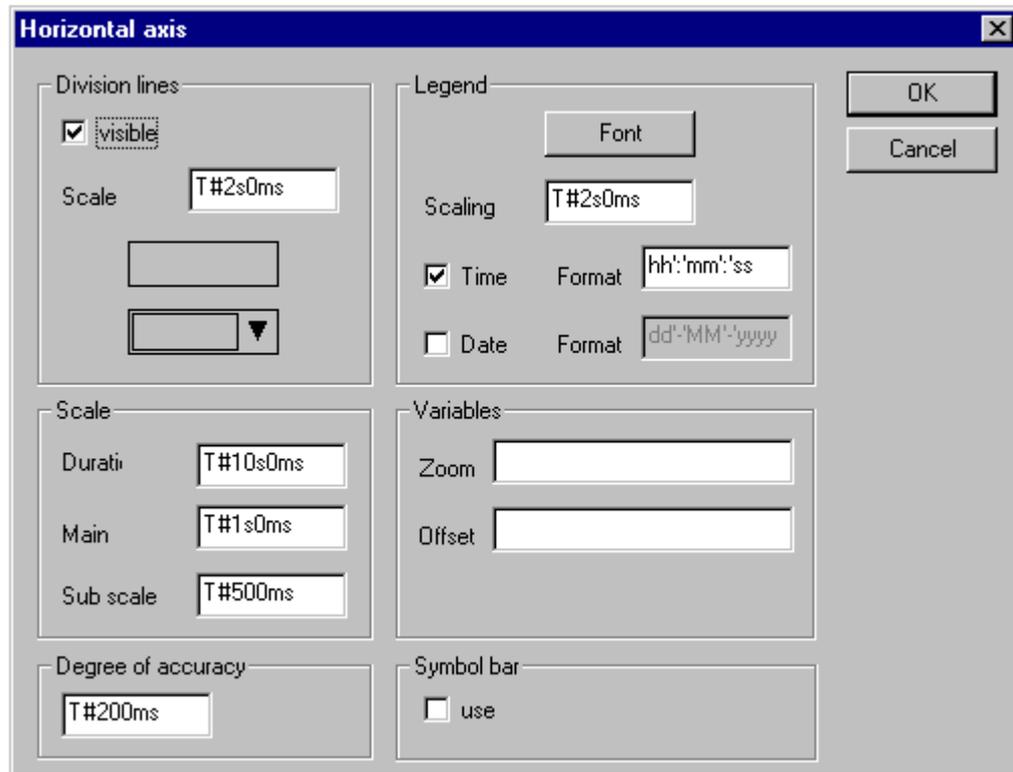
Figure 5-18: Dialog for configuring the horizontal axis

Division lines:
Select the option **visible** if you want vertical division lines to be displayed as an extension of the scale division lines. In this case you can define the **Scale**:
The number entered in this field defines the space between the subdivision lines along the horizontal axis. The representation (normal ___, dashed _ _ _, dotted ......, dash-dotted _ . _ .) and color of the lines can be defined by clicking on the rectangle showing the current line representation or by clicking on the field showing the current color in order to open a corresponding selection dialog.

Scale:
The displayed value range of the scale is defined by the entry in the field **Duration**. For instance, a value of T#10s0ms results in a scale range of 10 seconds. The **Main** and **Sub scale** to be displayed as longer and shorter scale marks are specified within the same syntax.

Degree of accuracy:
In this field, enter the recording interval to be used for the recording of the current values of the variable. Use the standard format for time values (minimum = 200 ms: T#200ms).

Legend:

In this area, the representation of the labeling can be defined. Clicking on the **Font** button opens the 'Font' standard dialog.

In the **Scaling** field you can define the intervals of the scale labeling (e.g. T#5s if you want to have the scale marks labeled in intervals of 5 seconds).

The labeling can contain the **time** and/or the **date**, depending on the selected option(s).

The desired **format** can be defined in the text field using the standard syntax for time and date.

Variables:

Here you can enter project variables containing the **Zoom** and **Offset** values for the horizontal scale. For example, a variable value of "10" will set the start of the display range of the vertical axis to "10".

Symbol bar:

If the option **use** is selected, a horizontal toolbar is displayed at the lower edge of the element. The toolbar contains buttons for scrolling step by step to the left or to the right and for jumping to the left or to the right end of the recording range (time axis) as well as two buttons for zooming in online mode. If a variable has been defined for the zoom function (refer to "variables" mentioned above), this variable contains the current zoom value.

Vertical axis:
This button opens the configuration dialog for the vertical axis:
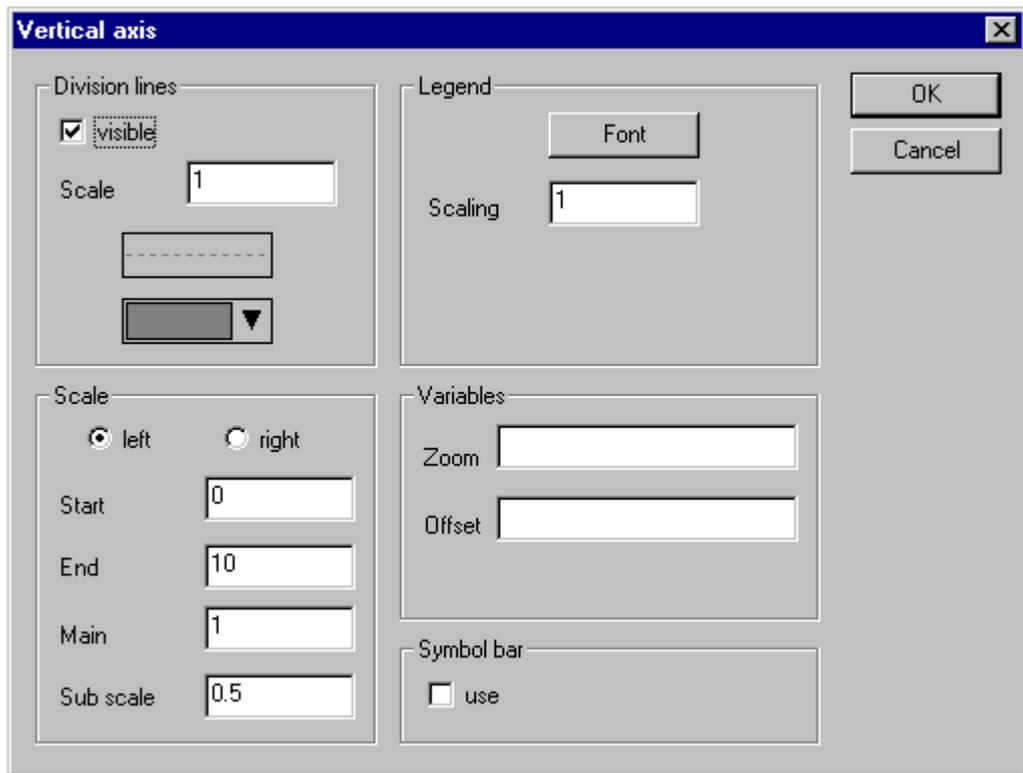
Figure 5-19: Dialog for configuring the vertical axis

Division lines:
Refer to the above description for the horizontal axis.

Scale:
Select whether you want the scale to be displayed on the **left** or on the **right** edge of the trend diagram.
Enter the **Start** value (lower end) and the **End** value (upper end) of the scale as well as the **Main** and the **Sub scale** (the longer and shorter scale marks are displayed as specified here).

Legend:
For descriptions of the **Font** button and the **Scaling** field please refer to the above description for the horizontal axis.

Symbol bar:
Refer to the above description for the horizontal axis.

Recording:

Here you can select whether you want the trend recording to be performed **'only online'** which means that the variation of the variable values with time is displayed in the selected scale range or whether you want the trend recording to be stored in a history which can be configured after clicking on the **Historie** button. This dialog corresponds to the dialog used for the alarm storage configuration.
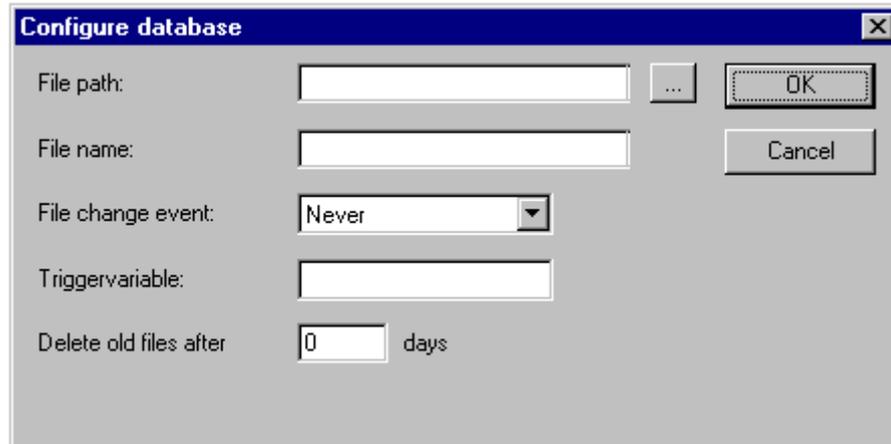
Figure 5-20: Dialog for configuring the history

In the recording file, one line containing the names and values of all recorded variables is written for each measurement. Every line starts with a unique identification in DWORD format which is derived from the date of measurement.

