SIEMENS

SIMATIC HMI

WinCC V8.0
ProDiag - Plant monitoring in WinCC

System Manual

08/2023
A5E52671436-AB

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

Document order number: Printout of the Online Help
Ⓟ 08/2023 Subject to change

Table of contents

Basics of supervision with ProDiag 1

Introduction

ProDiag is used to monitor and identify errors that occur in the plant or machine. You can use ProDiag to show the type of error, the cause of the error and the location of the error on the HMI device.

Use

You use ProDiag functions to monitor your plant and visualize it in Runtime. The main objective of ProDiag is to reduce downtimes and production failures after an error occurs and to prevent errors through early warnings. Diagnostics and display objects supply targeted information for the operator during troubleshooting.

Principle

In STEP 7, you create operand supervisions and configure the settings according to your requirements. When an error occurs, a supervision alarm is generated based on the criteria you have configured. The configured supervision instances are stored in the preset ProDiag function block. You can use the automatically generated ProDiag FBs or create and configure your own ProDiag FBs.

With WinCC, you visualize the ProDiag functions for display on an HMI device. You configure the screen objects which provide an overview of the current status of the plant and display the affected code or the step sequence when an error occurs.

Controls for monitoring with ProDiag 2

A distinction is made between two types of diagnostics:

- System diagnostics (< 5% of all error cases)
  Detection, signaling and evaluation of errors within the automation system (e.g. wire break, module failure, program error, etc.)
- Process diagnostics (> 95% of all error cases)
  The purpose of process diagnostics is to gather information on the type, location and cause of an error and to provide information on troubleshooting (e.g. motion disturbed, interlock not fulfilled, pressure too high, end position exited impermissibly, etc.).

Main goal of system diagnostics and process diagnostics:

- Reduction of downtimes and production outages after an error occurs, or prevention through early warnings and notices.
- Facilitation of troubleshooting or maintenance work through targeted instructions for the machine operator.

You can find a detailed description of the configuration of monitoring in the STEP 7 documentation.

You can find more information in the "SIMATIC STEP 7 Basic/Professional V18 and SIMATIC WinCC V18" documentation:

- SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Monitoring machines and systems with ProDiag" (https://support.industry.siemens.com/cs/ww/en/view/109815056/159080042635)

ProDiag Controls

The Graphics Designer offers the following ProDiag controls when creating pictures:

ProDiag overview (WinCC ProDiagOverviewControl)	Overview of the monitoring at Runtime and gathering of diagnostic information if an error occurs.
ProDiag overview (WinCC ProDiagOverviewControl)	Shows the current program status for executed steps of the GRAPH sequencer. Errors during execution of a program are shown directly in the corresponding step.
PLC code view (WinCC PlcCodeViewerControl)	Shows the GRAPH sequence with "Status" of the steps and the assignment list of the utilized operands.
PLC code view (WinCC PlcCodeViewerControl)	Shows the operands in the user program that contain errors and have triggered a selected alarm. In addition to the alarm, the list of operands with errors is shown in the same picture.

You can find the ProDiag Controls in the "Controls" palette > "Controls" selection window > "Prodiag Controls" node.

General procedure in runtime

1. The ProDiag overview (WinCC ProDiagOverviewControl) and GRAPH overview (WinCC GraphOverviewControl) controls call attention to errors in the sequence.
2. Click the alarm button to jump to the message in the alarm view.
3. To see details on an error, display the PLC code view control (WinCC PlcCodeViewerControl).
4. The Criteria analysis control (WinCC CriteriaAnalysisControl) allows you to analyze errors in the PLC user program.

See also

SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Monitoring machines and systems with ProDiag" (https://support.industry.siemens.com/cs/ww/en/view/109815056/159080042635)

3 Configuring supervisions

Introduction

In WinCC, you configure ProDiag supervisions in your user program in STEP 7 and the visualization for the HMI devices in WinCC.

Requirements

- A connection to an S7-1500 controller has been configured in Tag Management of WinCC.
  The configuration of a STEP 7 project is imported using a TIA export file in zip format.
  - More information:
    "Communication > SIMATIC S7-1200, S7-1500 Channel > Configuring the channel > How to download AS symbols offline"
- There is at least one Boolean tag, either in a data block, the tag table, or a block interface.
- A WinCC project is open.
- A picture has been created in Graphics Designer.

Procedure for configuration of supervisions with ProDiag

In STEP 7

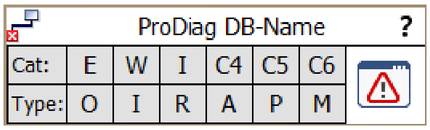
1. Create ProDiag function block for the supervisions.
2. Define settings on the ProDiag function block.
3. Define ProDiag supervision settings:
   - Define supervision type.
   - Set the delay time.
   - Select the conditions under which a supervision alarm is triggered.
   - Define the categories and subcategories.
4. Add supervision for a Boolean tag.
   You create either global or a local supervision for a Boolean tag.
5. Call the ProDiag function block.
6. Compile and download the project.

In WinCC

1. Insert the "ProDiag overview" control ("ProDiagOverviewControl") into the picture.
2. Specify connection to the PLC and status tag.
3. Configure the display and the representation of the object.
4. Configure the jump to the message view.

In addition, you can have the program flow displayed in the control "GRAPH overview" ("GraphOverviewControl") and the current program status in the control "PLC Code Viewer" ("PlcCodeViewer").

Assignment of icons in the ProDiag overview



The "State" status tag of PLC data type "SV_FB_State" of the ProDiag function block contains the group error bits for the supervision types and categories that you visualize using the ProDiag overview.

You can visualize two types of group error bits in the "ProDiag overview" object:

- Group error bit for supervisions (All, O, I, R, A, P, Merr, Mtxt)
- Group error bit for the categories (C1 ... C6)

The following table shows the assignment of the group error bits of the "State" status tag to the icons in the ProDiag overview.

ProDiag overview	Elements in ProDiag FB	Description
Icons		
⚠	All	Group error bit for all supervision types, excluding message text If value of All = TRUE, at least one supervision was triggered and the icon is visually highlighted in the ProDiag overview.
ℹ	SV_Types	Byte for all group error bits of the supervision types
	Mtxt (message text)	Group error bit for all alarms
?	-	Tooltip with information on this ProDiag overview
Categories		
E (Error)	C1 (Category)	Error
W (Warning)	C2 (Category)	Warning
	Categories	Group error byte for all categories.
I (Info)	C3 (Category)	Information
C4 ... C6	C4 ... C6 (Category)	Additional categories
Supervision types		
O	O (Operand)	Group error bit for operand supervisions
I	I (Interlock)	Group error bit for interlock supervisions
A	A (Action)	Group error bit for action supervisions
R	R (Reaction)	Group error bit for reaction supervisions
P	P (Position)	Group error bit for position supervisions
M	Merr (Message error)	Group error bit for error texts

Configuring the ProDiag overview 4

Introduction

The "ProDiag overview" ("ProDiagOverviewControl") is used to supervise a machine or system during Runtime and to gather diagnostic information for occurring errors.

Once you have set the status tag in the object and the connection to a ProDiag FB has been established, the status of the "State" status tag of the corresponding PLC data type is queried. In Runtime, the states of the monitored operands are represented as icons in the ProDiag overview, similar to the signal colors of a traffic light.

You configure the display and representation of the categories and supervision types that are shown in the "ProDiag overview" control in WinCC, independent of the supervision settings in STEP 7.

Requirements

- At least one S7-1500 controller has been created.
- At least one supervision instance has been configured.
- A ProDiag function block and ProDiag data block exist.
- A PC station or an HMI device that supports the ProDiag functionality has been created.
- A connection between the PLC and HMI device has been configured.
- A picture has been created and opened in Graphics Designer.
- A "ProDiag Overview" control has been inserted into the picture.

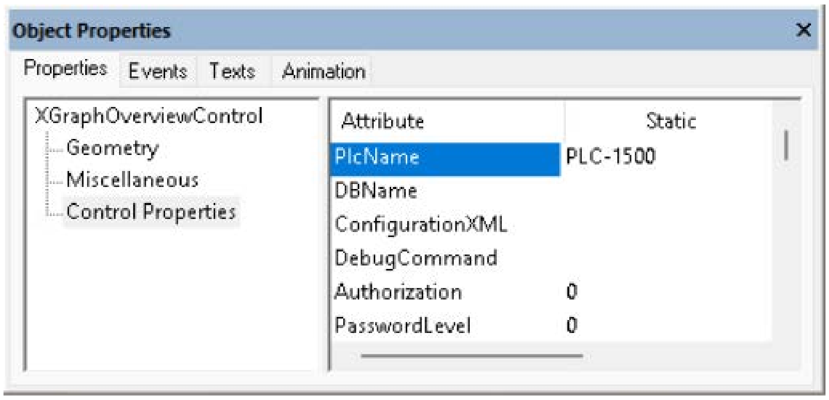
Procedure in Tag Management

1. Select the connection in the navigation area of the Tag Management.
2. Enter the name of the PLC under "Properties > Connection > Miscellaneous > PLC name".

Procedure in Graphics Designer

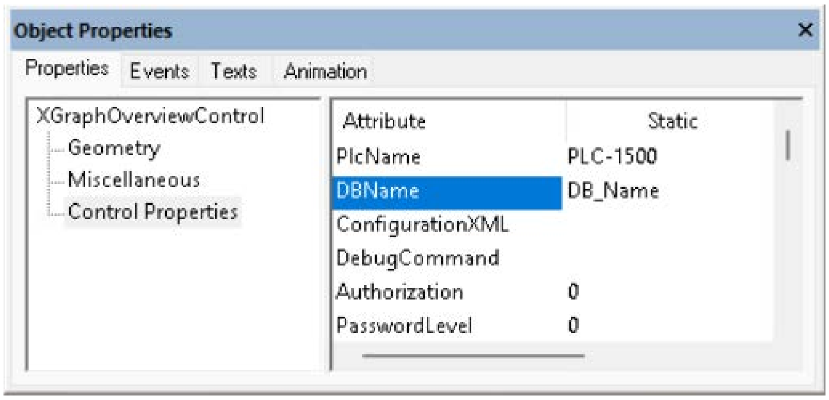
1. Select the "ProDiag overview" control.
2. In the "Object Properties > Properties" window, click "Control Properties".

3. Enter the name of the PLC that you specified in Tag Management under "PLCName".



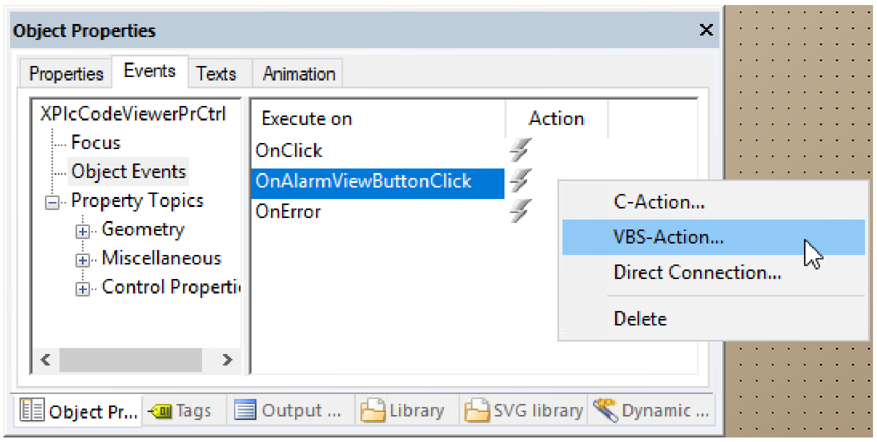
Attribute	Static
PlcName	PLC-1500
DBName	
ConfigurationXML	
DebugCommand	
Authorization	0
PasswordLevel	0

4. Enter the name of the ProDiag instance data block under "DBName".



Attribute	Static
PlcName	PLC-1500
DBName	DB_Name
ConfigurationXML	
DebugCommand	
Authorization	0
PasswordLevel	0

5. Define the names and colors for the supervision categories.
6. Define the names and colors for the supervision types.
7. Specify which symbols and buttons are displayed in the control in runtime.
8. In a multi-client system, enter the "Server prefix" under "ServerPrefixName".
9. You can configure a direct connection or a script at the "Alarm view button click" event (OnAlarmViewButtonClick) under "Object Properties > Events" to jump from the ProDiag overview to the alarm view in Runtime.



