Smart People. Expert Solutions.

SIEMENS OPEN LIBRARY

8 – SIMATIC Visualization Architect (SiVArc)

JUNE 11, 2019

Contents

1. Purpose

The purpose of this document is to detail the SIMANTIC Visualization Architect (SiVArc) properties and rules associated with the Siemens Open Library Function Blocks and Faceplates. It should be used as a reference when implementing the SiVArc functionality of any of the blocks.

2. Intended Use

This document is intended to be used by anyone utilizing the Open Library for PLC and HMI Development. This document should be used after reviewing the following documents:

1. 1- Siemens Open Library – Library Overview and Architecture
2. 2- Siemens Open Library – Initial Setup
3. 3- Siemens Open Library – Example Object Configuration
4. 4- Siemens Open Library – Detailed Block Overview

3. Revision History

Version	Date	Author	Comments
2.0	2017-11-3	DMC	Initial Release
3.0	2018-12-05	DMC	No Changes
4.0	2019-06-11	DMC	Include WinCC Professional

4. Open Library License

5. SiVArc in the Open Library

SiVArc is a plug-in tool for TIA Portal V14 and later that allows for the automatic creation of screens and screen objects using the function blocks in the control program, master copy screen templates, and screen rules. The open library includes screen rules and master screens to allow SiVArc to be used with all the device and most of the process objects in the library. These can be found under Master Copies > SiVArc in the library. Using the included screen rules, screens are created for each system in the program, and all the icons from that system populate to the screen. In WinCC Comfort/Advanced, pop-up screens are created for each instance of the function block, and in WinCC Professional, tag prefixes are specified to load the right instance into the pop-up. To determine which icon or popup faceplate is used with each instance of the function block, in cases where there is more than one option, condition statements looking at the network title for the network on which the instance of the block is called are used.

The example project has the SiVArc functionality integrated and the WaterTreatmentHMI_SiVArc and WaterTreatmentSCADA_SiVArc devices show the auto-generation produced by the control program and screen rules from the library. One thing to note is that the standard set of SiVArc rules will not generate the multiplexed valve system in the example project correctly.

6. Using SiVArc with the open library

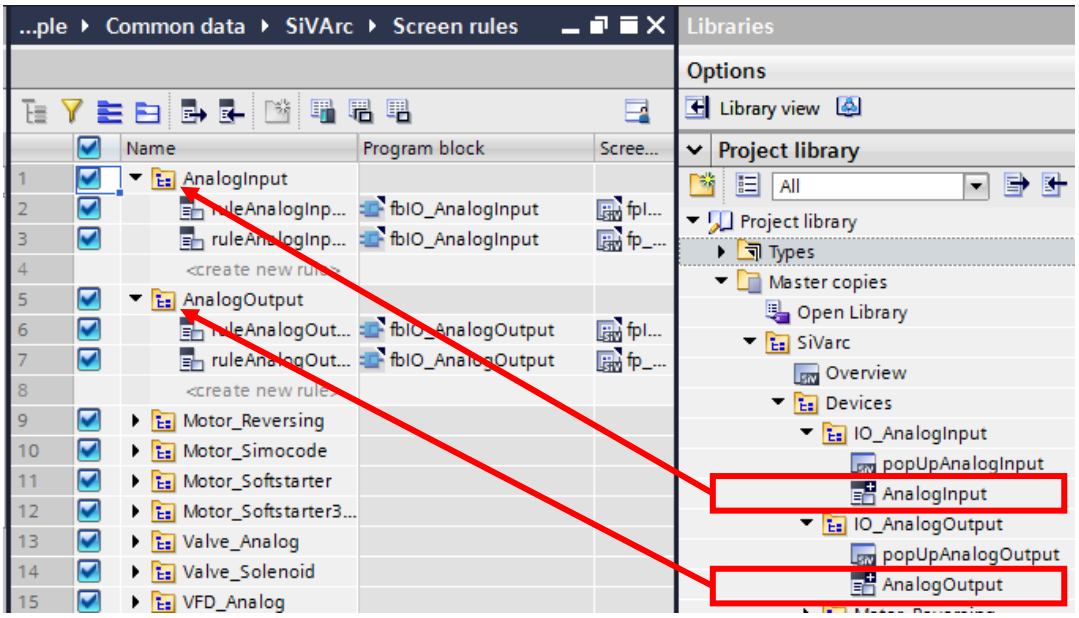
This section provides a checklist of things to do when adding SiVArc functionality to the project.