Choose variable:

After clicking on this button, the **Variables** dialog is opened. This dialog is used to configure for which variables trend recording shall be performed and how they shall be displayed there.
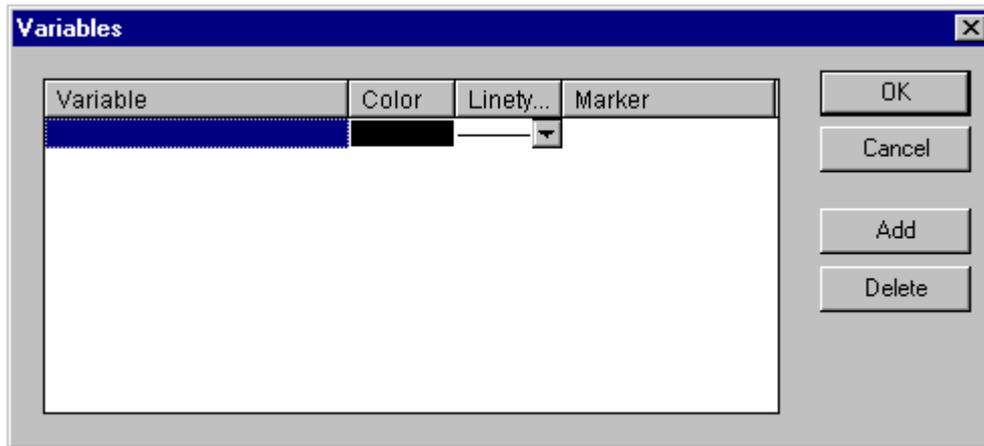
Figure 5-21: Dialog for selecting the variables for the trend element

Enter a project variable in the **Variable** column (an editing frame is opened after clicking on the field; use of the help manager or the Intellisense function is recommended). The line color and type used to display the variable in the record can be defined by clicking on the corresponding field in the **Color** column (standard dialog for color selection) and by selecting a line type from the list of the corresponding field in the **Linetype** column (normal ___, dashed _ _ _, dotted ....., dash-dotted _ . _ .).

In the **Marker** column you can define a variable which displays the value read from the recording when using the bookmark function in online mode. For this, a small gray-shaded triangle is displayed in the upper left corner of the recording diagram. If you select this triangle with the cursor and then keep the mouse button pressed, you can move a vertical line horizontally to the recording instants. The variable defined as 'bookmark' will then read the corresponding value from the respective recording curve.

Do this for all desired variables. With each click on the **Add** button, one more line for a variable entry is appended at the end of the list. Using the **Delete** button, marked lines can be deleted from the list.

Curve configuration:

This button opens the **Curve configuration** dialog. In this dialog you can do general settings for the representation of the trend curves.
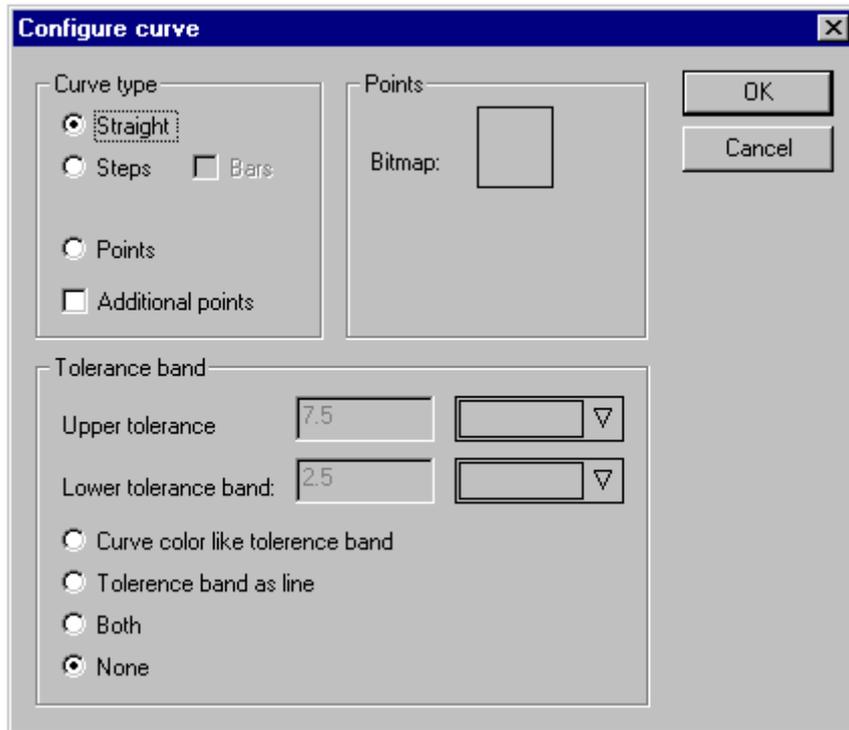
Figure 5-22: Kurvenkonfiguration dialog for the trend element

Curve type:

Select one of the options **Straight, Steps** or **Points**. For the first two types you can also display additional points by selecting the option **Additional points**. The representation of **Points** has to be defined by selecting a bitmap file. For this, click on the rectangle labeled with **Bitmap** to open the standard file selection dialog.

Tolerance band:

For the upper and/or the lower tolerance band you can define a limit value on the scale as well as a color (a color selection dialog appears after clicking on the **Color** button). Select the option **Curve color like tolerence band** if you want to have the tolerance bands displayed during online operation. Select the option **Tolerence band as line** if you want to have the recording curve to be displayed with the color of the corresponding tolerance band if the tolerance band is exceeded. Select **Both** or **None** if you want to select both display options mentioned before or none of the two display options.

Example:

Display of a trend element in online mode.

It has been configured as follows:

Orientation: Right-left, history selected

Horizontal axis:

      Division lines: T#2s,

      Duration T#10s,

      Main: T#1s,

      Sub scale: T#500ms,

      Degree of accuracy: T#200ms,

      Legend: Time ('hh':'mm':'ss') and Date ('dd'-'MM'-'yyyy') with an Scaling of T#5s.

      Symbol bar activated.

Vertical axis:

      Division lines: 20, dashed, gray;

      Scale on the left,

      Start: 0,

      End: 100,

      Main: 10,

      Sub scale: 5;

      Legend Scaling: 10;

Variables:

      Zoom: PLC_PRG.ivar_zoom

      Offset: PLC_PRG.ivar_offset

Symbol bar: deactivated

Choose variable:

      1. Variable: PLC_PRG.ivar1, green line, Bookmarks: PLC_PRG.ivar_lz1

      2. Variable: PLC_PRG.ivar2, orange line, Bookmarks: PLC_PRG.ivar_lz2

Curve configuration:

      Straight,

      Lower tolerance band: 20/blue,

      Upper tolerance band: 80/red,

      Option "Both" selected

In online mode:

Recording is running from the right to the left, the latest value on the right; the current value is added to the curve every 200 milliseconds. The variables PLC_PRG.ivar_lz1 and ...lz2 define the values of the green and the red curve for the point in time marked by the bookmark line on the time axis. Using the buttons below the diagram you can scroll and zoom into the diagram.
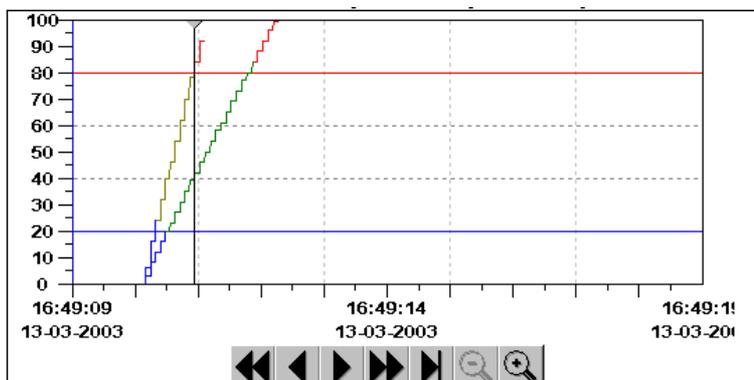
Figure 5-23: Example of an online representation of a trend element

6   Alarm handling with 907 AC 1131 V5.0 onwards

6.1   Alarm configuration

6.1.1 Alarm configuration, survey

The alarm system integrated in 907 AC 1131 allows to detect critical process states, to record these states or to illustrate the states for the user by means of a visualization.

Configuration is performed via the 'Alarm Configuration' of the Object Organizer in the Resources tab.

There, the **alarm classes** and **alarm groups** are defined.
The **alarm class** is used to classify an alarm. It describes specific parameters that are all together allocated to an alarm by assigning the class type. For this, the alarm is configured in an alarm group.
The **alarm group** is used to configure one or more alarms for use in the project. The alarms are assigned with specific classes and other parameters. This way it is possible to structure the available alarms. The different alarm groups are inserted and defined by the user under the subtopic 'System'.

For the visualization of alarms the '**alarm table**' element is available in the visualization. In this table the user is able to monitor and confirm alarms.

To obtain a history (i.e. recording of alarm events in a log file), this has to be specified and the storing behavior has to be defined for each group.