Result

The ProDiag overview is inserted in the screen. The current states of monitored events are displayed in Runtime.

See also

Configuring the display of criteria analysis (Page 41)

Configuring a GRAPH overview (Page 21)

5 Configuring jump to the alarm view

Introduction

After an error has occurred, you have the option in runtime to jump directly from the ProDiag overview or GRAPH overview to an alarm view or a picture with an alarm view.

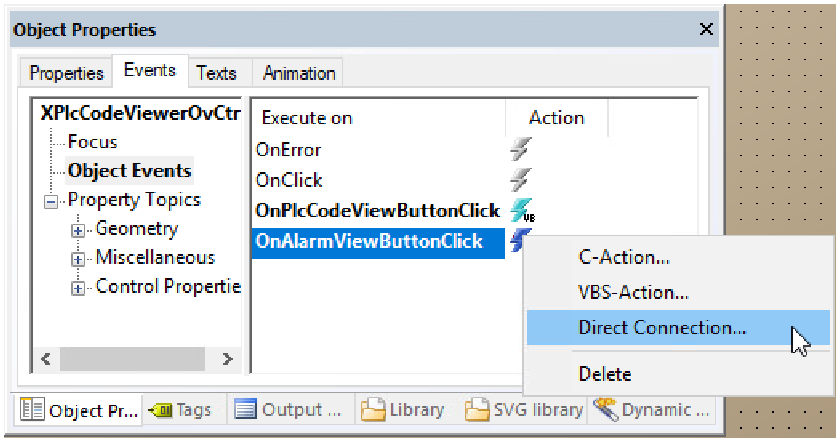
Typically, the alarm view is configured as "not visible" in error-free operation. The alarm view is displayed by calling it from a ProDiag control.

Requirement

- A picture has been created.
- An alarm view has been created.
- The "ProDiag overview" control and/or the "GRAPH overview" control has been configured.

Configuring the jump to the alarm view

1. Select the control.
2. In the "Object Properties" window, click on the "Events" tab.
3. Open the shortcut menu of the "Action" under "Object Events" > "OnAlarmViewButtonClick"



4. Display the picture window with the configured alarm view as follows:
   - Direct Connection: Specify the name of the picture window containing the configured alarm view.
   - VBS script: `ScreenItems("ctrlAlarm"), visible true`

Result

If you press the "Alarm view" button in the configured ProDiag overview in runtime, the picture window with the corresponding alarm view is displayed.

See also

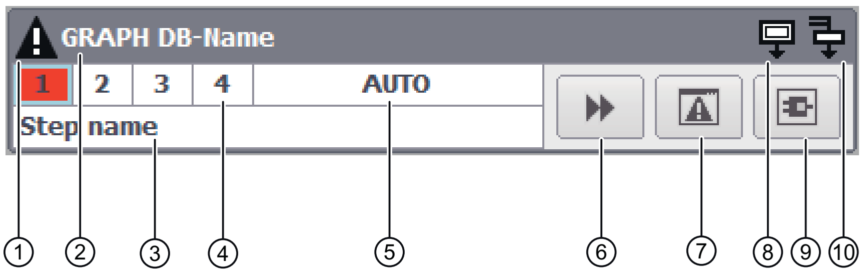
Configuring a GRAPH overview (Page 21)

Configuring the GRAPH overview 6

6.1 Displaying the status of the GRAPH sequencer

Introduction

The status of the GRAPH sequencer is visualized via the "GRAPH overview" control.



①	Error symbol
②	Name of GRAPH data block
③	Name of step that is currently being executed
④	Active step of sequencer If an error occurs in a step, the step number is highlighted in red.
⑤	Operating mode
⑥	"Step" button If the sequencer is in the last step, the "Initial step" button is displayed.
⑦	"Alarm view" button
⑧	Initial step symbol
⑨	"PLC code view" button
⑩	Simultaneous step symbol

Displaying the status of the GRAPH sequencer

You connect the GRAPH overview to the PLC via the GRAPH DB tag. After the connection to the control was established, the control shows the active step number of the GRAPH sequencer. The name of the active step is output in the sequencer. If there are additional simultaneous steps besides the currently executed step, the simultaneous step symbol lights up in the GRAPH overview.

Depending on the application, switch between the four operating modes of the sequencer. The currently selected mode is displayed in the GRAPH overview.

If an error occurs during execution of a step, the step in which the error occurs and the error symbol are highlighted visually.

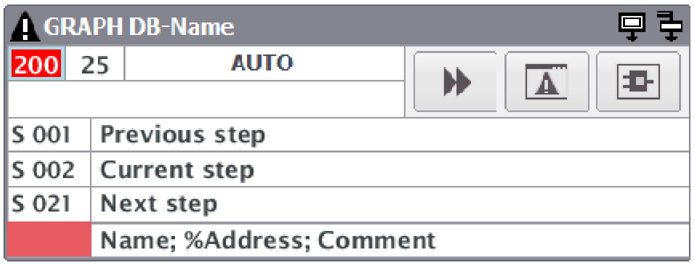
If an error occurs, you have the possibility to jump to the message in the configured alarm view or view the affected program in the PLC code view.

Single-line mode

You have the option of displaying the GRAPH overview in the standard view or in single-line mode. You can, for example, configure several GRAPH overviews in the single-line mode in a picture. This provides you with the overview of the current status of multiple production areas, whose supervisions are summarized in the corresponding GRAPH FBs.

Display of steps and criteria analysis

In extended mode, you activate the additional lines for display of the previous and next step as well as the first operand of the criteria analysis with errors.



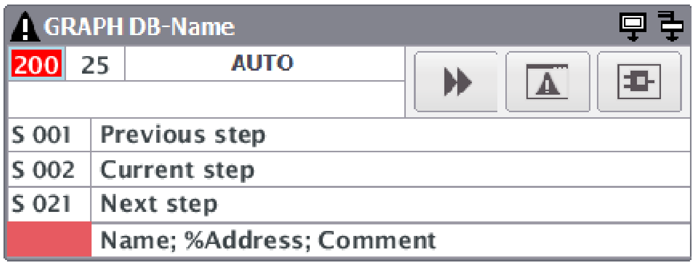
S 001	Previous step
S 002	Current step
S 021	Next step
🟥	Name; %Address; Comment

6.2 Show step history

Introduction

In the GRAPH overview you can have the previous and next step of the GRAPH sequencer displayed in addition to the current step. For this, the active steps are internally recorded after a screen with a GRAPH overview is called. If an error occurs in a step, you see the actions that were executed before and after the step with an error.

You specify the multilingual step names displayed in the GRAPH overview in a multilingual text list in the user program beforehand.



S 001	Previous step
S 002	Current step
S 021	Next step
🟥	Name; %Address; Comment

Marking of the steps

The GRAPH sequencer is evaluated from left to right. The step numbers in the GRAPH overview are additionally identified when there is a simultaneous or an alternative branch directly after or before this step. This enables you to see the following information about the program flow at a glance:

- Plus sign
  - Previous step: A simultaneous branch is present before the active step. Multiple steps were active simultaneously.
  - Next step: a simultaneous branch is present after the active step. More than one step will be activated.
- Question mark
  - Previous step: A step different than the first (left) step of the sequencer could have been active before the active step.
  - Next step: an alternative branch is present after the active step. This means that a step different than the first (left) step of the sequencer can also be activated.
- No character: The previous or next step is unambiguous.

Note

Limitations affecting step determination

During the initial call of the GRAPH overview the previous and next steps from the GRAPH block are determined in the sequence of the programmed sequencer.

When calling the GRAPH overview for the first time and after a picture change no record of the active steps exists and a unique determination of the previous step is thus not possible.

The GRAPH overview updates the view at regular intervals. If the sequencer runs rapidly or if you switch often manually between the steps, it is possible that a step other than the active step of the sequencer is determined.

6.3 Configuring a GRAPH overview

Introduction

You can use the GRAPH overview to view the current program status for the executed steps of a GRAPH sequencer.

Requirement

- A PLC with a GRAPH data block has been created.
- The GRAPH data block contains at least one tag that is visible on the HMI device and is accessible from the HMI device.

Note

Process tag on the HMI device

The process tag you use for the GRAPH overview must be visible on the HMI device and accessible from the HMI device.

To identify the tags of the GRAPH data block as visible and accessible for an HMI device, open the GRAPH function block, select the block in the work area and select "Edit > Internal parameters visible/accessible from HMI" in the menu bar.

Then compile the program blocks.

- An HMI device has been created.
- A picture has been created.
- The "Object Properties" window is displayed.

Procedure

1. Use drag-and-drop to move the "GRAPH overview" control (WinCC GraphOverviewControl) from the palette "Controls" > "Controls" selection window > "ProDiag Controls" node into the configured picture.
2. Select the "GRAPH overview" control.
3. In the "Object Properties" window, click on "Properties > Control Properties".
4. Enter the name of the connection under "PlcName".
5. Enter the name of the instance of the sequencer under "DBName".
   Example: `SeqData.autoMaschine`
6. Specify which icons are to be displayed in the GRAPH overview.
   To display the GRAPH overview in compatibility mode without toolbar buttons and operating mode display, enable the "Single-line mode" property (SingleLineMode).
7. Specify the buttons to be displayed in the object.
8. In a multi-client system, enter the "Server prefix" under ServerPrefixName.

9. You can dynamize the properties in the "GRAPH overview" control with tags and color-code states.
   More information:
   – "Working with WinCC > VBA for Automated Configuration > VBA in the Graphics Designer > Creating Dynamics with VBA"
10. In order to jump to the alarm view and PLC code view in Runtime, you can assign system functions to the buttons in the GRAPH overview under "Object Properties > Events".
   More information:
   – "Configuring jump to the alarm view (Page 15)"
   – "Configuring the jump to the PLC code view (Page 25)"

Displaying the result of the criteria analysis

You have the option of displaying the result of the criteria analysis in the GRAPH overview when an error is present.

The following information is then displayed in the last line of the GRAPH overview:

- Symbol name
- Absolute address
- Value of the first operand with errors
- Comment

To display the criteria analysis, select the "ShowPLCCodeViewerButton" option under "Object Properties > Control Properties".

Note

No information in the GRAPH overview

No information for the criteria analysis is output in the GRAPH overview in the following cases:

- The initial value acquisition was not activated in the block properties or the initial values were not acquired.
- There is no error in the active step.

Displaying the previous and next steps

To display the previous and next steps in the GRAPH overview, select the "ShowPreviousAndNextStep" option under "Object Properties > Control Properties".

Result

The GRAPH overview is inserted in the picture.

The current state of the GRAPH sequencer is displayed in Runtime.

See also

Configuring jump to the alarm view (Page 15)

Configuring the jump to the PLC code view (Page 25)

Configuring the display of criteria analysis (Page 41)

Configuring the ProDiag overview (Page 11)

6.4 Configuring the operating mode

Introduction

Four different operating modes are available for executing the GRAPH overview (WinCC GraphOverviewControl):

Auto (default setting)	Automatically switches to the next step when the transition is fulfilled.
TAP	Switches to the next step when the transition is fulfilled and there is an edge change from "0" to "1" at parameter T_PUSH.
TOP	Switches to the next step when the transition is fulfilled or there is an edge change from "0" to "1" at parameter T_PUSH.
Manual	The next step is not enabled automatically when the transition is fulfilled. The steps can be selected and deselected manually.

Note

You set the operating mode by modifying the interface parameters of the GRAPH block in your user program.

Note

The name of the operating mode is language dependent and can be translated for the required Runtime languages.

In runtime, you have the option of renaming the operating mode that is displayed in the GRAPH overview.

Configuring labels for the operating modes

A picture with the configured GRAPH overview is open.

1. In the "Object Properties" window, click on "Properties > Control Properties".
2. Configure the labels for the four operating modes:
   – OperationMode AUTO
   – OperationMode TOP
   – OperationMode TAP
   – OperationMode MAN

6.5 Configuring the jump to the PLC code view

Introduction

After an error has occurred, you have the option to jump directly from the GRAPH overview to the PLC code view.