6.1. Make sure the SiVArc plug-in is installed and compatible with the project.

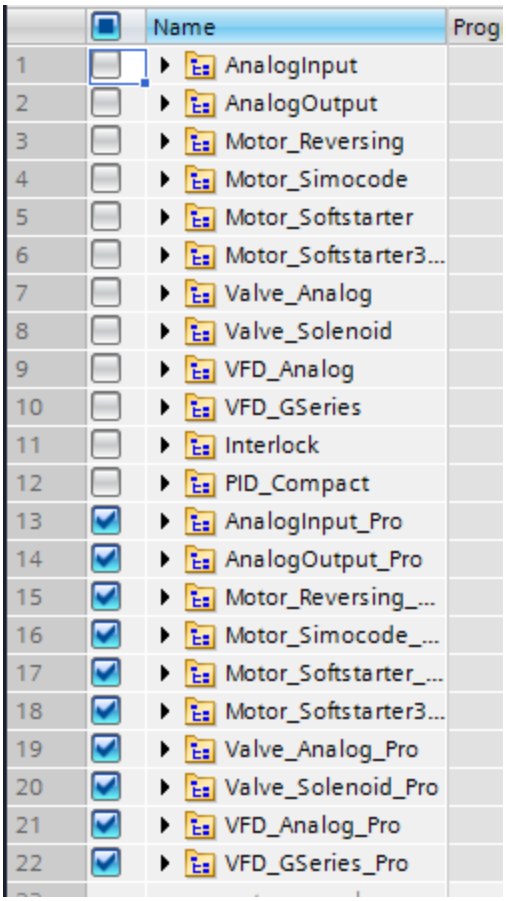
The SiVArc plug-in requires a license; a trial (21 day) license can be found on the Siemens website (PN: 6AV2107-0PX04-0AA7) . SiVArc is only compatible with TIA Portal V14 and later, and is only compatible with a limited set of PLC's. The S7-1200 and S7-1500 are included in this set, the S7-300 and S7-400 are not included.

6.2. Add Screen Rules.

The screen rules from the library can be added by dragging them to the "Screen Rules" page found under Common Data>SiVArc in the project tree. An example of this is shown below.



Rules for both WinCC Comfort/Advanced and WinCC Professional devices are included in the library; make sure to bring in the ones that match the desired implementation. If both WinCC Comfort/Advanced and WinCC Professional are in one project, it is important to select only the rules applicable to the platform prior to generating the visualization. Otherwise, two sets of screens will be generated. This can be accomplished by checking the boxes next to only the desired rules. An example is shown below with only the WinCC Professional rules active.