If you select the 'Alarm configuration' entry from the resources, the **'Alarm Configuration'** dialog is opened divided in two windows the functionality of which is like for the task configuration. On the left-hand side the configuration tree is displayed and on the right-hand side the corresponding configuration dialog of the entry selected in the tree.
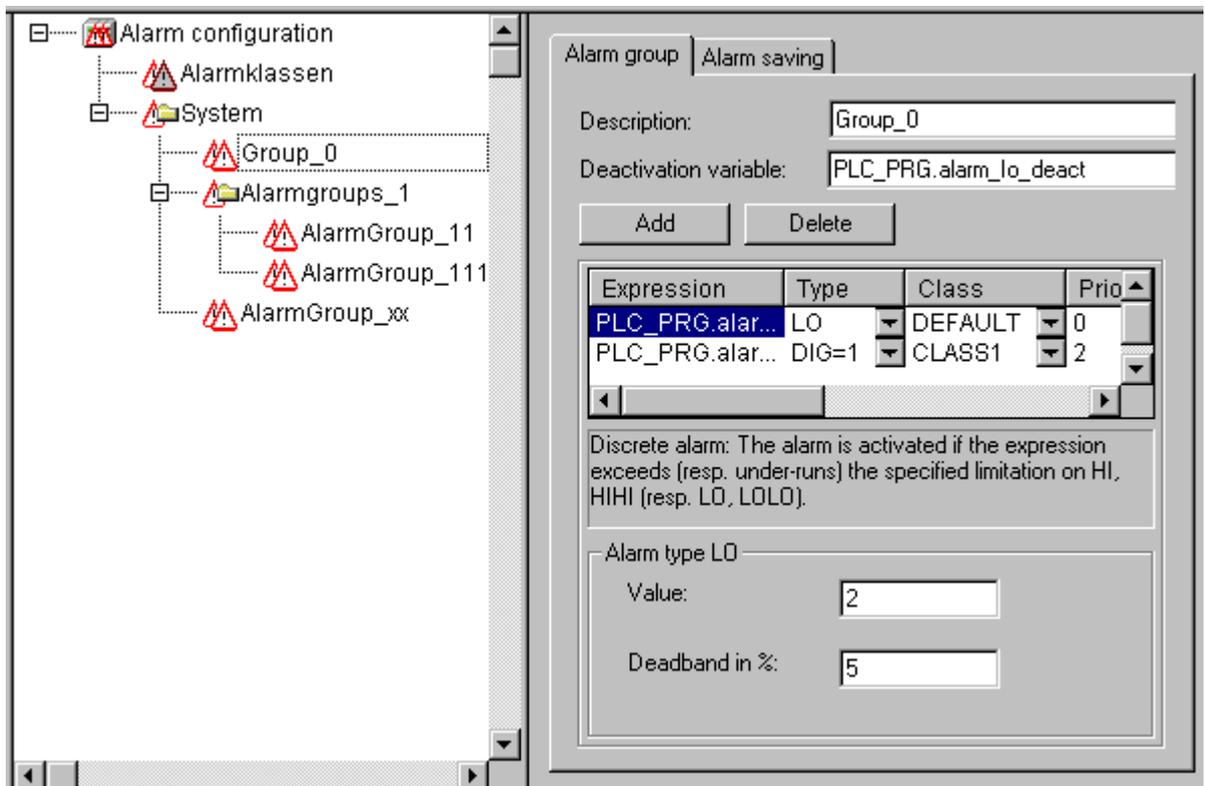
Figure 6-1: Alarm configuration example

Open the present tree by clicking with the mouse on the plus sign in front of the 'Alarm configuration' entry. For new configurations it contains only the entries 'Alarm classes' and 'System'.

6.1.2 Overview of alarms, terms

The use of an alarm system in 907 AC 1131 follows the following generally applicable descriptions and definitions regarding alarms:

Alarm:
An alarm is generally considered as a special condition (value of an expression).

Priority:
The priority or "severity" of an alarm describes the weight or importance of an alarm condition. The highest priority is "0", the lowest possible definition is "255".

Alarm condition:
An expression/variable configured for alarm monitoring can have the following states: NORM (no alarm condition), INTO (alarm condition has just occurred, "alarm is coming"), ACK (alarm occurred and has been acknowledged by the user), OUTOF (alarm condition has stopped again, "alarm disappeared" but has not yet been acknowledged).

Subcondition:
An alarm condition can have limits (Lo, Hi) and "extreme" limits (LoLo, HiHi). Example: The value of an expression increases and first exceeds the Hi limit which trips a Hi alarm. If this value increases further and exceeds the HiHi limit before the Hi alarm has been acknowledged, the Hi alarm is automatically acknowledged internally and the alarm list (internal list for alarm administration) does now only contain the HiHi alarm condition. In this case, the Hi condition is designated as subcondition.

Alarm acknowledgement:

One major application of alarms is to inform the user about an alarm situation. With this, it is often necessary to guarantee that the user has actually received the notification (refer to possible actions for the alarm class configuration). The user has to "acknowledge" the alarm before it can be deleted from the alarm list.

Alarm event:

An alarm event must not be confused with an alarm condition. An alarm event describes the momentary occurrence of a change, e.g. from normal to alarm condition, while an alarm condition can be valid over a longer time.

907 AC 1131 supports the following possibilities:

- Deactivation of the alarm generation for individual alarms or for complete alarm groups.

- Selection of the alarms to be displayed by means of alarm groups and the alarm priority of individual alarms.

- Storage of all occurred alarm events.

- Visualization element 'alarm table' of the 907 AC 1131 visualization.

6.1.3 Alarm classes

Alarm classes are used to generally describe particular alarm criteria, such as the philosophy of acknowledgement (alarm confirmation by the user), the execution of actions (what action shall be executed automatically on the occurrence of particular alarm conditions (refer to chapter 6.1.2 "Overview of alarms, terms") and the visualization in the alarm table (refer to chapter Visualization). Alarm classes are globally defined in the alarm configuration (refer to chapter 6.1.1 "Alarm configuration, survey") and then available as a "basic configuration" for each alarm group.

Configuration of alarm classes:

Highlight the entry 'Alarm classes' in the alarm configuration tree. The 'Alarm classes' configuration dialog is opened:
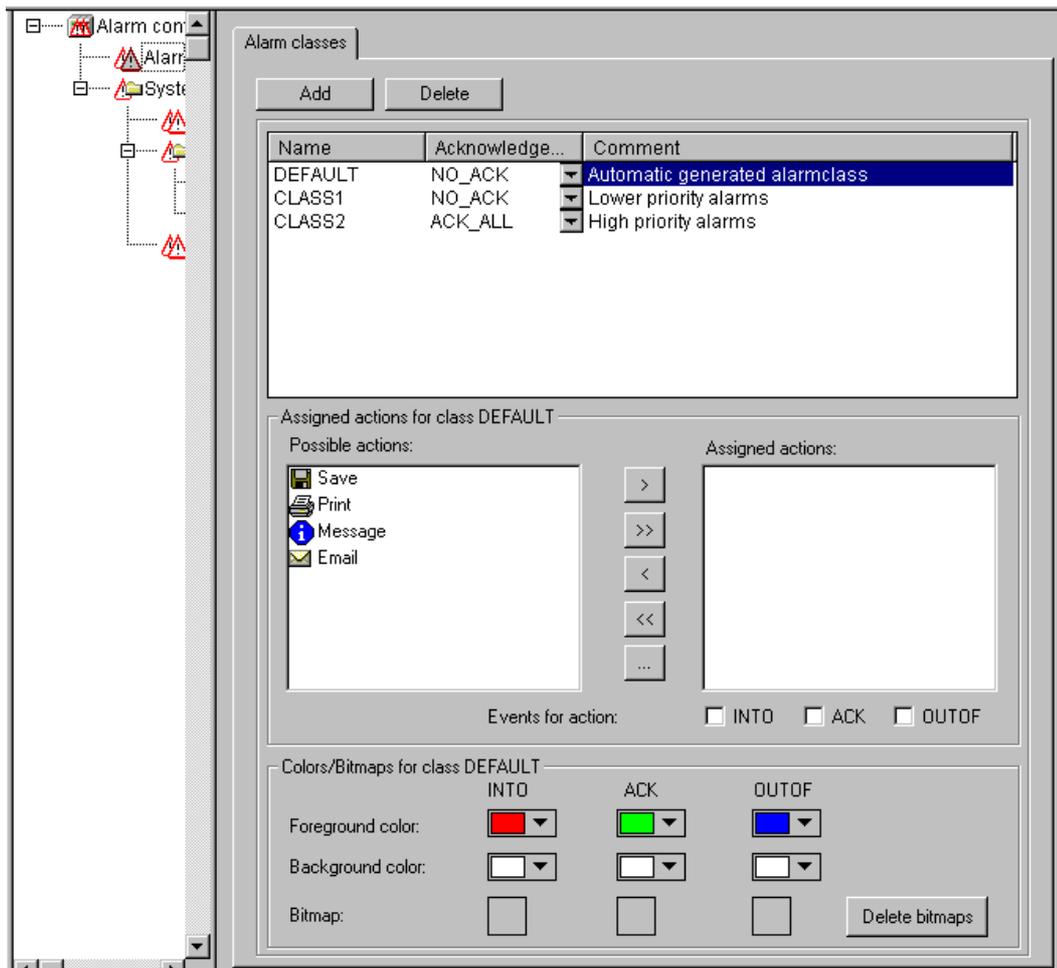
Figure 6-2: Configuration dialog 'Alarm classes'

Press the **Add** button to create a new alarm class. As a result, a new line is inserted in the upper window which first only displays the setting "NOACK" (no acknowledgement) in the 'Acknowledgement' column. Assign a name for the alarm class by clicking with the mouse in the field below **Name** to open an editing frame and select another acknowledge type from the selection list below **Acknowledgement**, if necessary.

Note regarding the acknowledgement: It is not enough to confirm the message boxes indicating the occurrence of the alarm with the configured message text (see below). Alarms have to be acknowledged either in the alarm list which is available for the user during online mode in an alarm table (visualization) or in a log file (definition inside the alarm group) from which the date (DWORD) of the alarm to be acknowledged can be read.

Possible types of acknowledgement:

NO_ACK:	No alarm acknowledgement by the user required.
ACK_INTO:	An "appearing alarm condition" (status "INTO", alarm occurs) has to be acknowledged by the user.
ACK_OUTOF:	A "disappearing alarm condition" (status "OUTOF", alarm finished) has to be acknowledged by the user.
ACK_ALL:	Appearing and disappearing alarm conditions have to be acknowledged by the user.

Furthermore you can enter a **comment**.

Entries for further alarm classes will be appended to the end of the list.

Clicking on the **Remove** button deletes the currently selected entry from the list.