Requirement

- A picture has been created.
- The "GRAPH overview" control has been configured.

Configuring the jump to the PLC code view

1. Select the "PLC code view" control.
2. In the "Object Properties" window, click on the "Events" tab.
3. Open the shortcut menu of the action under "Object Events" > "OnPlcCodeViewButtonClick"
4. Display the PLC code view using the function `ShowPLCCodeViewFromAlarm`:
   You can find the function in the "Global Script" editor under "C-Editor > Standard functions > ProDiag".
   Parameters:
   - `AlarmScreenName`: Name of the picture that contains the AlarmControl.
   - `AlarmViewName`: Name of the alarm view.
   - `BaseScreenName`: The current picture or the picture in which the "GRAPH overview" control is located.
   - `ScreenWindowName`: Picture window in which the `PLCCodeViewScreen` picture is displayed.
   - `PLCCodeViewScreenName`: Name of the picture with the PLC code view.
   - `PLCCodeViewName`: Name of the PLC code view.
   - `ErrorTag`: In the event of an error, the error tag is used to output information about the error that has occurred.

Example

```
void ShowPLCCodeFromAlarmInView(LPCTSTR AlarmScreenName, LPCTSTR AlarmViewName, LPCTSTR
BaseScreenName, LPCTSTR ScreenWindowName, LPCTSTR PLCCodeViewScreenName, LPCTSTR
PLCCodeViewName, LPCTSTR ErrorTag)
{
BOOL bret;
CMN_ERROR err;
char szError[1024]

bRet = FALSE;
```

```
memset(&err, 0, sizeof(err));

bRet = ShowPLCCodeFromAlarmInView(AlarmScreenName, AlarmViewName, BaseScreenName,
ScreenWindowName, PLCCodeViewScreenName, PLCCodeViewName, ErrorTag, &err);
if (FALSE == bRet)
{
   spritf(szError, "ShowPLCCodeFromAlarmInView(..) Failed:
0x%081x,0x%081x,0x%081x,0x%081x,0x%081x,[%s]",
   spritf(szError, "ShowPLCCodeFromAlarmInView(..): ERROR\z\n%ld\\z\n%ld\\z\n%ld\\z\n%ld\
\z\n%ld\\z\n%ld\,
      err.dwError1, err.dwError2, err.dwError3, err.dwError4, err.dwError5,
err.szErrorText);
   if NULL != ErrorTag)
   {
      SetTagChar(ErrorTag,szError);
   }
   spritf(szError, "ShowPLCCodeFromAlarmInView(..) failed:
0x%081x,0x%081x,0x%081x,0x%081x,0x%081x,[%s]",
      err.dwError1, err.dwError2, err.dwError3, err.dwError4, err.dwError5,
err.szErrorText);
   printf("%s\r\n",szError);
}
```

Result

If you press the "Alarm view" button in the configured ProDiag overview in runtime, the PLC code view is displayed.

See also

Configuring a GRAPH overview (Page 21)

Configuring the PLC code view 7

7.1 Configuring the PLC code view

Introduction

To display the PLC program networks in the LAD, FBD and GRAPH programming languages in runtime, insert a PLC code view in your project.

The PLC code view is displayed in Runtime by clicking the "PLC code view" button.

Requirement

- A connection between the PLC and HMI device has been created.
- A picture has been created.
- The control "ProDiag overview" (WinCC ProDiagOverviewControl) and/or the control "GRAPH overview"·(WinCC·GraphOverviewControl) has been created and configured in the picture.

Procedure

1. Use drag-and-drop to move the "PLC code view" control (WinCC PlcCodeViewerControl) from the palette "Controls" > "Controls" selection window > "ProDiag Controls" node into the configured picture.

   Note

   Best Practice

   To make it easy to show and hide the view in runtime, create the control in a picture window.

2. In the "Object Properties" window, click on "Properties > Control Properties".
3. Specify additional properties of the control here.
4. In a multi-client system, enter the "Server prefix" under ServerPrefixName.
5. To display or handle errors in the configuration of the display of a network in Runtime, configure a corresponding script under "Object Properties > Events > Object Events > OnError".

Result

The PLC code view is inserted in the screen. In Runtime, PLC user programs that are programmed in the LAD, FBD or GRAPH programming languages can be displayed.

The control is displayed in Runtime if it is activated when a fault has occurred by clicking the "PLC code view" button in the "ProDiag overview" control (WinCC ProDiagOverviewControl) and/or in the "GRAPH overview control" (WinCC·GraphOverviewControl). Alternatively, the control can be activated using a separate button in the picture.

7.2 Views of the PLC code view

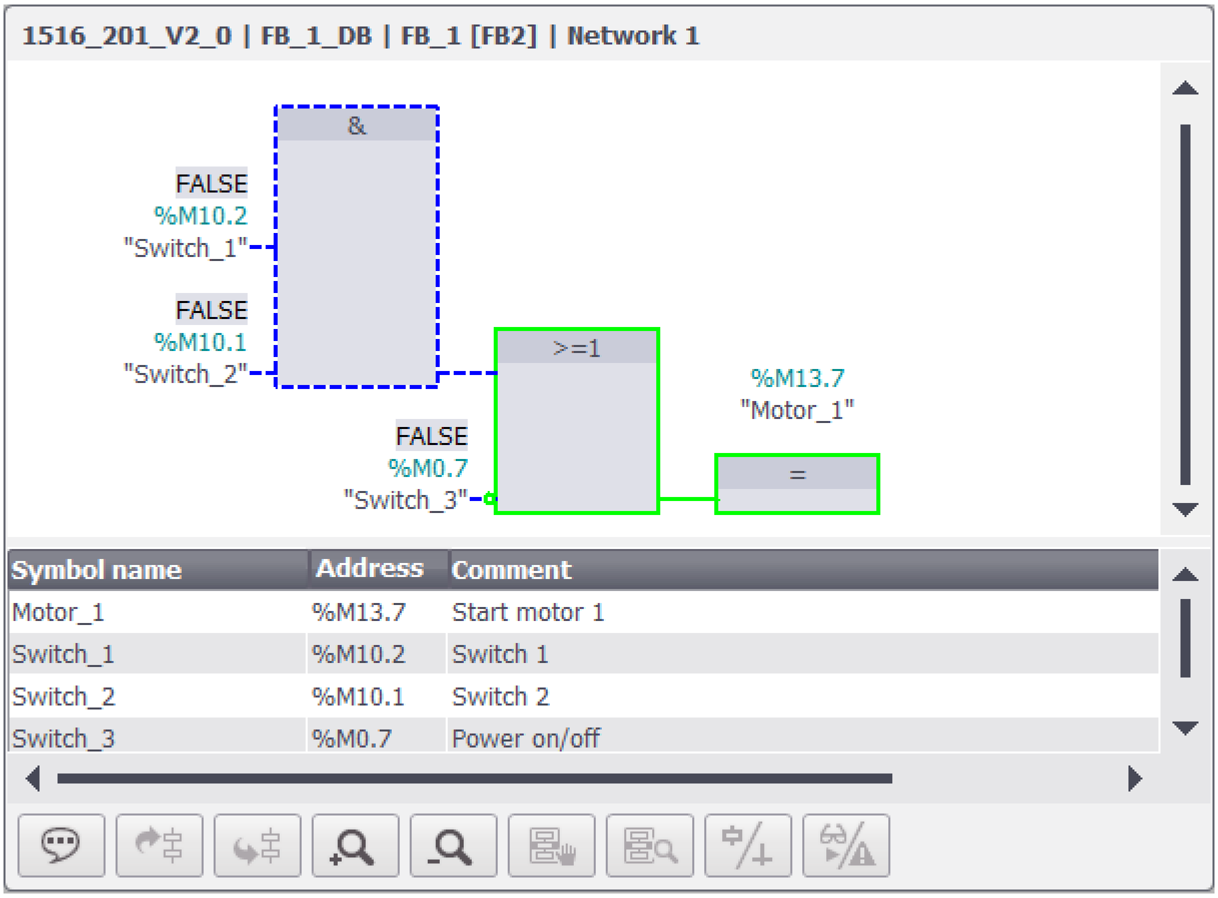
Introduction

In the PLC code view, you display various information about the user program.

- Information area
- Symbol table
- Detail view
- "Transition/Interlock view"
- Initial value/actual value view

Information area

The information area shows the LAD/FBD program code of your user program or the GRAPH sequencer of a GRAPH function block. After the connection to the PLC has been established, the network status is displayed in the information area highlighted via the program code.



Symbol name	Address	Comment
Motor_1	%M13.7	Start motor 1
Switch_1	%M10.2	Switch 1
Switch_2	%M10.1	Switch 2
Switch_3	%M0.7	Power on/off

Symbol table

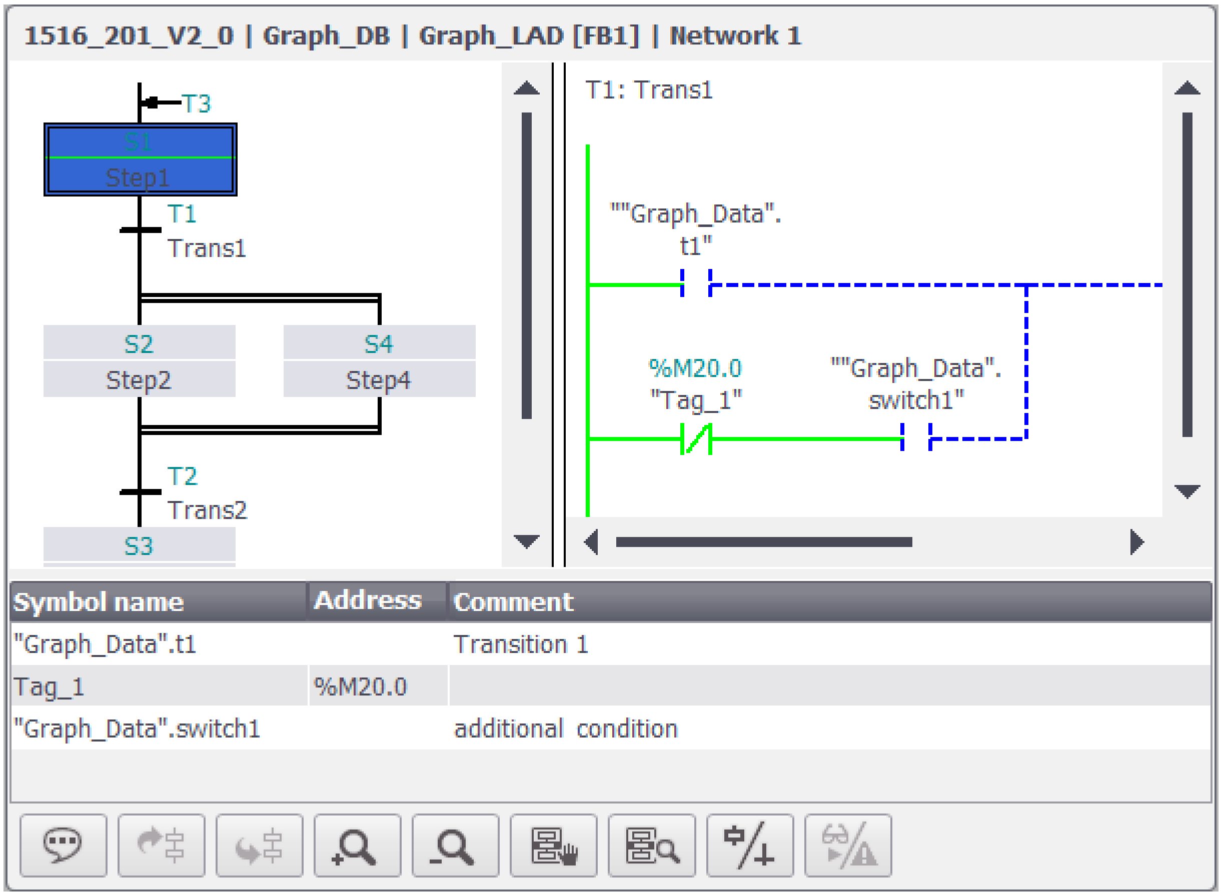
The symbol table shows the symbols which the displayed network uses. The three columns of the symbol table display the symbol names of the tags, the absolute address and the comments. As the comments are loaded from the user program, these are displayed in the language that is stored in the PLC.