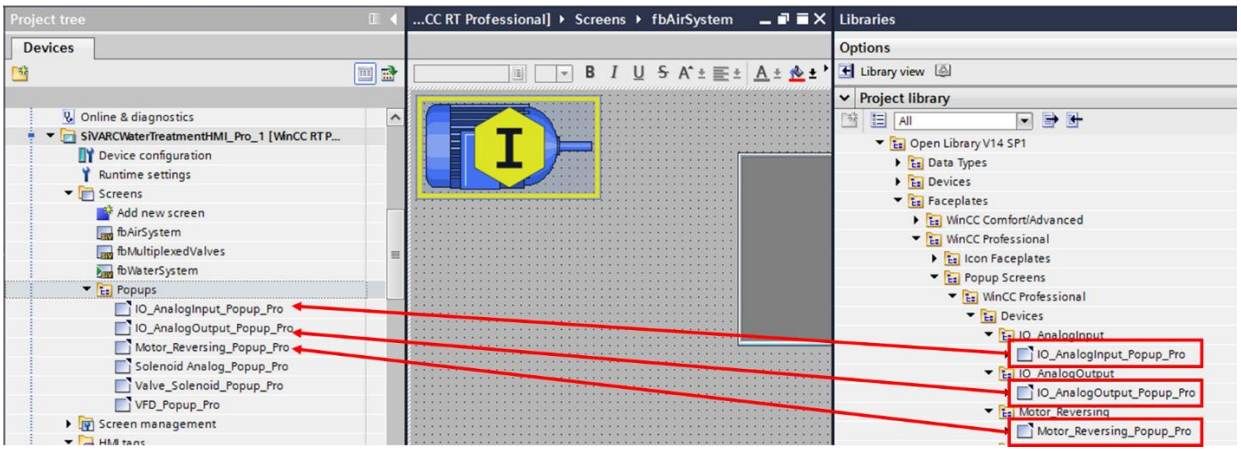


		Name	Prog
1	☐	AnalogInput	
2	☐	AnalogOutput	
3	☐	Motor_Reversing	
4	☐	Motor_Simocode	
5	☐	Motor_Softstarter	
6	☐	Motor_Softstarter3...	
7	☐	Valve_Analog	
8	☐	Valve_Solenoid	
9	☐	VFD_Analog	
10	☐	VFD_GSeries	
11	☐	Interlock	
12	☐	PID_Compact	
13	☑	AnalogInput_Pro	
14	☑	AnalogOutput_Pro	
15	☑	Motor_Reversing_...	
16	☑	Motor_Simocode_...	
17	☑	Motor_Softstarter_...	
18	☑	Motor_Softstarter3...	
19	☑	Valve_Analog_Pro	
20	☑	Valve_Solenoid_Pro	
21	☑	VFD_Analog_Pro	
22	☑	VFD_GSeries_Pro	

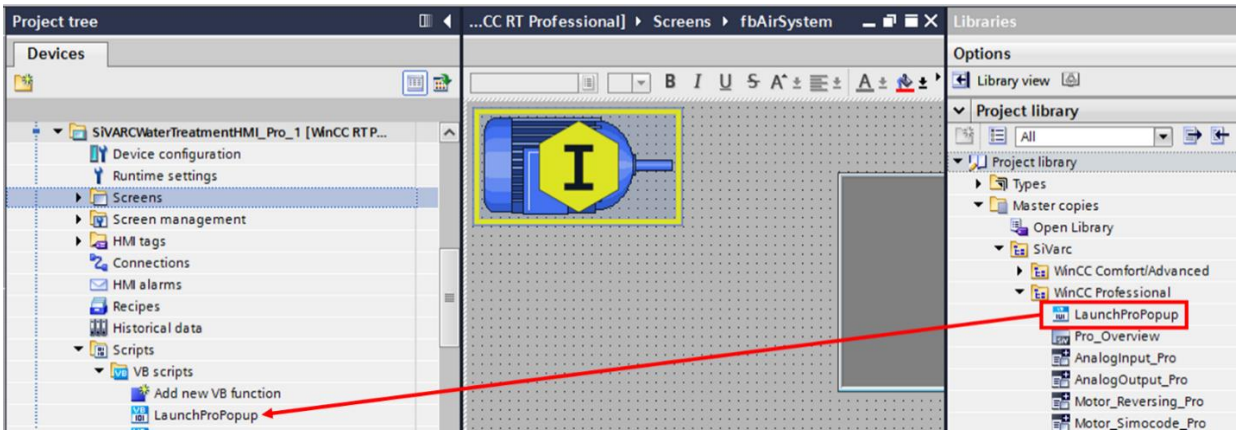
Other screen rules can be added to customize the project as desired; make sure the types of program block and screen object are selected from the project library if you want all blocks of a type to autogenerate the visualization.