Actions assigned to classes:

Each alarm class can have an assigned list of actions to be initiated on the occurrence of alarm events (refer to chapter 6.1.2 "Overview of alarms, terms"). The following action types are supported:

Save:	The alarm event is internally stored, e.g. to be output into a log file (refer to chapter 6.1.5 "Alarm storage"). Please note: For this, it is necessary that this file has been previously defined in the alarm group configuration!
Print:	The alarm event is output to a printer.
Message:	The current visualization displays a message window containing the text to be defined here.
E-Mail:	An email is sent.

In the list of **'Possible actions'**, highlight the desired action and move it to the **'Assigned actions'** field by clicking on the "**>**" button. Using the "**>>**" button you can select all actions at once. In the same way you can also remove one or all actions from the selection again by using the button "**<**" or "**<<**". If a selected action is highlighted in the list, you can open a corresponding dialog by clicking on the "**...**" button where you can define the desired email address and printer settings as well as a message text.

Definition of the message for an action:

In the fields **Message text** (for action 'Message'), **Message** (for action 'E-Mail') or **Text output** (for action 'Print'), create a template consisting of dummy parameters which defines the final alarm message in combination with the message text specified for the corresponding alarm (specified in the 'Message' field of the alarm group). Line breaks in the fields 'Message' and 'E-Mail' have to be inserted using the key combination <Ctrl>+<Return>. The following dummy parameters can be used:

MESSAGE	The message text defined for the alarm in the alarm group configuration is output.
DATE	Date of change to the corresponding status (INTO).
TIME	The time of the alarm occurrence is output.
EXPRESSION	Expression (defined in the alarm group) which tripped the alarm.
PRIORITY	Alarm priority (defined in the alarm group).
VALUE	Current value of the expression.
TYPE	Alarm type (defined in the alarm group).
CLASS	Alarm class (defined in the alarm group).
TATGETVALUE	Target value for alarm types DEV+ and DEV- (defined in the alarm group).
DEADBAND	Tolerance of the alarm (defined in the alarm group).
ALLDEFAULT	All information about the alarm is output as described for the output into a stored file (history) (refer to chapter 6.1.5 "Alarm storage").

Example:

Define the message box (action 'Message') by entering the following into the message window:
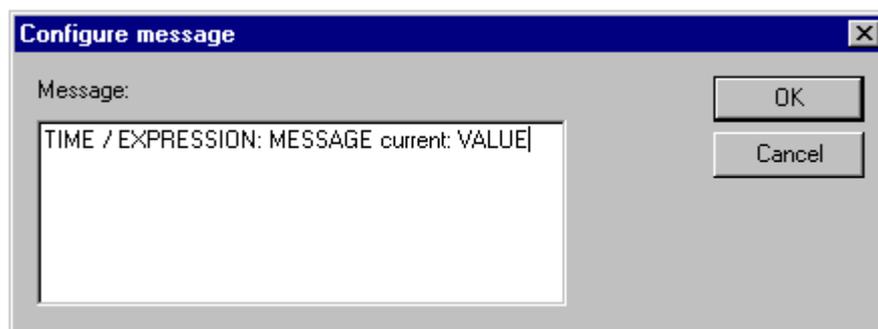
Furthermore, at the alarm definition in the alarm group, enter the following into the 'Message' table field: "Temperature critical !". The alarm message is then output as follows:
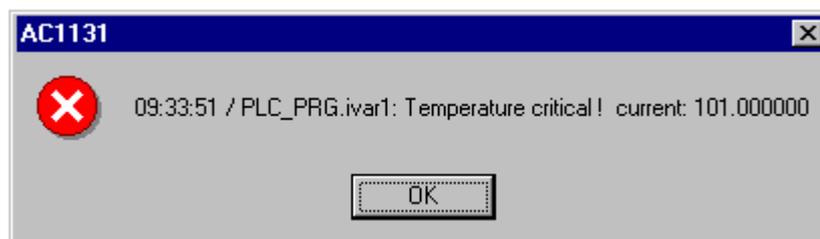
Note:

Using a *.vis file or a translation file *.tlt, the message text can also be considered when the language of the project is switched. However, to enable a string to be entered into the translation file *.tlt, a "#" sign has to be inserted at the beginning and at the end of the corresponding character string, as for all visualization-related texts (e.g. for the example shown above: "#Temperature critical !#" and "TIME /EXPRESSION: MESSAGE #current#: VALUE" in order to get the corresponding text parts as ALARMTEXT_ITEM into the translation file).

A **storage file** for the action 'Save' is defined within the alarm group configuration.

Events for an action:

For each action it has to be determined by which alarm event it is initiated (refer to chapter 6.1.2 "Overview of alarms, terms").

Activate one of the following events:

INTO:	The alarm occurs.
ACK:	The alarm is acknowledged by the user.
OUTOF:	The alarm condition is stopped.

Colors/bitmaps for a class

For each alarm class different colors and bitmaps can be assigned which are used to distinguish the alarms when visualizing the alarm table (refer to chapter 6.1.1 "Alarm configuration, survey"). Select the fore- and background colors for the possible events INTO, ACK and OUTOF (see above). For this, you can open the color selection standard dialog by clicking on the arrow icons or the bitmap file selection dialog by clicking on the corresponding gray square.

6.1.4 Alarm groups

Alarm groups are used to structure different alarms. Each alarm is assigned to one alarm group and managed by this group. For all alarms of one group a common deactivation variable as well as common parameters regarding the alarm storage can be assigned. The group can also be used to structure the available alarms. Even a single alarm has to be configured in an alarm group.

To establish a hierarchical alarm group structure in the alarm configuration, folder elements can be used.

Configuring an alarm group:

The **Alarm group** dialog is automatically displayed when an alarm group is selected in the alarm configuration tree:

Figure 6-3: Configuration dialog 'Alarm group'

Description:
Here you can enter a name for the alarm group.

Deactivation variable:
Here a boolean project variable can be entered which deactivates alarm triggering for **all** alarms on the occurrence of a rising edge and which activates alarm triggering again when a falling edge occurs.

Using the **Add** button, individual alarms which are defined by the following parameters can be added to the group.

Parameters for an individual alarm:

Expression:
Project variable to which the alarm is related. For a correct input use the input aid (<F2>) or the Intellisense function. You can also enter an expression (e.g. "a + b").

Type:

The following alarm types can be used (please note the individually assigned comment displayed below the table):

DIG=0	Digital alarm. Becomes active if the value of the expression becomes FALSE.
DIG=1	Digital alarm. Becomes active if the value of the expression becomes TRUE.
LOLO	Analog alarm. Becomes active if the value of the expression falls below the value specified under 'Alarm type LOLO'. A tolerance range can be specified in per cent of the value. If the value falls below the LOLO value but is still within this tolerance range, no alarm is initiated.
LO	See description for LOLO.
HI	Analog alarm. Becomes active if the value of the expression exceeds the value specified under 'Alarm type HIHI'. A tolerance range can be specified in per cent of the value. If the value exceeds the HIHI value but is still within this tolerance range, no alarm is initiated.
HIHI	See description for HI.
DEV-	Deviation (from target value). The alarm becomes active if the value of the expression falls below the target value specified under 'Alarm type DEV-' + percentage deviation. Percentage deviation = target value * (deviation in %) / 100.
DEV+	Deviation (from target value). The alarm becomes active if the value of the expression exceeds the target value specified under 'Alarm type DEV-' + the specified deviation. Percentage deviation = target value * (deviation in %) / 100.
ROC	Rate of change (changes per unit of time). The alarm becomes active if the value of the expression has changed too much compared with the previous value. The limit value of the intensity of change which triggers the alarm is defined by the number of units (value change) that change per second, minute or hour.

Class:

Select the desired alarm class. You can select from the classes defined in the alarm class configuration before the project was saved the last time. The class includes a set of parameters e.g. for alarm acknowledgement and storage (refer to chapter 6.1.3 "Alarm classes").

Priority:

Here you can assign priorities from 0 to 152 where 0 is the highest priority. The priority affects the sorting in the alarm table.

Message:

Here you can define the text for the message box that is displayed in case of an alarm. The text is output in combination with the template defined for the selected alarm class (for an example please refer to chapter 6.1.3 "Alarm classes"). **The message box has to be confirmed by the user with 'OK'. This, however, does not automatically acknowledge the alarm!** To acknowledge an alarm, the alarm list has to be accessed which can be done using the 'Alarm table' visualization element (refer to chapter 6.2.1 "Overview of alarms, terms") or using the date of the alarm entry which can be taken from a storage file that can be created optionally (refer to chapter 6.1.5 "Alarm storage").

Deactivation:
Here you can enter a project variable which deactivates the alarm initiation as soon as it receives a rising edge. However, please note that this entry is overwritten by an entry specified in the field 'Deactivation variable' (see above).

6.1.5 Alarm storage

For each alarm group a file can be defined to store the **alarm events** (refer to chapter 6.1.2 "Overview of alarms, terms"), if "storing" in the action list of the affected class (refer to chapter 6.1.3 "Alarm classes") has been activated.