You can dynamically change the height and width of the symbol table in Runtime.

Detail view

The detail view is also available for displaying your GRAPH sequencer.

The detail view provides detailed information about the selected network and the pending errors. In the detail view you see, for example, the transition of the first active step of the GRAPH sequencer. The detail view is opened in Runtime with the "Details" button.

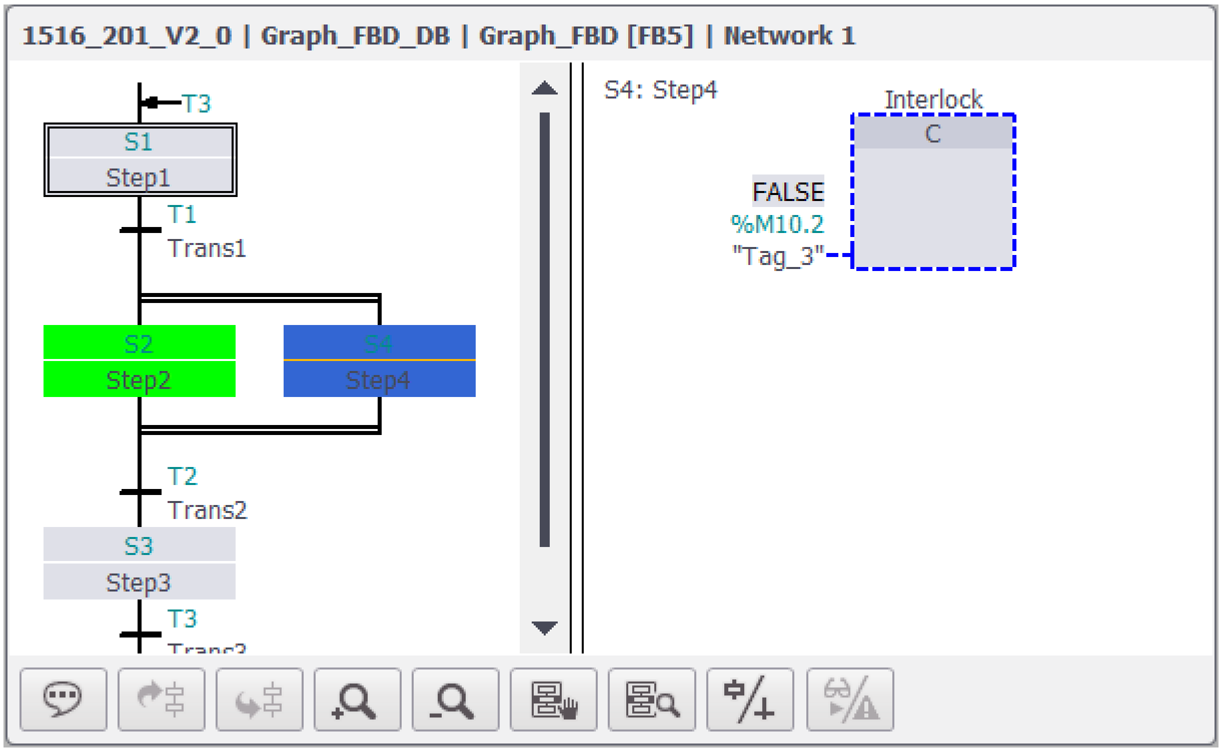


Symbol name	Address	Comment
"Graph_Data".t1		Transition 1
Tag_1	%M20.0	
"Graph_Data".switch1		additional condition

"Transition/Interlock view"

You can switch between the transition/interlock views during the display of your user program. The transition/interlock view is available when the detail view is displayed.

You switch between the transition/interlock views in Runtime by using the "Transition or Interlock" button. When the button is enabled the interlock network is displayed, when disabled the transition network.

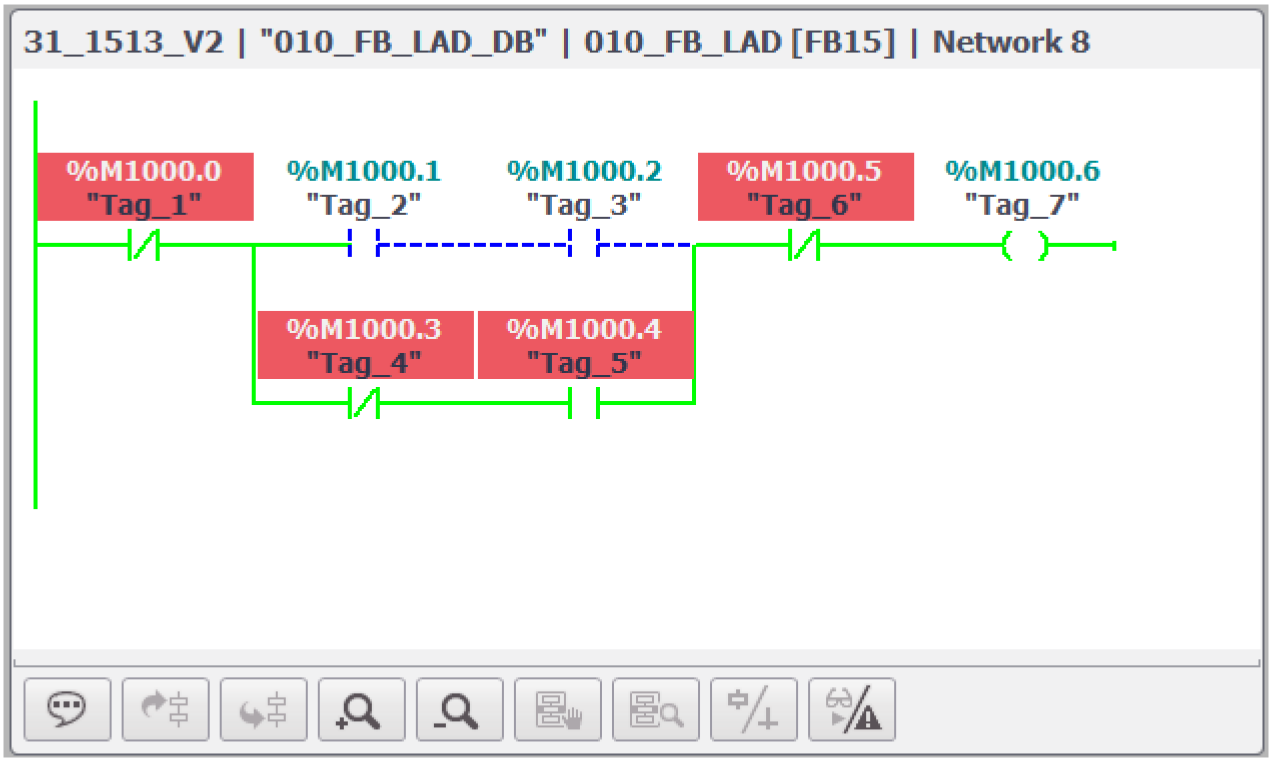


Initial value/actual value view

If you have activated the initial value recording in the user program for the corresponding function block, you can switch in the PLC code view between the initial value/actual value views. The actual value view uses the current values from the PLC and displays the current program status. The initial value view uses the values that were recorded at the time of the error and displays the operands with error and criteria. The operands with error are visually highlighted in the initial value view.

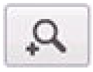
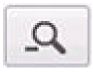
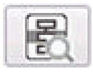
To switch between the actual values and initial values, use the "Initial values or actual values" button. When initial value acquisition is activated, the initial value view is displayed by default in the PLC code view after the jump.

If errors occur, the faulty operands are highlighted in the initial value view in Runtime.



Buttons on the toolbar

The following table shows the toolbar buttons and their meaning.

Operator control	Designation	Function
	"Symbol area"	Shows the symbol table.
	"Previous network"	Navigates to the previous network
	"Next network"	Navigates to the next network
	"Zoom in"	Enlarges the information area.
	"Zoom out"	Reduces the information area.
	"Detail"	Shows the detail view.
	"Step mode"	Switches between manual step selection and automatic selection of the active step.

Operator control	Designation	Function
	"Transition or Interlock"	Switches between the representation of transition and interlock networks,
	"Actual values or initial val-ues"	Switches between actual value view and initial val-ue view.

7.3 Supported instructions

Introduction

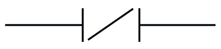
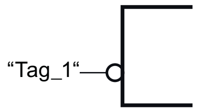
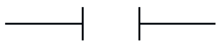
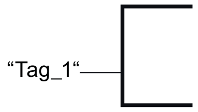
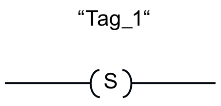
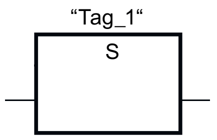
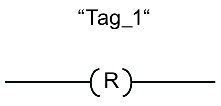
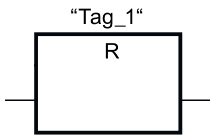
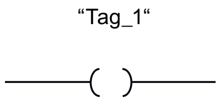
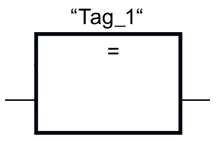
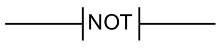
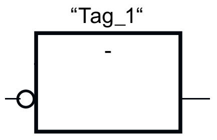
The "PLC code view" control supports instructions in the FBD and LAD programming languages.

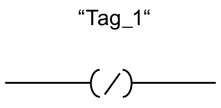
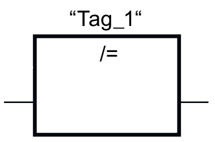
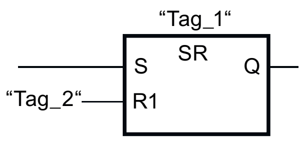
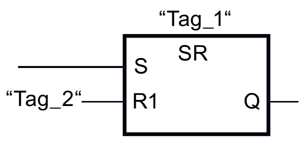
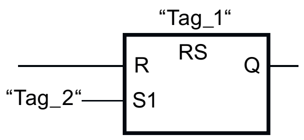
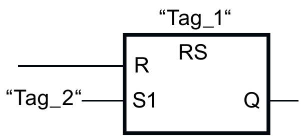
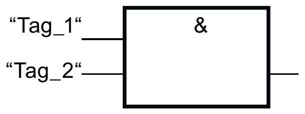
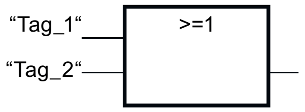
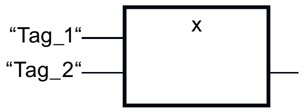
Instructions

The following instructions are supported in the PLC code view:

- Bit logic operations
- Comparator operations
- Timers
- Counters

Bit logic operations

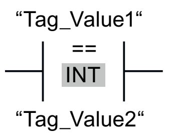
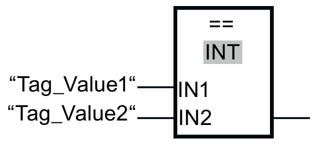
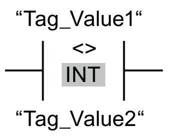
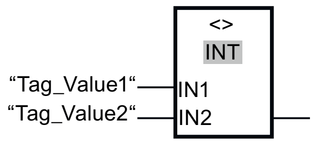
Instruction	LAD	FBD
Normally closed contact		
Normally open contact		
Set output		
Reset output		
Assignment		
NOT: Invert RLO		

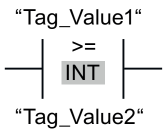
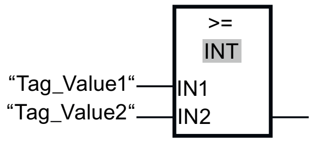
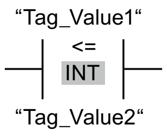
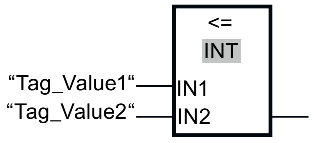
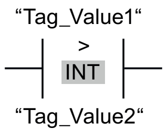
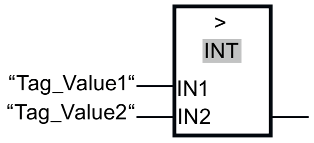
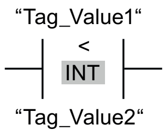
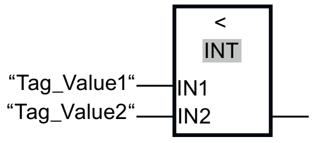
Instruction	LAD	FBD
Negated contact		
SR: Set/reset flip-flop		
RS: Reset/set flip-flop		
AND	Is implemented by the corresponding interconnection of normally closed contacts or normally open contacts.	
OR	Is implemented by the corresponding interconnection of normally closed contacts or normally open contacts.	
EXCLUSIVE OR	Is implemented by the corresponding interconnection of normally closed contacts or normally open contacts.	

You can find additional information in the documentation "SIMATIC STEP 7 Basic/Professional V18 and SIMATIC WinCC V18":

- SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Instructions" (https://support.industry.siemens.com/cs/ww/en/view/109815056/162415423115)

Comparator operations

Instruction	LAD	FBD
CMP ==: Equal		
CMP <>: Not equal		

Instruction	LAD	FBD
CMP >=: Greater or equal		
CMP <=: Less or equal		
CMP >: Greater than		
CMP <: Less than		

Note

Comparator: Supported data types

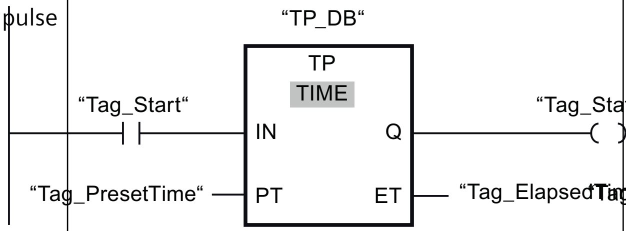
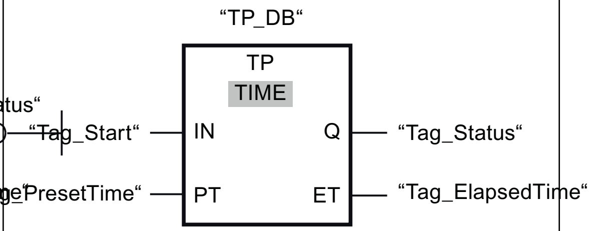
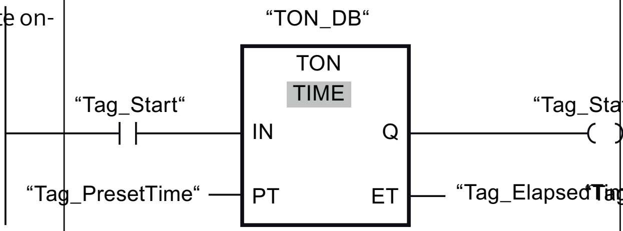
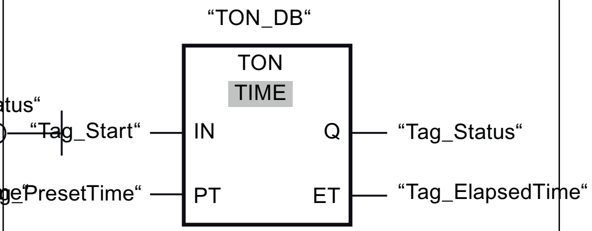
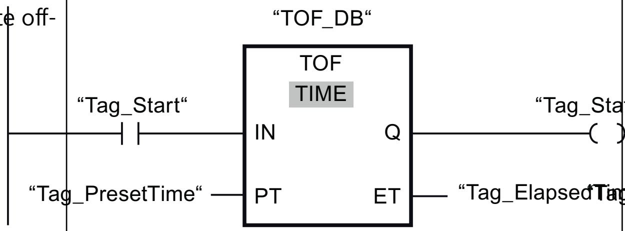
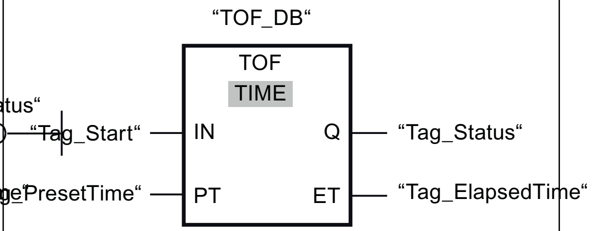
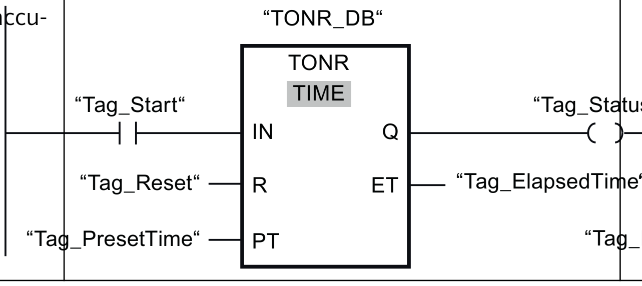
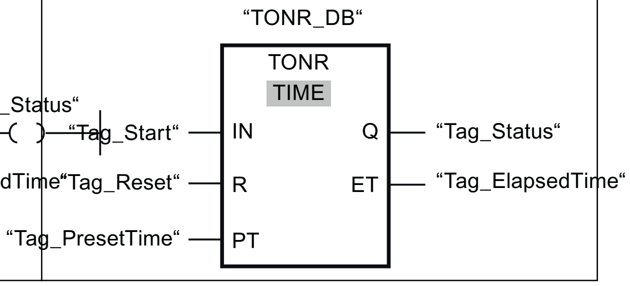
The PLC code view supports the following data types when displaying the sequencer:

- Binary numbers
- Integers
- Floating-point numbers
- Times, exception: S5TIME

You can find additional information in the documentation "SIMATIC STEP 7 Basic/ Professional V18 and SIMATIC WinCC V18":

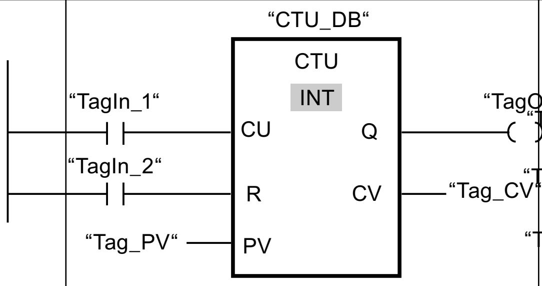
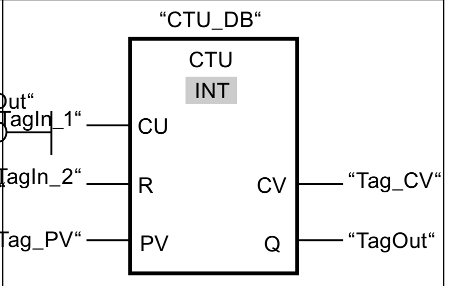
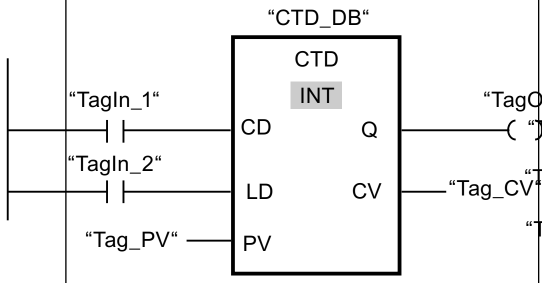
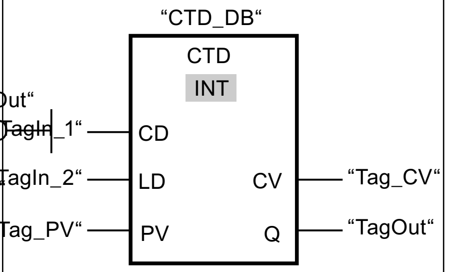
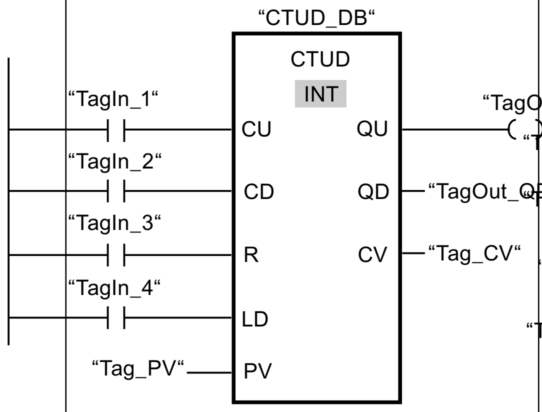
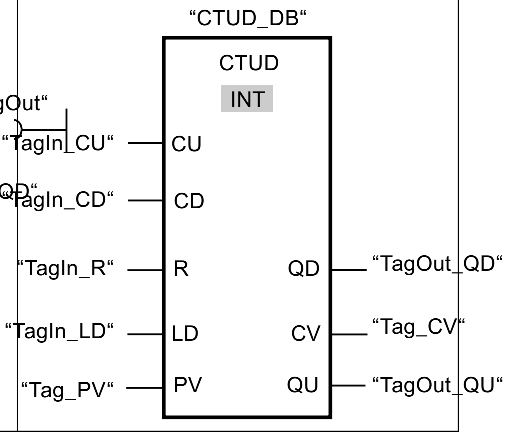
- SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Instructions" (https://support.industry.siemens.com/cs/ww/en/view/109815056/162415423115)

Timers

Instruction	LAD	FBD
TP: Generate pulse	 "TP_DB" TP TIME "Tag_Start" IN Q "Tag_Status" "Tag_PresetTime" PT ET "Tag_ElapsedTime"	 "TP_DB" TP TIME "Tag_Start" IN Q "Tag_Status" "Tag_PresetTime" PT ET "Tag_ElapsedTime"
TON: Generate on-delay	 "TON_DB" TON TIME "Tag_Start" IN Q "Tag_Status" "Tag_PresetTime" PT ET "Tag_ElapsedTime"	 "TON_DB" TON TIME "Tag_Start" IN Q "Tag_Status" "Tag_PresetTime" PT ET "Tag_ElapsedTime"
TOF: Generate off-delay	 "TOF_DB" TOF TIME "Tag_Start" IN Q "Tag_Status" "Tag_PresetTime" PT ET "Tag_ElapsedTime"	 "TOF_DB" TOF TIME "Tag_Start" IN Q "Tag_Status" "Tag_PresetTime" PT ET "Tag_ElapsedTime"
TONR: Time accu-mulator	 "TONR_DB" TONR TIME "Tag_Start" IN Q "Tag_Status" "Tag_Reset" R ET "Tag_ElapsedTime" "Tag_PresetTime" PT	 "TONR_DB" TONR TIME "Tag_Start" IN Q "Tag_Status" "Tag_Reset" R ET "Tag_ElapsedTime" "Tag_PresetTime" PT

Counters *

* Counters support the integer data types.

Instruction	LAD	FBD
CTU: Count up	 "CTU_DB" CTU INT "TagIn_1" CU Q "TagOut" "TagIn_2" R CV "Tag_CV" "Tag_PV" PV	 "CTU_DB" CTU INT "TagIn_1" CU "TagIn_2" R CV "Tag_CV" "Tag_PV" PV Q "TagOut"
CTD: Count down	 "CTD_DB" CTD INT "TagIn_1" CD Q "TagOut" "TagIn_2" LD CV "Tag_CV" "Tag_PV" PV	 "CTD_DB" CTD INT "TagIn_1" CD "TagIn_2" LD CV "Tag_CV" "Tag_PV" PV Q "TagOut"
CTUD: Count up and down	 "CTUD_DB" CTUD INT "TagIn_1" CU QU "TagOut" "TagIn_2" CD QD "TagOut_QD" "TagIn_3" R CV "Tag_CV" "TagIn_4" LD "Tag_PV" PV	 "CTUD_DB" CTUD INT "TagIn_CU" CU "TagIn_CD" CD "TagIn_R" R QD "TagOut_QD" "TagIn_LD" LD CV "Tag_CV" "Tag_PV" PV QU "TagOut_QU"

See also

Supported data types (Page 38)

SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Instructions" (https://support.industry.siemens.com/cs/ww/en/view/109815056/162415423115)

7.4 Supported data types

Supported data types

The following table shows the data types supported in the "PLC Code View" control and their display formats:

Data type	Length	Format
Binary numbers		
BOOL *	1 bit	Bool
BYTE	8 bits	Hex
WORD	16 bits	Hex
DWORD	32 bits	Hex
LWORD	64 bits	Hex
Integers		
SINT	8 bits	Dec
USINT	8 bits	Dec
INT	16 bits	Dec
UINT	16 bits	Dec
DINT	32 bits	Dec
UDINT	32 bits	Dec
LINT	64 bits	Dec
ULINT	64 bits	Dec
Floating-point numbers		
REAL	32 bits	Floating point number
LREAL	64 bits	Floating point number
Timers		
S5TIME	16 bits	SIMATIC_Timer
TIME	32 bits	Time
LTIME	64 bits	LTime
Date and time		
DTL **	12 bytes	Date and Time
* Only for the Boolean operators ** Complete DTL structures are not supported. Only single elements of DTL structures are supported.		