To configure the alarm storage, select the alarm group in the alarm configuration tree and then select the 'Alarm storage' tab in the dialog.
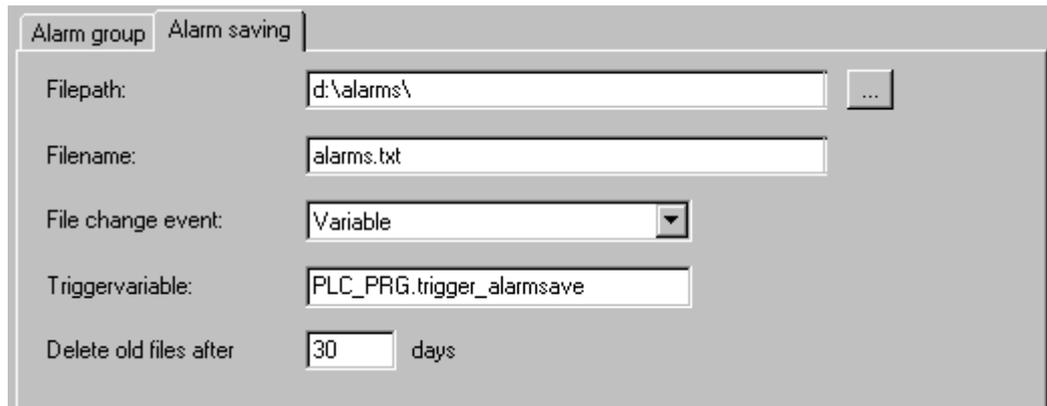
Figure 6-4: Configuration dialog 'Alarm storage'

The following inputs are possible:

Filepath:
Directory path for the file specified under 'File name'. Clicking on the "..." button opens the standard dialog for selecting a directory.

Filename:
Name of the file where the alarm events shall be stored (e.g. "alarmlog"). The file is then automatically created with the file name specified here, which is completed by a number and the extension ".alm". The number specifies the version of the log file. The first file stored is provided with a 0. The following files which are created as a result of the conditions defined under **'File change event'** are numbered with 1, 2, etc. in an ascending order (examples: "alarmlog0.alm", "alarmlog1.alm).

File change event:
Here you can enter the condition for the creation of a new storage file. Possible conditions: Never, after one hour, after one day, after one week, after one month, after a rising edge of the variable specified under 'Triggervariable', after a specific number of entries is reached specified under 'Number of records'.

Trigger variable and Number of records:
Refer to Dateiwechselereignis.

Delete old files after .. days:
Number of days following the creation date after which all alarm storage files will be deleted, except the current file.

The storage file (history) contains the following entries (for examples see the second table row):

Date/time as DWORD	Date	Time	Event	Expression	Alarm type	Limit value	Tolerance	Curr. Value	Class	Priority	Message
10469633326	16:08:03	16:08:52	INTO	PLC_PRG.bLO	HIHI	25	2	13	a_class2	1	Temperature too high

Using the DWORD value of the date, an alarm can be accessed in order to confirm it. The remaining information reflect the parameters of the alarm set in the configuration.

Example of a storage file in text format:

1046963332,6.3.03 16:08:52,INTO,PLC_PRG.ivar5,HIHI,,,,   9.00,a_class2,0,
1046963333,6.3.03 16:08:53,INTO,PLC_PRG.ivar4,ROC,2,,,   6.00,a_class2,2,
1046963333,6.3.03 16:08:53,INTO,PLC_PRG.ivar3,DEV-,,,,   -6.00,a_class2,5,
1046963334,6.3.03 16:08:54,INTO,PLC_PRG.ivar2,LOLO,-35,,3, -47.00,warning,10,warning: low temperature !
1046963334,6.3.03 16:08:54,INTO,PLC_PRG.ivar1,HI,20,,5,   47.00,a_class1,2,temperature too high ! Acknowledge !

6.2 Visualization of alarms using the alarm table

6.2.1 'Insert' 'Alarm table'

Icon:

Using this command you can insert an alarm table into your current visualization.

With the left mouse button pressed, drag the element to the desired size. Before the element is represented with its final appearance, the configuration dialog **'Configure alarm table'** is displayed (refer to chapter 6.2.2 "Configuring the alarm table"). Aside from the categories 'Tooltip' and 'Security' which are also available for other elements, in this dialog the categories 'Alarm table', 'Settings for sorting', 'Columns' and 'Settings for alarm table' are additionally available for defining the content and representation. Using the alarm table you can visualize the alarms defined in the **alarm configuration** (refer to chapter 6.1.1 "Alarm configuration, survey") of the project.

6.2.2 Configuring the alarm table

The **Configure alarm table** dialog is displayed as soon as an alarm table for the representation of alarms defined in the **alarm configuration** (refer to chapter 6.1.1 "Alarm configuration, survey") is inserted into the visualization. Apart from the categories for the Tooltip and Security configuration which are also known for other elements, the following settings for defining the representation and selection behavior in the table can be done.

Category 'Alarm table':
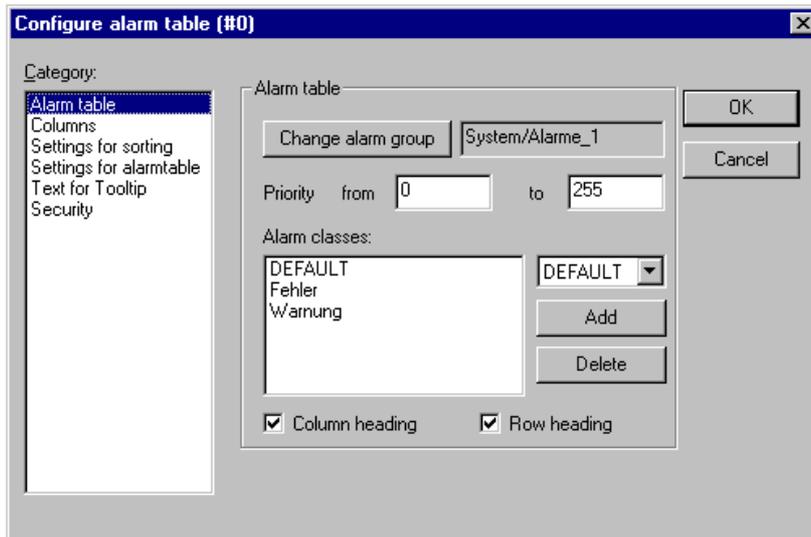
Figure 6-5: Dialog for configuring an alarm table, category 'Alarm table'

Define what you want to have displayed in the table:

Alarm group:
Select the group to be displayed (which can also contain only one single alarm) from the selection tree of the alarm configuration which is opened by clicking on this button. All groups defined in the alarm configuration are offered for selection (refer to chapter 6.1.4 "Alarm groups").

Priority:
Define the priorities the alarms of which you want to have displayed. The maximum possible bandwidth is: **from**: 0 **to**: 255. The priority is defined in the alarm group (refer to chapter 6.1.4 "Alarm groups").

Alarm classes:
Select the class you want to have displayed (defined in the alarm configuration, refer to chapter 6.1 "Alarm configuration") from the selection list and then press the **Add** button to enter the class into the **Alarm classes** window. Do this for all required classes. Using the **Delete** button you can remove a highlighted entry from the window.

Activate the options **Column heading** and/or **Row heading** if you want the headers to be displayed in the table.

Category 'Settings for sorting':
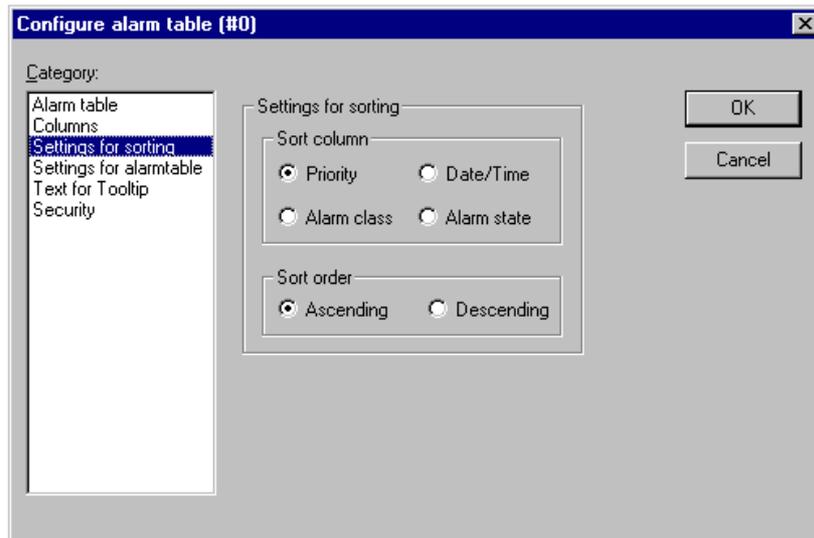
Figure 6-6: Dialog for configuring an alarm table, category 'Settings for sorting'

Here you can define the sorting criteria for the alarm table:

Sorting column:
The table can be sorted according to the priority, alarm class, date/time or alarm condition.

Sorting order:
Ascending or **Descending**. Example: Ascending sorting according to the priority means that the table starts with the alarms of priority 0 (if available) and the alarms with higher priority numbers are sorted subsequently according to their number.

Category 'Columns':
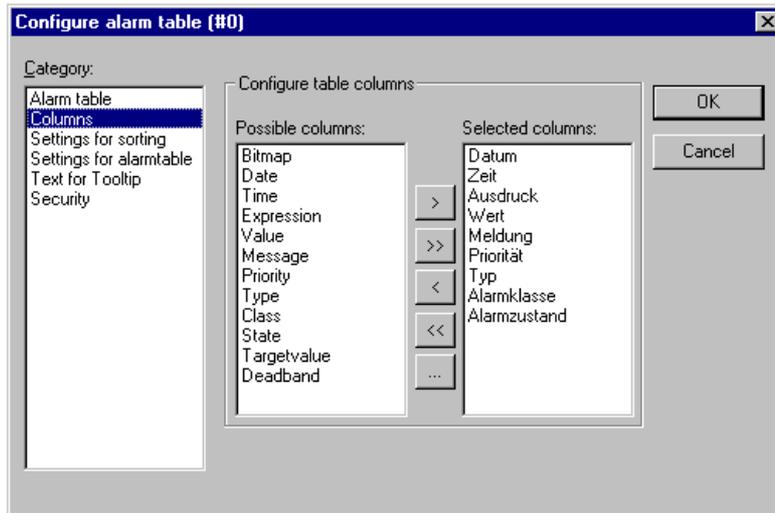
Figure 6-7: Dialog for configuring an alarm table, category 'Columns'

Here you can define which of the following columns (alarm parameters) you want to have displayed in the alarm table: Except of date, time (alarm state change) and alarm state, the parameters are specified in the alarm group configuration: bitmap, date, time, expression, value, message, priority, type, alarm class, alarm state, target value (for the alarm types DEV+ and DEV-) and tolerance.