You can find additional information in the documentation "SIMATIC STEP 7 Basic/Professional V18 and SIMATIC WinCC V18":

- SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Data types" (https://support.industry.siemens.com/cs/ww/en/view/109815056/160201843595)

See also

Supported instructions (Page 33)

SiePortal: "STEP 7 and WinCC Engineering V18 > Programming PLC > Data types" (https://support.industry.siemens.com/cs/ww/en/view/109815056/160201843595)

7.5 Restrictions for the PLC code view

SCL

If you access a program network created with SCL, error code 4100 is output.

Operands and UDTs

Operands that are declared in the "#Temp" or "#InOut" area are generally not supported by the PLC code view. This applies both to elementary data types and to data types that are contained in UDTs. Data types of a UDT can be declared in the "#In" and "#Out" area and displayed in the PLC code view. The same limitations as for elementary data types apply to the data types of the UDT.

Limitation regarding the data types

Using the data types STRING, WSTRING, CHAR, WCHAR, S5TIME prevents the network from being displayed in the PLC code view.

Note

The upstream network must not contain any tags from the "#Temp" or "#InOut" areas of an FB.

The following data types may not be used for tags: STRING, WSTRING, CHAR, WCHAR, S5TIME

Note

The 64-bit PLC data types LINT, ULINT and LWORD are mapped to the HMI data type LREAL in the HMI channel. There is a loss of accuracy at values of > 2^50.

Use of 64-bit data types

The use of 64-bit data types may result in a slight loss of accuracy as these data types are mapped in the HMI channel to the data type Double. It is therefore possible that integer data types are displayed with decimal places.

Addressing of tags

Please note the following restrictions when addressing tags:

- The "PLC code view" control only supports symbolic addressing of tags. If the operand is not addressed symbolically, the network with the operand cannot be displayed and an error message is generated.
- The use of array elements with a tag as index is not supported, e.g. #myArray[MyTag]. Use numerical values when addressing the array elements, for example #myArray[6].
- Slice access enables you to address specific areas within declared variables. The PLC code view only supports the access width "Bit" on the Boolean tags during the slice accesses.

Jump to the PLC code view

For the jump from a supervision alarm to the PLC code view, the instance name must conform to the following naming convention when using supported local operands in a function block: <FB-Name>_DB.

The jump to a function or an organization block is only possible if only global operands are used.

8 Configuring the display of criteria analysis

Introduction

The "Criteria analysis" control (WinCC CriteriaAnalysisControl) shows the operands in the user program that contain errors and have triggered a selected alarm. In addition to the alarm, the list of operands with errors is shown in the same picture.

Requirement

- A connection has been configured.
- A connection between the PLC and HMI device has been created.
- A picture has been created.
- An alarm control (WinCC AlarmControl) has been configured and connected to the ProDiag controls on the picture.
- More information:
  - Configuring the ProDiag overview (Page 11)
  - Configuring a GRAPH overview (Page 21)

Procedure

1. Use drag-and-drop to move the control "Criteria analysis" (WinCC CriteriaAnalysisControl) from the palette "Controls" > "Controls" selection window > "ProDiag Controls" node into the configured picture.
2. In the "Object Properties" window, click on "Properties > Control Properties".
3. Enter the name of the configured alarm view under "Control Properties > SourceControl".
4. In a multi-client system, enter the "Server prefix" under ServerPrefixName.

See also

Configuring the ProDiag overview (Page 11)

Configuring a GRAPH overview (Page 21)

9 Functions for displaying PLC code

9.1 Display in STEP 7

9.1.1 Basics

General information

You can use this function to switch from a screen in WinCC Runtime directly to the point of use of a process tag in the program code of STEP 7. This provides the possibility of quick and simple diagnostics of faults. To speed up the jump function, it is possible to start WinCC and open a project.

During the jump, there is first a check to determine whether the project is already open. If this is not the case, the project is started automatically. In WinCC, the corresponding editor is opened and the point of use, the assignment, the call or the step is searched for.

Header files

The declaration of the functions and structures explained here takes place in the header files:

kopapi.h	API interface definition header file

Libraries

The DLL link for the functions explained here is provided in the Lib files kopapi.dll.

kopapi.lib	Library
kopapi.dll	

9.1.2 OpenTIAPortalProject

Description

This function can be used to open a project. This will speed up the individual jump functions. Bear in mind that a program open in the background requires memory and computing time.

Declaration

```
BOOL OpenTIAPortalProject (
    DWORD      dwFlags,
    LPCTSTR    lpszTiaPortalProjectPath,
    LPCTSTR    lpszErrorTag,
    LPCMN_ERROR    lpdmError);
```

Parameters

dwFlags

Bit array in which the individual values are ORed bit-by-bit. dwFlags should be 0 by default.

- KOPAPI_FLAG_TIAPORTAL_CHECK_PROJECT_STATE: If this bit is set, the status of the project is checked. The project is not opened.
- The project is open.
  Return value FALSE
  error.dwError1 = KOPAPI_E_TIAPORTAL_PROJECT_NOT_OPEN
- The project is open and contains no changes.
  Return value TRUE
  error.dwError1 = 0
- The project is open and contains changes.
  Return value FALSE
  error.dwError1 = KOPAPI_E_TIAPORTAL_PROJECT_MODIFIED
- The project is opened read only.
  Return value FALSE
  error.dwError1 = KOPAPI_E_TIAPORTAL_PROJECT_READ_ONLY

KOPAPI_FLAG_TIAPORTAL_DONT_USE_MODIFIED_PROJECT: If this bit is set, the call is canceled when the project is already open and contains changes.

KOPAPI_FLAG_TIAPORTAL_OPEN_READONLY: When this bit is set to TRUE, the project is opened in write-protected mode in the engineering system of the TIA Portal.

lpszTiaPortalProjectPath

Name of the project including specification of the absolute path, e.g. "D:\TIAProjects\Project1\Project1.ap19"

Please note that within a C script the backslash has to be written using an Escape sequence:

FunctionX(..., "D:\\TIAProjects\\Project1\\Project1.ap19", ...);

lpszErrorTag

Name of an internal WinCC tag of the String data type. When asynchronous functions that do not supply a result immediately are called, the error information is returned in `lpszErrorTag`.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

9.1.3 OpenTIAPortalIECPLByCall

Description

The function is used for the LAD and FBD languages and shows the preceding logic of a network input of a standard block.

Declaration

```
BOOL OpenTIAPortalIECPLByCall (
    DWORD      dwFlags,
    LPCTSTR    lpszTiaPortalProjectPath,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszContainingBlock,
    LPCTSTR    lpszCalledBlock,
    LPCTSTR    lpszPin,
    LPCTSTR    lpszErrorTag,
    LPCMN_ERROR    lpdmError);
```

Parameters

dwFlags

Bit array in which the individual values are ORed bit-by-bit. dwFlags should be 0 by default.

- IECPLVIEWER_PIN_SUBSTRING_SEARCH=0x0001: A substring is sought in the search for the pin name, which means the pin name starts with the string transferred in lpszPin. If this bit is not set, the complete pin name is compared to lpszPin.
- KOPAPI_FLAG_TIAPORTAL_SUPPRESS_PROGRAM_STATUS=0x0004: The TIA Portal does not go into online mode after the block is opened. If this bit is not set, online mode is started after opening the block.

- KOPAPI_FLAG_TIAPORTAL_DONT_USE_MODIFIED_PROJECT 0x0008L: If this bit is set, the call is canceled when the project is already open and contains changes.
- KOPAPI_FLAG_TIAPORTAL_OPEN_READONLY: When this bit is set to TRUE, the project is opened in write-protected mode in the engineering system of the TIA Portal.

lpszTiaPortalProjectPath

Name of the project including specification of the absolute path, e.g. "D:\TIAProjects\Project1\Project1.ap19"

Please note that within a C script the backslash has to be written using an Escape sequence:

FunctionX(..., "D:\\TIAProjects\\Project1\\Project1.ap19", ...);

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal.

lpszContainingBlock

Name of the block to be opened and displayed or name of the instance of an FB.

The following can be used as name:

- Name of a single instance DB. Its FB is then displayed. Example "Station1"
- Name of a multi-instance in an instance DB. Its FB is then displayed. When multi-instance name paths are specified, this is the data hierarchy as it is shown in the DB editor, for example, and not the call structure. The first part of the name ("Line1") cannot be enclosed in quotes, because this is a global icon, as can be recognized by the context. Quotation marks are required for the individual name components if they contain special characters such as spaces, periods etc. Example: "Line1.Cell1.Station1"
- Name of an FC or OB

Using the name of an FB is not permissible.

lpszCalledBlock

Name of the local or global instance that is called in the code block belonging to lpszContainingBlock.

- With local instances, the hash symbol # must also be included here, e.g. "#feeder1".
- With global instance DBs, the global name must be specified without the hash symbol #, e.g. "feeder3".

The use of the name of an FC is not allowed.

If lpszCalledBlock is called several times within lpszContainingBlock or its FB, the entry point is always the first call of lpszCalledBlock.

With lpszCalledBlock=ZERO, only lpszContainingBlock is opened and shown in the status, but there is no search for a specific block call or a specific network. lpszPin is ignored in this case.

lpszPin

Name of the input pin of lpszCalledBlock. The parameter is used to make the specified pin visible within the network in the editor.

Only lpszCalledBlock is shown as visible with lpszPin=ZERO.

lpszErrorTag

Name of an internal WinCC tag of the String data type. When asynchronous functions that do not supply a result immediately are called, the error information is returned in `lpszErrorTag`.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

9.1.4 OpenTIAPortalIECPLByAssignment

Description

The function is used for the LAD and FBD languages and shows the assignment to an operand and its preceding logic.

Declaration

```
BOOL OpenTIAPortalIECPLByAssignment (
    DWORD      dwFlags,
    LPCTSTR    lpszTiaPortalProjectPath,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszContainingBlock,
    LPCTSTR    lpszOperand,
    LPCTSTR    lpszErrorTag,
    LPCMN_ERROR    lpdmError);
```

Parameters

dwFlags

Bit array in which the individual values are ORed bit-by-bit. dwFlags should be 0 by default.

- KOPAPI_FLAG_TIAPORTAL_SUPPRESS_PROGRAM_STATUS=0x0004: The TIA Portal does not go to online mode after the block has been opened. If this bit is not set, online mode is started after opening the block.
- KOPAPI_FLAG_TIAPORTAL_DONT_USE_MODIFIED_PROJECT 0x0008L: If this bit is set, the call is canceled when the project is already open and contains changes.
- KOPAPI_FLAG_TIAPORTAL_OPEN_READONLY: When this bit is set to TRUE, the project is opened in write-protected mode in the engineering system of the TIA Portal.

lpszTiaPortalProjectPath

Name of the project including specification of the absolute path, e.g. "D:\TIAProjects\Project1\Project1.ap19"

Please note that within a C script the backslash has to be written using an Escape sequence:

FunctionX(..., "D:\\TIAProjects\\Project1\\Project1.ap19", ...);

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal.

lpszContainingBlock

Name of the block to be opened and displayed or name of the instance of an FB.

The following can be used as name:

- Name of a single instance DB. Its FB is then displayed. Example "Station1"
- Name of a multi-instance in an instance DB. Its FB is then displayed. When multi-instance name paths are specified, this is the data hierarchy as it is shown in the DB editor, for example, and not the call structure. The first part of the name ("Line1") cannot be enclosed in quotes, because this is a global icon, as can be recognized by the context. Quotation marks are required for the individual name components if they contain special characters such as spaces, periods etc. Example: "Line1.Cell1.Station1"
- Name of an FC or OB

Using the name of an FB is not permissible.

lpszOperand

Name of a local or global operand to which an assignment is made.