Using the buttons "**>**" and "**>>**" you can move individual parameters or all parameters from the left-hand to the right-hand window. The selection displayed in the right-hand window is then displayed in the alarm table. Using the buttons "**<**" or "**<<**" the entries can be removed again from the selection.

If the corresponding column entry is highlighted in the right-hand window, the **Configure columns** dialog can be opened for each entry using the **Column properties** button, where you can specify the **column heading** and **column width**.

Category 'Settings for alarm table':
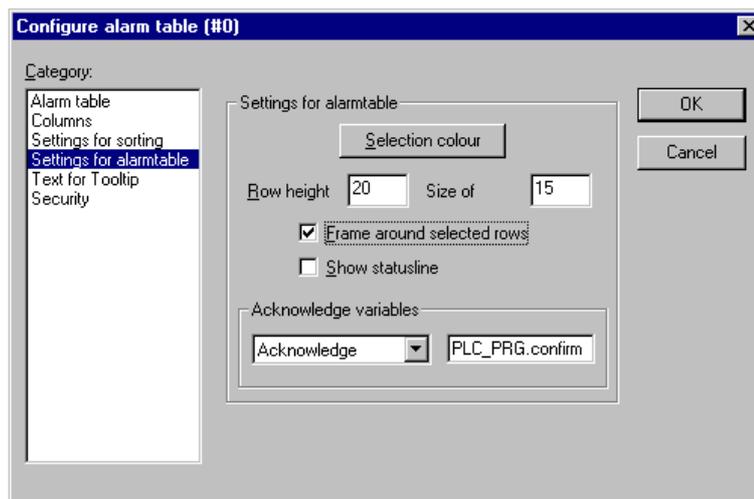
Figure 6-8: Dialog for configuring an alarm table, category 'Settings for alarm table'

Here you define the representation of selected table fields:

Selection color:

This button opens a standard dialog where you can select the color to be used to display selected fields.

Row height:

Height of the table fields in pixel.

Frame around selected rows:

If this option is selected, the selected table row is bordered with a frame.

Show statusline:

If this option is selected, a status bar containing the following buttons for the operation in online mode is displayed below the alarm table:

Acknowledge selection:	All selected alarm entries in the alarm table are acknowledged.
Acknowledge all:	All entries in the alarm table are acknowledged.
History:	If this button is pressed, all events that already occurred (transitions between alarm states) are displayed in the table instead of the current state of all alarms. No acknowledgement can be performed in this list! New events are permanently added. If you have defined a storage file, this history can also be found in this file for all alarm classes which have the action 'Save' assigned.
Start:	Finishes 'Stop' (see below).
Stop:	Stops the permanent update of the list with the new events until the update is initiated again with 'Start'.

Acknowledge variables:

This possibility for configuration is only available, if the option "Display status line" (see above) is not selected. The functions of the buttons described above for the status line can then be controlled by means of variables. To assign the variables, select a function from the selection list and enter a project variable into the input field beside it. For example, in online mode it is then possible to acknowledge all alarms by the rising edge of the assigned variable.

7 Importing projects from 907 PC 331

7.1 Introduction

The programming and test software 907 AC 1131 offers the option to import programs (in IL) which were created for the controllers 07 KR91, 07 KT 92, 07 KT 93 and 07 KT 94 using the programming and test software 907 PC 331. Aim of the import is to avoid laborious typing out of existing programs. However, it will always be necessary to rework the programs. The functionality of the blocks which are not contained in the 907 AC 1131 library ABB-BIB4.LIB must be adapted. A list of these blocks can be found in the block library documentation. In chapter 7.4 "Reworking programs" the peculiarities to be observed are described once again.

To import projects created with 907 PC 331 perform the following steps:

- Create an ASCII file containing the variable list using 907 PC 331
- Create an ASCII file containing the translated IL using 907 PC 331
- Load the variable list into 907 AC 1131 using AC1131_V431.EXE
- Load the program file into 907 AC 1131 using AC1131_V431.EXE
- If necessary, correct the created project using AC1131_V431.EXE
- Open and edit the project using AC1131.EXE V5.x

7.2 Creating the ASCII files for the program and variables using 907 PC 331

To import a project generated with 907 PC 331, the variables list and the instruction list are required as ASCII files.

7.2.1 Creating an ASCII file of the variables list

The ASCII file of the variables list is generated by printing the variables list from the 907 PC 331 into a file. To receive standardized files the following print parameters have to be set:

Menu: Print param./Print parameter

- Page length: 100
- Left margin: 0
- Papersave: Y
- Convers. file: (none)
- Initial. file: (none)
- Re-init file: (none)

Menu: Print param./Print format editor

- No. of header lines: 0
- No. of footer lines: 0

Menu: Print 1/Variable list

- Start variable: (none)
- End variable: (none)
- Start symbol: (none)
- End symbol: (none)
- Sorting Abs./Symb. (A/S):   A
- Print with cover sheet (Y/N): N
- First page: 1
- Print destination: projectname.VAR

The ASCII file is represented as follows: (Examples for variables)

E	Variable	Symbol	Long text
	E 02,00	Eingang E2,0	Input 0 at module 2
	E 62,00	Onboard E0	Input 0 at basic unit
	E 62,09		
	EW 06,00	AnaIn1	Analog input channel 1
	A 62,01		
	AW 02,00		
	M 000,00	I+	
	M 000,01	JR2	AREC_5 received job from node 2
	M 255,15	1.SPS-Zykl	
	MW 000,01	FKT_Feld+0	
	MW 013,04	TXT-Nummer	Text numbers register
	K 00,00	Bit=0	0
	K 00,01	B1	1
	K 00,01	Bit=1	1
	KW 00,00	MAST_SLV	-2
	KW 00,01	INIT_M	0
	KW 00,02	INIT_MW	0
	KW 00,03	INIT_MD	0
	KW 00,04	INIT_S	0
	KW 00,05	INIT_VW	0
	KW 00,06	MODE_SST	101
	KW 00,10	SLV_SEND	0
	KW 00,11	SLV_REC	0
	KW 01,00	0	0
	KW 01,01	1	1
	KD 00,00	ZYKL_ZEIT	10
	KD 00,06	25	25
	# +00001	1	1
	# +00004	#_4	#_4
	# +00127	#H 7F	127
	#H C800		
	MRK 199	MRK 199	Label 199
	B 000		
	B 000	BA-Nr	Block No. (pre-assigned 0)
	MD 00,00		

7.2.2 Creating an ASCII file of the program (IL)

Section 7.4 "Reworking programs" describes the peculiarities to be observed when importing a program. Please read this section before starting the import. To facilitate reworking, the program should be prepared as follows:

- As only comment lines are added into the instruction list which start with a semicolon, all comment lines to be imported must start with a semicolon.
- To quickly find again segment plans, a comment line with the segment plan number should be added at the beginning of each segment plan.
- The connection elements of the 907 PC 331 (I+, I-, RS, SR, =1) should also be marked by comment lines, if they should be replaced by blocks of the 907 AC 1131 when reworking the program.
- If you have used blocks which have been converted to several types, you have to verify whether the constants K 00,00 or K 00,01 are assigned to the block input for operand type selection. Otherwise it is possible that the wrong block is used.

907 PC 331	907 AC 1131
SHIFT	SHIFTW, SHIFTD
FIFO	FIFOB, FIFOW
LIFO	LIFOB, LIFOW
INITS	INITSB, INITSW, INITSD
INITV	INITVB16, INITVB32, INITVW16, INITVW32, INITVD16, INITVD32

The program ASCII file is generated by printing the translated IL from the 907 PC 331 into a file. To receive standardized files the following print parameters have to be set:

Menu: Print param./Print parameter

• Page length:	100
• Left margin:	0
• Papersave:	Y
• Convers. file:	(none)
• Initial. file:	(none)
• Re-init file:	(none)

Menu: Print param./Print format editor

• No. of header lines:	0
• No. of footer lines:	0

Menu: Print 1/IL

• Start TP No.:	(none)
• End TP No.:	(none)
• Start TP name:	(none)
• End TP name:	(none)
• Print with cover sheet (Y/N):	N
• First page:	1
• Print destination:	projectname.AWL

☞ **Note:**

If the program has been created in FBD/LD, only comment lines that start with a semicolon are imported to IL.

The ASCII file is represented as follows: (Example for program start)

```
 Set    Word    Command Operand          Symbol      Long text


                ; ARCNET - CE's
00000 00000  !BA         000
      00001  APOLL
                ; Initialization of the ARCNET coupler  NODE=7


00001 00002  !       M 255,15
      00004  =N      M   238,01


00002 00006  !BA         000
      00007  AINIT
      00008          M   238,01
      00009          KW  30,04        TIMOUT      100
      00010          M   017,02
      00011          M   017,00
      00012          MW  030,02
      00013          MW  030,03
      00014          MW  030,00
      00015          M   017,03
      00016          MW  030,05
      00017          MW  030,04
      00018          MW  030,01
      00019          M   017,04
                ; Receiver block ARCNET
```

7.3 Importing the programs to 907 AC 1131

7.3.1 Sequence of steps to import the program to 907 AC 1131

Importing a project into 907 AC 1131 is performed in two phases:

1. Reading and editing the project using AC1131_V431.EXE

2. Further processing of the project using AC1131.EXE (V5.0 onwards)

The project to be imported is loaded into an "empty" 907 AC 1131 project. The "empty" project must contain the following:

- One POU PLC_PRG with at least one "dummy statement" (e.g. in IL LD 0, in FBD TRUE).