Name of the local or global instance that is called in the code block belonging to lpszContainingBlock.

- For local operands, the hash sign # must also be specified here.
- For global operands, the global name without hash sign # must be specified.

If lpszOperand is written several times within lpszContainingBlock or its FB, the entry point is always the first write access of lpszOperand.

lpszErrorTag

Name of an internal WinCC tag of the String data type. When asynchronous functions that do not supply a result immediately are called, the error information is returned in `lpszErrorTag`.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

9.1.5 OpenTIAPortalS7GraphByBlock

Description

The function is used for the S7 Graph languages and shows a step in a sequencer.

Declaration

```
BOOL OpenTIAPortalS7GraphByBlock (
    DWORD      dwFlags,
    LPCTSTR    lpszTiaPortalProjectPath,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszBlock,
    DWORD      dwStepNumber,
    LPCTSTR    lpszErrorTag,
    LPCMN_ERROR    lpdmError);
```

Parameters

dwFlags

Bit array in which the individual values are ORed bit-by-bit. dwFlags should be 0 by default.

- KOPAPI_FLAG_TIAPORTAL_SUPPRESS_PROGRAM_STATUS=0x0004: The TIA Portal does not go to online mode after the block has been opened. If this bit is not set, online mode is started after opening the block.
- KOPAPI_FLAG_TIAPORTAL_DONT_USE_MODIFIED_PROJECT 0x00000008L: If this bit is set, the call is canceled when the project is already open and contains changes.
- KOPAPI_FLAG_TIAPORTAL_OPEN_READONLY: When this bit is set to TRUE, the project is opened in write-protected mode in the engineering system of the TIA Portal.

lpszTiaPortalProjectPath

Name of the project including specification of the absolute path, e.g. "D:\TIAProjects\Project1\Project1.ap19"

Please note that within a C script the backslash has to be written using an Escape sequence:

FunctionX(..., "D:\\TIAProjects\\Project1\\Project1.ap19", ...);

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal.

lpszBlock

Instance name of the S7-Graph block to be displayed.

dwStepNumber

Number of the step to be displayed.

dwStepNumber=0 automatically searches for the active step and enables "Track active step" mode.

lpszErrorTag

Name of an internal WinCC tag of the String data type. When asynchronous functions that do not supply a result immediately are called, the error information is returned in `lpszErrorTag`.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

9.1.6 Example: Integration in a WinCC function

Description

The example shows the embedding of the function calls in a user-defined function. For better maintainability, the call should be swapped out to a global script function in which the path to the TIA Portal and the name of the PLC station are also determined.

```
void OnClick(char* screenName, char* objectName, char* propertyName)
    {
    // Funktionen bekannt machen'
    #pragma code("KOPAPI.dll")
    #include "KOPAPI.h"
    #pragma code()

    BOOL useTiaPortal = TRUE; // use the TIA Portal or the viewer control?
    char* pTiaPortalProject = "c:\\Projects\\myproject\\myproject.ap14";
    char* pStationName = "PLC_Name"; // TODO: read from internal tag
   char* pContainingBlock = "Block_IDB"; // TODO: get from current selection
    char* pOperand = "OUT"; // TODO: get from current selection
    CMN_ERROR error;
    BOOL result;

    if(useTiaPortal)
        {
        result = OpenTIAPortalIECPLByAssignment(0, pTiaPortalProject,
pStationName, pContainingBlock, pOperand, &error);
        }
    else
        {
        char *pServerPrefix = NULL, *pTagPrefix = NULL, *pWindowPrefix =
NULL;
        // determine ServerPrefix of the current environment
        GetServerTagPrefix(&pServerPrefix, &pTagPrefix, &pWindowPrefix);
        // make the screen which contains the viewer control visible
        SetVisible("SYSTEM#Basic_Screen", "Screen_window_IECPLViewer",
TRUE);
        result = OpenViewerIECPLByAssignment(0, pServerPrefix,
"SYSTEM#IECPLViewer", "IECPLViewerObject", pStationName, pContainingBlock,
pOperand, &error);
        }
}
```

9.2 Display in the PLC code view

9.2.1 Basics

General information

You can use these functions to display the current program status of PLC programs in a PLC code view.

The screen that contains the PLC code view must have been opened via ActivateScreen(), for example.

Header files

The declaration of the functions and structures explained here takes place in the header files:

kopapi.h	API interface definition header file

Libraries

The DLL link for the functions explained here is provided in the Lib files: pdecscli.dll

kopapi.lib	Library
kopapi.dll	

Note on defining multi-instance names

When multi-instance name paths are specified, this is the data hierarchy as it is shown in the DB editor, for example, and not the call structure. The first part of the name ("Line1") cannot be enclosed in quotes, because this is a global icon, as can be recognized by the context. Quotation marks are required for the individual name components if they contain special characters such as spaces, periods etc. Example: "Line1.Cell1.Station1".

Names that do not contain special characters must not be put in quotation marks.

9.2.2 OpenViewerS7GraphByBlock

Description

The function shows a step from a sequencer called in the PLC language S7-Graph in the PLC code view.

Declaration

```
BOOL OpenViewerS7GraphByBlock (
    DWORD      dwFlags,
    LPCTSTR    lpszServerPrefix,
    LPCTSTR    lpszPictureName,
    LPCTSTR    lpszObjectName,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszBlock,
    DWORD      dwStepNumber,
    LPCMN_ERROR    lpdmError);
```

Parameter

dwFlags

Bit array in which the individual values are ORed bit-by-bit.

Note

The bit 0x4 must be set to show hierarchical comments.

lpszServerPrefix

The parameter is reserved for later upgrades.

lpszPictureName

Name of the screen with the PLC code view.

lpszObjectName

Name of the PLC code view.

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal. No commas are permitted in the name.

lpszBlock

Instance name of the S7-Graph block to be displayed. If the name contains special characters such as spaces, periods etc., quotation marks must be used.

dwStepNumber

Number of the step to be displayed.

dwStepNumber=0 automatically searches for the active step and enables "Track active step" mode.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

Example

The example shows the embedding of the function call in a user-defined C function. For better maintainability, the call should be swapped out to a global script function in which the name of the PLC station is also determined.

```
BOOL OpenViewerS7GraphByBlock(char* screenName, char* objectName, char*
cpuName, char* instanceDBName)
{
#pragma code("kopapi.dll")
#include "kopapi.h"
#pragma code()
BOOL result;
CMN_ERROR error;
char* serverPrefix = "";
DWORD dwFlags = 0;
DWORD stepNumber = 0;
result = OpenViewerS7GraphByBlock(dwFlags, serverPrefix, screenName,
objectName,
  cpuName, instanceDBName, stepNumber, &error);
if(!result)
{
  // there are only few reasons why the call to OpenViewerS7GraphByBlock will
  // fail, in most cases the viewer control could not be found
  // most of the errors have to be handled in the OnError event of the viewer
  printf("OpenViewerS7GraphByBlock failed: err1=%ld, err2=%ld, err3=%ld,
err4=%ld,   err5=%ld, text=\"%s\"\r\n", result, error.dwError1,
error.dwError2,   error.dwError3, error.dwError4, error.dwError5,
error.szErrorText);
}
return result;
}
```

9.2.3 OpenViewerIECPLByCall

Description

The function shows the preceding logic of a network input of a standard block in the PLC code view in the PLC languages LAD and FBD.

Note

If the input is occupied by a constant (TRUE, FALSE), this value is not displayed in the PLC code view. To display the value in the PLC code view correctly, you should allocate a tag to the input that has the appropriate value.

Declaration

```
BOOL OpenViewerIECPLByCall (
    DWORD      dwFlags,
    LPCTSTR    lpszServerPrefix,
    LPCTSTR    lpszPictureName,
    LPCTSTR    lpszObjectName,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszContainingBlock,
    LPCTSTR    lpszCalledBlock,
    LPCTSTR    lpszPin,
    LPCMN_ERROR    lpdmError);
```

Parameter

dwFlags

Bit array in which the individual values are ORed bit-by-bit.

Note

The bit 0x4 must be set to show hierarchical comments.

- IECPLVIEWER_PIN_SUBSTRING_SEARCH=0x0001: A substring is sought in the search for the pin name, which means the pin name starts with the string transferred in lpszPin. If this bit is not set, the complete pin name is compared to lpszPin.

lpszServerPrefix

The parameter is reserved for later upgrades.

lpszPictureName

Name of the screen with the PLC code view.

lpszObjectName

Name of the PLC code view.

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal. No commas are permitted in the name.

lpszContainingBlock

Name of the block to be opened and displayed or name of the instance of an FB.

The following can be used as the name:

- Name of a single instance DB. Its FB is then displayed. Example "Station1"
- Name of a multi-instance in an instance DB. Its FB is then displayed. When multi-instance name paths are specified, these are data hierarchies such as those displayed in the DB editor, and not those in the call structure. The first part of the name ("Line1") cannot be enclosed in quotes, because this is a global icon, as can be recognized by the context. Quotation marks are required for the individual name components when special characters such as spaces, dots, etc., occur in it. Example: "Line1.Cell1.Station1"
- Name of an FC or OB

The use of the name of a FB is not allowed.

lpszCalledBlock

Name of the local or global instance that is called in the code block belonging to lpszContainingBlock.

- For local instances, the hash sign # must be specified here, for example, "#feeder1".
- For global instances, the global name without hash sign # must be specified here, for example, "feeder3".

The use of the name of an FC is not allowed.

If lpszCalledBlock is called several times within lpszContainingBlock or its FB, the entry point is always the first call of lpszCalledBlock.

lpszPin

Name of the input pin of lpszCalledBlock. The parameter is used to display the network interconnected to the input pin in the PLC code view.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

Example

The example shows the embedding of the function call in a user-defined C function. For better maintainability, the call should be swapped out to a global script function in which the name of the PLC station is also determined.

```
BOOL OpenCodeViewerByCall(BOOL matchSubstringPin, char* screenName, char*
objectName, char* cpuName, char* containingBlock, char* calledBlock, char*
pinName)
{
#pragma code("kopapi.dll")
#include "kopapi.h"
#pragma code()
BOOL result;
CMN_ERROR error;
char* serverPrefix = "";
DWORD dwFlags = 0;
if(matchSubstringPin)
  dwFlags |= KOPAPI_FLAG_TIAPORTAL_PIN_SUBSTRING_SEARCH;
result = OpenViewerIECPLByCall(dwFlags, serverPrefix, screenName,
objectName,
  cpuName, containingBlock, calledBlock, pinName, &error);
if(!result)
{
  // there are only few reasons why the call to OpenViewerIECPLByCall will
fail, in   // most cases the viewer control could not be found
  // most of the errors have to be handled in the OnError event of the viewer
  printf("OpenViewerIECPLByCall failed: err1=%ld, err2=%ld, err3=%ld,
err4=%ld,   err5=%ld, text=\"%s\"\r\n", result, error.dwError1,
error.dwError2,   error.dwError3, error.dwError4, error.dwError5,
error.szErrorText);
}
return result;
}
```

9.2.4 OpenViewerIECPLByFCCall

Description

The function shows the preceding logic of a network input of a standard block with consideration of the UDT instance in the PLC code view for the PLC languages LAD and FBD.

Note

If the input is occupied by a constant (TRUE, FALSE), this value is not displayed in the PLC code view. To display the value in the PLC code view correctly, you should allocate a tag to the input that has the appropriate value.

Declaration

```
BOOL OpenViewerIECPLByFCCall (
    DWORD      dwFlags,
    LPCTSTR    lpszServerPrefix,
    LPCTSTR    lpszPictureName,
    LPCTSTR    lpszObjectName,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszContainingBlock,
    LPCTSTR    lpszCalledBlock,
    LPCTSTR    lpszPin,
    LPCTSTR    lpszUdtInstance
    LPCMN_ERROR    lpdmError);
```

Parameters

dwFlags

Bit array in which the individual values are ORed bit-by-bit.

Note

The bit 0x4 must be set to show hierarchical comments.