- The library management must contain the libraries "Standard.lib" (or IEC_S90_V41.LIB) and "ABB-BIB4.lib".

- The controller configuration must contain an entry of a series 90 basic unit (e.g. 07KT97).

- Under "Resources/Global Variables" the object "Schrittketten" has to be included (using "Project"/"Import"->"Step_var.exp").

The provided project file "**ABB_LEER.PRO**" already contains these entries.

Caution:
The file **Step_Var.EXP** must be located in the library directory of the 907 AC 1131 software, e.g. ..\AC1131\Library.

Sequence of steps to import the project:

To import a project into the 907 AC 1131 perform the following steps:

- Copy the files "projectname.VAR" and "projectname.AWL" into your project directory.

- Execute **AC1131_V431.EXE**, e.g. by double clicking the file.

- Open the provided project "ABB_LEER.PRO".

- Save the project under the desired project name.

- Translate the project using "Project/Rebuild All" or <F11>.

- Import the variables list "projectname.VAR" to the project. For that purpose, select the menu item "Online/Upload and import PC331 projects/Import variables from 907PC33 projects into global variables list". A window appears where you can select the *.VAR files. Select your variables file. The variables list is loaded. In the folder "Resources/Global Variables" one or several new entries (according to the size of the variables file) such as "GlobalVariables3" ... "GlobalVariablesYY" and the entry "Schrittketten" (if not already available) are created.

- Translate the project using "Project/Rebuild All" or <F11>.

- If necessary, correct the reported errors and translate the project again.

- Import the program file "projectname.AWL" into the project. To do this, call the menu "Online/Upload and import PC331 projects/Import ABB IL from 907PC33 projects". A window appears to select the *.AWL files. Select your program file. For loading into the language IL confirm with "OK". The program file is loaded. A new block of the type program (PRG) named with the "projectname" of your IL file (projectname.AWL) is created in the "POU" folder.

⚠️ **Caution:**

Loading the program file can take several minutes.

- If necessary, correct the reported errors and translate the project again.

- In the PLC_PRG insert a block call for your block or delete PLC_PRG and rename your block to PLC_PRG.

- After successful translation, the block can be converted to Function Block Diagram (FBD). Please verify your project after conversion. Pay special attention to program parts with set - S - and reset - R - instructions.

- Translate the project using "Project/Rebuild All" or <F11>.

- Check your project for non-importable functionality (refer to chapter 7.4 "Reworking programs") and redesign them.

- Set the cycle time for the program using the task configuration.

- Check the system constants.

- Save the project and exit AC1131_V431.EXE.

- Execute AC1131.EXE (V5.0 onwards)

- Open the project and save it under a different name (e.g. supplement name_V50.pro). If you save the project with and EXE file of version V5.x, you can't open it any more with AC1131_V431.EXE.

☞ **Note:**

Of course, reworking the project (e.g. redesign of non-importable functionality) can also be done using the V5.x-EXE.

7.3.2 Importing the variables to 907 AC 1131

The variables from the variable file "projectname.VAR" are imported as global variables to the addressable flag area "%Mxxx" of the 907 AC 1131 software.

Variable name

The unique identification of a variable is only possible via the absolute variable name (refer to the example of the variables list) because the symbolic name:

- either does not have to be available,
- can include special characters (symbols which are not conform to the IEC1131-3),
- or can be assigned twice.

Therefore the variable names are generated from the absolute variable names during the import. If a symbolic name exists, it is attached to the absolute name. Special characters are replaced by "_".

The long text of a variable is added as comment to the variables list.

For the constants word KW xx,yy and double word KD xx,yy the value appears in the long text and is entered as initialization value during the import.

The variable K 00,00 (boolean constant 0) is not included in the variables list and replaced by FALSE when importing the program. K 00,01 (boolean constant 1) is replaced by TRUE accordingly.

The direct constants #, #H are not included in the variables list and directly entered as constants when importing the program. As the IEC1131-3 does not allow HEX constants with the upper bit set as INT, these numbers are displayed as negative decimal values.

Examples:

- # +00001 → 1
- #H 5F → 16#5F
- #H FFFF → -1

The block number B 000 is not imported to the variables list, because it is not required for 907 AC 1131 blocks.

Examples for variables

```
907 PC 331:     EW 06,00        AnaIn1          Analog input channel 1
907 AC 1131: EW006_00_AnaIn1 AT %IW1006.0 : INT; (*Analog input channel 1 *)


907 PC 331:     EW 06,01        AnaIn2          Analog input channel 1
907 AC 1131: EW006_01_AnaIn2 AT %IW1006.1 : INT; (*Analog input channel 2 *)


907 PC 331:     KW 00,00        MastSlv          -1
907 AC 1131: KW00_00_MastSlv AT %MW3000.0 : INT := -1;


907 PC 331:     KW 01,00        Time Out         100
907 AC 1131: KW01_00_Time_Out AT %MW3001.0 : INT := 100;
```

7.3.3 Importing the programs to 907 AC 1131

After the variables have been imported into 907 AC 1131 and translated successfully after correction (if necessary), the program IL can be loaded. The necessary steps for this procedure are described in section 7.3.1 "Sequence of steps to import the program to 907 AC 1131".

During the import the ABB operators are converted into IEC operators. The ABB blocks are converted to the corresponding 907 AC 1131 ABB blocks. The corresponding operators / blocks are described in the block library documentation. Chapter 7.4 "Reworking programs" in this documentation describes all the cases for which a program rework is necessary.

For blocks with several variants (because of duplication) the optimum block for the given number of duplications is selected.

Example:

- 907 PC 331:   FKG with 6 sample points
- 907 AC 1131:   FKG8 and entry n:=6

For the blocks which have been converted into function blocks, instances are created automatically as local variable (for every block call).

For the timer blocks, a type conversion DINT_TO_TIME is inserted automatically before the time value input.

As intermediate flag, the local variable d_u_mm_y : BOOL; is inserted.

In the operating system of the controllers 07 KR 91, 07 KT 92, 07 KT 93 and 07 KT 94, the rules of the boolean algebra (AND before OR and MULTIPLICATION/DIVISION before ADDITION/SUBTRACTION) are observed automatically. In order to keep this functionality also with the 907 AC 1131, the corresponding parentheses are set while importing the project.

7.3.4 Importing the step chains

In 907 AC 1131 (as well as in the IEC 1131-3) no operands of the type step chain are available. To import projects created with 907 PC 331, for the step chains S000,00-S255,15 the range %MW5000-%MW5255 has been reserved in the addressable flag range %M.

While loading the variables list, the steps are correspondingly entered into the global variables.

While loading the program, the file Step_Var.EXP is loaded to the folder "Global Variables/Schrittketten". It includes one definition of the step chain number for each step chain.

Chain0 AT %MW5000 : INT;
...
Chain255 AT %MW5255 : INT;

The steps 0...15 are defined as Step00...Step15 as follows:

Step00 : INT:=1;
Step01 : INT:=2;
Step02 : INT:=4;
...
Step15 : INT:=-32768;

When importing the program the steps are called as follows:

907 PC 331	907 AC 1131
S000,00	Chain0 Step00
S000,01	Chain0 Step01
S255,00	Chain255 Step00
S255,15	Chain255 Step15

Example for a step combination:

907 PC 331:	907 AC 1131 (import)	
! S127,06	LD	Chain127
& M080,06	EQ	Step06
= S127,07	AND	M080_06
=S M250,00	ST	d_u_mm_y
	LD	d_u_mm_y
	SEL	Chain127,Step07
	ST	Chain127
	LD	d_u_mm_y
	S	M250_00

7.4   Reworking programs

After the program has been loaded, it has to be translated using "Online/Rebuild All" or <F11>. If no errors are reported, the blocks could be imported successfully. If any errors are reported, they can be displayed one after the other by pressing <F4>.

In the most cases, errors result from impossible type conversions which must be reprogrammed.

Example: Cannot convert DINT to INT.

Not imported program parts

Now check the program for parts which were not imported. For this, select the menu "Project/Global Search". Select your block "projectname [PRG]" and search for:

ERROR!!! Import not possible

The cursor then jumps to the first program part which was not imported. The program code is included as comment with the line number of the projectname.AWL file in the new program.

Program parts which can not be imported are for instance all blocks which were not converted into the 907 AC 1131 block library. These are blocks such as APOLL, ABORT, PE which are not required any more.

The functionality of those blocks which are not converted (e.g. ADAPT, FEHSU, SFEHSU, SINIT, DRUCK, EMAS) must be programmed again.

Accessing direct addresses:

If you have used blocks with direct memory access in the program, these program parts must also be edited. Either the functionality must be reworked or the new 32 bit addresses must be entered in any case. This can be done by placing the address operator ADR before the address input. This particularly concerns the following blocks:

- WOS   Write word to direct memory address
- DWOS   Write double word to direct memory address
- WOL   Read word from direct memory address
- DWOL   Read double word from direct memory address
- COPY   Copying memory areas
- AWM   Selection multiplexer
- USM   Switchover MUX

Accessing the address of boolean variables:

If you used blocks in the program which access addresses of boolean variables, this program parts must also be edited. The boolean variable input of the following blocks has to be prefixed by the block ADR as described in the block library documentation. This particularly concerns the following blocks:

- IDLB   Indirect binary reading
- IDSB   Indirect binary writing
- INITSB   Initialization of operand memory areas
- INITVB   Initialization of binary variables with values
- FIFOB   Stack memory (first in/first out) for binary data
- LIFOB   Stack memory (last in/last out) for binary data

In most cases the address operator ADR is inserted automatically. Verify for completeness.

Blocks with several types:

The following 907 PC 331 blocks with different types at the inputs and outputs were converted to multiple blocks.

907 PC 331	907 AC 1131
SHIFT	SHIFTW, SHIFTD
FIFO	FIFOB, FIFOW
LIFO	LIFOB, LIFOW
INITS	INITSB, INITSW, INITSD
INITV	INITVB16, INITVB32, INITVW16, INITVW32, INITVD16, INITVD32
COUNTB	COUNTBW, COUNTBD

SHIFT block:	SHIFT block for word/double word
	Input D/W of type E, A; M, K, S := 0 → word / := 1 → double word
Conversion:	D/W := K0,0, E, A, M, S → SHIFTW
	D/W := K0,1 → SHIFTD
Rework:	Check the block type
FIFO block:	Stack memory (first in/first out) for binary or word data
	Input B/W of type E, A; M, K, S := 0 → word / := 1 → binary
Conversion:	B/W := K0,0, E, A, M, S → FIFOW (do not prefix by ADR)
	B/W := K0,1 → FIFOB (input ANF must be prefixed by ADR)
Rework:	Check the block type, ADR
LIFO block:	Stack memory (last in/last out) for binary or word data
	Input B/W of type E, A; M, K, S := 0 → word / := 1 → binary
Conversion:	B/W := K0,0, E, A, M, S → LIFOW (do not prefix with ADR)
	B/W := K0,1 → LIFOB (input ANF must be prefixed by ADR)
Rework:	Check the block type, ADR

INITS block:	Word wise initialization of operand memory areas	
	Input VAR of type	E, A, M, K, S → binary, number of words (2 bits each)
		EW, AW, MW, KW → word, number of words
		MD, KD → double word, number of words (*2)
Conversion:	E, A, M, K, S → INITSB (prefix input VAR by ADR, number of bits/BOOLS)	
	EW, AW, MW, KW → INITSW (do not prefix by ADR, number in words)	
	MD, KD → INITSD (do not prefix by ADR, number in double words)	
Rework:	Check block type, ADR, adapt number	

INITV block:	Initialization of binary and word variables with numerical values	
	Input #W/B := 0 → word variable / := 1 → binary variable	
	Input #n – number of duplications	
Conversion:	#W/B := 0 and #n < 17 → INITVB16 (ADR before V0...V15)	
	#W/B := 0 and 16 < #n < 32 → INITVB32 (ADR before V0...V31)	
	#W/B := 1 and #n < 17 → INITVW16 (do not prefix by ADR)	
	#W/B := 1 and 16 < #n < 32 → NITVW32 (do not prefix by ADR)	
Rework:	Check the block type, ADR	

COUNTB block:	Testing number of bits in word/double word	
	Input W/DW of type E, M, K, S := 0 → word / := 1 → double word	
Conversion:	D/W := K0,0, E, A, M, S → COUNTBW	
	D/W := K0,1 → COUNTBD	
Rework:	Check the block type	

Word conversion blocks:

The blocks for packing or unpacking two words into one double word and vice versa (in EBS operating system blocks **WWDW**, **DWWW**) are loaded to the blocks **WSWDW** and **DW2W**. In contrast to WWDW, for the block W2WDW the first word is stored in the high part of the double word and the second word is stored in the low part of the double word. Correspondingly for block DW2W, the low part of the double word is unpacked to the second word and the high part of the double word is unpacked to the first word.

In order to obtain correct results again, the first and the second word at these blocks must be changed. This is not performed automatically.

ARCNET blocks:

The block **APOLL** is not required any more.

For every operation at the block **AREC** a block AREC is generated. The operand at the 907 PC 331 AREC block output UJOB is entered to all UJOB outputs of the new AREC blocks. When reworking, another variable of the type BOOL has to be assigned to the output UJOB (e.g. AREC_0_UJOB ... AREC_n_UJOB). These variables must be connected with an OR operation after the last AREC block and assigned to the original variable UJOB.

The blocks **ASEND**, **ASEND+** are transferred into the corresponding blocks ASEND1, ASEND4 and ASEND16.

CS31 system bus blocks:

The blocks **CS31QU**, **CS31CO** and **CS31TE** for the CS31 system bus are converted into the corresponding blocks.

For the block **CS31TE** you have to observe that one block CS31TE is generated for each module (parameter #n in 907 PC 331). The block header (parameter 0/1, TIME, RDY, OK, ERNO) is not converted. The outputs at the 907 PC 331 blocks MO and ME are assigned to the outputs OK (block found, connected to bus) and AER (error module address, address not found on bus). The output DER (error data transfer range) must be allocated a new variable (e.g. CS31TE_n_DER : BOOL;). For an evaluation the combination of the output bits must be observed. Explanation:

AER	DER	OK	Meaning
FALSE	FALSE	TRUE	Module connected to bus
TRUE	FALSE	FALSE	Module address not found
FALSE	TRUE	FALSE	Module address ok, data range incorrect

Other combinations are not possible.

RCOM/RCOM+ blocks:

In 907 PC 331, the RCOM/RCOM+ blocks were used as connection elements (CE). This must be observed for programs which are using the remote protocol RCOM/RCOM+. Correspondingly the CEs are resolved in the IL. The program code of the RCOM/RCOM+ CEs must be replaced with the new blocks from the RCOM library.

MODBUS blocks:

907 AC 1131 provides uniform blocks (MODINIT, MODMAST) for the communication via the serial interfaces COM1 and COM2 or the external MODBUS coupler 07 KP 93. For the basic unit 07 KT 94, the setting of the MODBUS slave address is done using the system constant KW 00,06 / %MW3000.6. For the 07 KT 97, the system constant KW 00,06 / %MW3000.6 is not longer significant. The setting of the interface is done using the MODINIT block.

The MODBUS master functionality must be programmed with the MODMAST block.

Connection elements in 907 PC 331:

In the 907 PC 331 software, the following blocks are implemented as connection elements (CE):

- **907 PC 331   907 AC 1131**
- I+   R_TRIG   Positive edge
- I-   F_TRIG   Negative edge
- SR   RS   Reset memory, dominating
- RS   SR   SET memory , dominating
- =1   XOR   Exclusive OR for binary values

The CEs are resolved in the IL. In order to use the 907 AC 1131 blocks, the imported IL must be reworked accordingly. Reworking is not absolutely necessary since the logical connection is performed correctly.

Positive edge: (I+ → R_TRIG)

907 PC 331/IL:			Import		Rework	
!N	M 238,02	0_1	LDN	M238_02_0_1	**Declaration:**	
=R	M 000,00	Q	R	M000_00_Q	R_TRIG1 : R_TRIG;	
!	M 238,02	0_1	LD	M238_02_0_1		
&N	M 000,00	Q	ANDN	M000_00_Q	**Instructions:**	
=S	M 000,00	Q	S	M000_00_Q	CAL	R_TRIG1(CLK := M238_02_0_1)
=	M 020,00	PULS	LD	M238_02_0_1	LD	R_TRIG1.Q
			ANDN	M000_00_Q	ST	M020_00_PULS
			ST	M020_00_PULS		

Negative edge: (I- → F_TRIG)

907 PC 331/IL:			Import		Rework	
!	M 238,02	1_0	LD	M238_02_1_0	**Declaration:**	
=S	M 000,00	Q	S	M000_00_Q	F_TRIG1 : F_TRIG;	
!N	M 238,02	1_0	LDN	M238_02_1_0		
&	M 000,00	Q	AND	M000_00_Q	**Instructions:**	
=R	M 000,00	Q	R	M000_00_Q	CAL	F_TRIG1(CLK := M238_02_1_0)
=	M 020,00	PULS	LDN	M238_02_0_1	LD	F_TRIG1.Q
			AND	M000_00_Q	ST	M020_00_PULS
			ST	M020_00_PULS		

Reset memory, dominating: (SR → RS)

907 PC 331/IL:			Import		Rework	
!	M 238,02	S	LD	M238_02_S	**Declaration:**	
=S	M 000,00	Q1	S	M000_00_Q1	RSInst1 : RS;	
!	M 238,00	R	LD	M238_00_R		
=R	M 000,00	Q1	R	M000_00_Q1	**Instructions:**	
!	M 000,00	Q1	LD	M000_00_Q1	CAL	RSInst1(SET := M238_02_S,
=	M 020,00	Q	ST	M020_00_Q		RESET1 := M238_00_R);
					LD	RSInst1.Q1
					ST	M020_00_Q

Set memory, dominating: (RS → SR)

907 PC 331/IL:		Import		Rework
!	M 238,02	R	LD	M238_02_R	**Declaration:**
=R	M 000,00	Q1	R	M000_00_Q1	SRInst1 : SR;
!	M 238,00	S	LD	M238_00_S	
=S	M 000,00	Q1	S	M000_00_Q1	**Instructions:**
!	M 000,00	Q1	LD	M000_00_Q1	CAL SRInst1(SET := M238_00_S,
=	M 020,00	Q	ST	M020_00_Q	RESET1 := M238_02_R);

LD   SRInst1.Q1
ST   M020_00_Q

Binary exclusive OR: (=1→ XOR)

907 PC 331/IL:		Import		Rework
!	M 010,00	E1	LD	M010_00_E1	LD M010_00_E1
&N	M 010,01	E2	ANDN	M010_01_E2	XOR M010_00_E2
/N	M 010,00	E1	ORN	M010_00_E1	ST M011_00_A1
&	M 010,01	E2	AND	M010_01_E2	
=	M 011,00	A1	ST	M011_00_A1	

User connection elements:

User connection elements are resolved in the IL and correspondingly imported as IL.

8 Figures

9   Index

ABB