- IECPLVIEWER_PIN_SUBSTRING_SEARCH=0x0001: A substring is sought in the search for the pin name, which means the pin name starts with the string transferred in lpszPin. If this bit is not set, the complete pin name is compared to lpszPin.

lpszServerPrefix

The parameter is reserved for future development and must be an empty string ("").

lpszPictureName

Name of the screen with the PLC code view.

lpszObjectName

Name of the PLC code view.

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal. No commas are permitted in the name.

lpszContainingBlock

Name of the block to be opened and displayed or name of the instance of an FB.

The following can be used as name:

- Name of a single instance DB. Its FB is then displayed. Example "Station1"
- Name of a multi-instance in an instance DB. Its FB is then displayed. When multi-instance name paths are specified, this is the data hierarchy as it is shown in the DB editor, for example, and not the call structure. The first part of the name ("Line1") cannot be enclosed in quotes, because this is a global icon, as can be recognized by the context. Quotation marks are required for the individual name components if they contain special characters such as spaces, periods etc. Example: "Line1.Cell1.Station1"
- Name of an FC or OB

Using the name of an FB is not permissible.

lpszCalledBlock

Name of the local or global instance that is called in the code block belonging to lpszContainingBlock.

- With local instances, the hash symbol # must also be included here, e.g. "#feeder1".
- With global instance DBs, the global name must be specified without the hash symbol #, e.g. "feeder3".

The use of the name of an FC is not allowed.

If lpszCalledBlock is called several times within lpszContainingBlock or its FB, the entry point is always the first call of lpszCalledBlock.

lpszPin

Name of the input pin of lpszCalledBlock. The parameter is used to display the network interconnected to the input pin in the PLC code view.

lpszUdtInstance

The parameter is used to limit the display of FBs or FCs called multiple times. The limitation is based on an UDT instance interconnected to a given input pin or inout pin.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The function has been completed without any errors.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

Example

The example shows the embedding of the function call in a user-defined C function. For better maintainability, the call should be swapped out to a global script function in which the name of the PLC station is also determined.

```
BOOL OpenCodeViewerByFCCall(BOOL matchSubstringPin, char* screenName,
char* objectName, char* cpuName, char* containingBlock, char* calledBlock,
char* pinName, char* udtInstance)
{
#pragma code("kopapi.dll")
#include "kopapi.h"
#pragma code()
BOOL result;
CMN_ERROR error;
char* serverPrefix = "";
DWORD dwFlags = 0;
if(matchSubstringPin)
  dwFlags |= KOPAPI_FLAG_TIAPORTAL_PIN_SUBSTRING_SEARCH;
result = OpenViewerIECPLByFCCall(dwFlags, serverPrefix, screenName,
objectName, cpuName, containingBlock, calledBlock, pinName, udtInstance,
&error);
if(!result)
{
  // there are only few reasons why the call to OpenViewerIECPLByFCCall will
fail,
// in most cases the viewer control could not be found
// most of the errors have to be handled in the OnError event of the viewer
control
  printf("OpenViewerIECPLByFCCall failed: err1=%ld, err2=%ld, "" err3=%ld,
err4=%ld,   err5=%ld, text=\"%s\"\r\n", result, error.dwError1,
error.dwError2,   error.dwError3, error.dwError4, error.dwError5,
error.szErrorText);
}
return result;
}
```

9.2.5 OpenViewerIECPLByAssignment

Description

The function shows the assignment to an operand and its preceding logic in the PLC code view for the PLC languages LAD and FBD.

Declaration

```
BOOL OpenViewerIECPLByAssignment (
    DWORD      dwFlags,
    LPCTSTR    lpszServerPrefix,
    LPCTSTR    lpszPictureName,
    LPCTSTR    lpszObjectName,
    LPCTSTR    lpszCpuName,
    LPCTSTR    lpszContainingBlock,
    LPCTSTR    lpszOperand,
    LPCMN_ERROR    lpdmError);
```

Parameter

dwFlags

Bit array in which the individual values are ORed bit-by-bit.

Note

The bit 0x4 must be set to show hierarchical comments.

lpszServerPrefix

The parameter is reserved for later upgrades.

lpszPictureName

Name of the screen with the PLC code view.

lpszObjectName

Name of the PLC code view.

lpszCpuName

Name of the S7-CPU. The name is identical to the station name displayed in the project tree in the TIA Portal. No commas are permitted in the name.

lpszContainingBlock

Name of the block to be opened and displayed or name of the instance of an FB.

The following can be used as name:

- Name of a single instance DB. Its FB is then displayed. Example "Station1"
- Name of a multi-instance in an instance DB. Its FB is then displayed. When multi-instance name paths are specified, this is the data hierarchy as it is shown in the DB editor, for example, and not the call structure. The first part of the name ("Line1") cannot be enclosed in quotes, because this is a global icon, as can be recognized by the context. Quotation marks are required for the individual name components if they contain special characters such as spaces, periods etc. Example: "Line1.Cell1.Station1"
- Name of an FC or OB

Using the name of an FB is not permissible.

lpszOperand

Name of a local or global operand to which an assignment is made.

Name of the local or global instance that is called in the code block belonging to lpszContainingBlock. For global operands, the global name without hash sign # must be specified.

The following restrictions apply to operands:

- Tags from the "TEMP" area cannot be used
- Input tags cannot be used
- Static tags are only allowed in FBs
- Networks within an FC can only be displayed if no local tags are used within the network
- Networks within an FB that use tags from the "TEMP" area cannot be displayed

If lpszOperand is written several times within lpszContainingBlock or its FB, the entry point is always the first write access of lpszOperand.

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The data was transmitted correctly to the PLC code view. Error messages such as "Operand is not supported" or "Operand not found" are output in the PLC code view and cannot be queried using the function.

FALSE

An error has occurred.

Required files

kopapi.h

kopapi.lib

kopapi.dll

Example

The example shows the embedding of the function call in a user-defined C function. For better maintainability, the call should be swapped out to a global script function in which the name of the PLC station is also determined.

```
BOOL OpenCodeViewerByAssignment(char* screenName, char* objectName, char*
cpuName, char* containingBlock, char* operand)
{
#pragma code("kopapi.dll")
#include "kopapi.h"
#pragma code()
BOOL result;
CMN_ERROR error;
char* serverPrefix = "";
DWORD dwFlags = 0;
result = OpenViewerIECPLByAssignment(dwFlags, serverPrefix, screenName,
objectName, cpuName, containingBlock, operand, &error);
if(!result)
{
  // there are only few reasons why the call to OpenViewerIECPLByAssignment
will
  // fail, in most cases the viewer control could not be found
  // most of the errors have to be handled in the OnError event of the viewer
  printf("OpenViewerIECPLByAssignment failed: err1=%ld, err2=%ld, err3=%ld,
  err4=%ld, err5=%ld, text=\"%s\"\r\n", result, error.dwError1,
error.dwError2,   error.dwError3, error.dwError4, error.dwError5,
error.szErrorText);
}
return result;
}
```

9.2.6 IsJumpableProDiagAlarm

Description

The function returns a Boolean value.

The Boolean value contains the information on whether a jump to the PLC code of the selected alarm is possible.

Definition

```
BOOL WINAPI IsJumpableProDiagAlarm (
    LPCSTR screenName,
    LPCSTR objectName,
    long id,
    LPCMN_ERRORA lpdmError)
```

Parameters

screenName

Pointer to the data of the screen in which the relevant object is configured.

objectName

Pointer to the data with the name of the respective object (alarm)

id

Number of the alarm in the alarm list, not the ID of the alarm.

If several alarms are selected, the value is"0".

lpdmError

Pointer to the data of the extended error message in the CMN_ERROR structure. The system writes error information to this structure if an error occurs.

Return value

TRUE

The jump to the PLC code of the selected alarm is possible.

FALSE

The jump to the PLC code of the selected alarm is not possible.

Required files

kopapi.h

kopapi.lib

kopapi.dll

Example

The example shows the embedding of the function call in a user-defined C function, which is started when an alarm is selected in the alarm view.

```
#include "GlobalDefinitions.h"
void OnSelectedIdChanged(char* screenName, char* objectName, long id) {
    #pragma code("KOPAPI.dll")
    #include "kopapi.h"
    #pragma code()
    CMN_ERROR errorStruct;
    BOOL bResult;

    bResult = IsJumpableProDiagAlarm ( screenName, objectName, id, 
&errorStruct);
    SetPropBOOL ( screenname, "Button_Jump", "Enabled", bResult);
}
```

9.3 Error-handling

General information

Execution of the functions for the network entry is divided in a synchronous part and an asynchronous part. The synchronous part checks the parameters and transfers them to the asynchronous part.

The return value of the functions specifies whether an error has occurred in the asynchronous section of the function. If an error has occurred, the return value is FALSE. These types of errors occur when parameters such as the parameter `lpszErrorTag` are not supplied or supplied incorrectly.

Errors that occur in the asynchronous functions are reported through the lpszErrorTag parameter. Such errors include: Project not found, block does not exist, ... `lpszErrorTag` contains the name of a tag with the String data type. In order to respond to the return value, you have to configure a function that is triggered with the cycle "To change" at a value change of the error tags. The error tag should also be used to prevent further function calls or the operation of a button, for example, as long as the error tag has the value "RUNNING".

lpszErrorTag

Name of a tag of the String data type. If the tag is not required, NULL can be transferred as the parameter. If required, the tag name can include a ServerPrefix.

The value of the error tag is changed as follows:

- When the asynchronous part is started, the value of the error tag is set to "RUNNING".
- When the asynchronous part is completed without an error, the value of the error tag is set to "OK".
- When the asynchronous part was terminated with an error, the error tag contains a multi-line string whose lines are separated by a linefeed character ('\n').

In the case of an error, the error tag has the following structure:

- Line 1: "ERROR"
- Line 2 - Line 6: Decimal numbers, data type: 32 bits unsigned; DWORD
- Line 7: Error text

Error text

When the asynchronous part was terminated with an error, the error tag contains a multi-line string whose lines are separated by a linefeed character ('\n'). The seventh line contains one of the following error texts:

IDS_E_IS_TIA_PROJECT	KOPAPI: This function cannot be used in a TIA Portal project.
IDS_E_NO_TIA_PROJECT	KOPAPI: The function call is disabled because no TIA Portal project is open!
IDS_E_TIAPORTAL_UNKNOWN_FLAGS	dwFlags contains an undefined value.

IDS_E_TIAPORTAL_ERRORTAG_NOT_EXIST	The error tag [%s] does not exist.
IDS_E_TIAPORTAL_PREVIOUS_CALL_IS_RUN-NING	The previous call is still running (RUNNING). Please terminate it first.
IDS_E_TIAPORTAL_CANNOT_WRITE_ERROR-TAG	The error tag [%s] cannot be written.
IDS_E_TIAPORTAL_COMACCESS_REGIS-TERPS_FAILED	No connection to TIA Portal possible.
IDS_E_TIAPORTAL_CANNOT_START_PORTAL	TIA Portal cannot be started.
IDS_E_TIAPORTAL_CANNOT_SEARCH_STAR-TED_PORTAL	A search for a started Portal is not possible.
IDS_E_TIAPORTAL_NO_PORTAL_STARTED	A Portal could not be started.
IDS_E_TIAPORTAL_EXCEPTION_SYNC_PART	Exception in synchronous processing part.
IDS_E_TIAPORTAL_EXCEPTION_ASYNC_PART	Exception in asynchronous processing part.
IDS_E_TIAPORTAL_NOT_INSTALLED	No TIA Portal installed.
IDS_E_TIAPORTAL_PROJECT_CANNOT_OPEN	Project [%s] cannot be opened.
IDS_E_TIAPORTAL_AL-READY_OPENED_WITH_OTHER_PROJECT	TIA Portal is opened with another project.
IDS_E_TIAPORTAL_CREATECOMMAND	CreateCommand error.
IDS_E_TIAPORTAL_COMMAND_ADDARGU-MENT	Command AddArgument error.
IDS_E_TIAPORTAL_EXECUTECOMMAND	Serious error in ExecuteCommand.
IDS_E_TIAPORTAL_COMMAND_ERROR	TIA project command error: [%s, %ld].
IDS_E_TIAPORTAL_COMMAND_UNKNOWN	TIA project unknown error: [%s, %ld].

Index