6.3. Additional steps for WinCC Professional

As the WinCC Professional implementation in the library uses a different method for pop-ups, a few more things need to be added to the project from the library. First, the pop-up screens for all objects that are used need to be added. This can be done by dragging them from the project library, and it is recommended to put them in a folder for organization purposes.

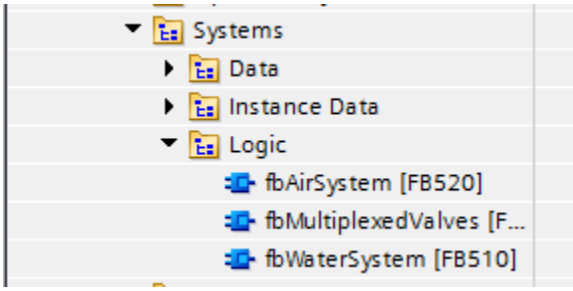


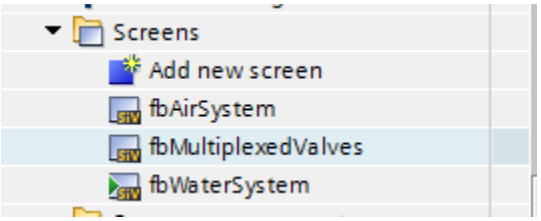
A script also needs to be pulled in from the library to the VB scripts folder for the appropriate display. The script is used on all of the WinCC Professional device icons to launch the pop-up with the right tag prefixes.



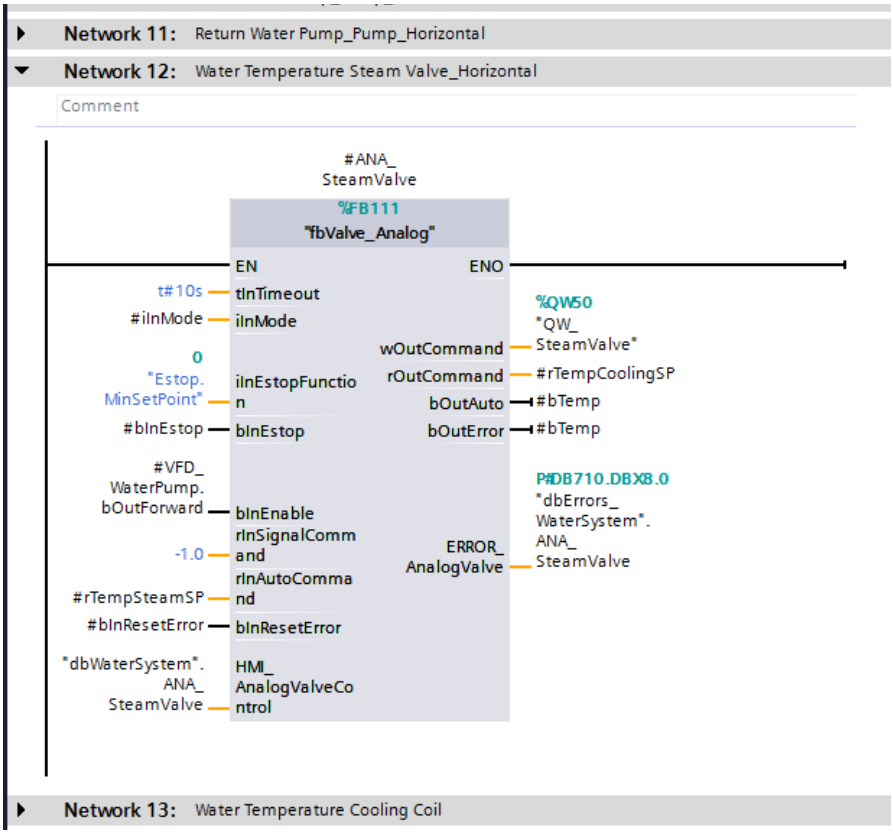
6.4. Check Organization and Network naming.

Check to make sure the organization of the program lines up with the desired screens. Per the rules included in the library, a screen is created for each system block that calls library objects. So, for example, the example project has 3 system blocks, and creates three screens to line up with these.



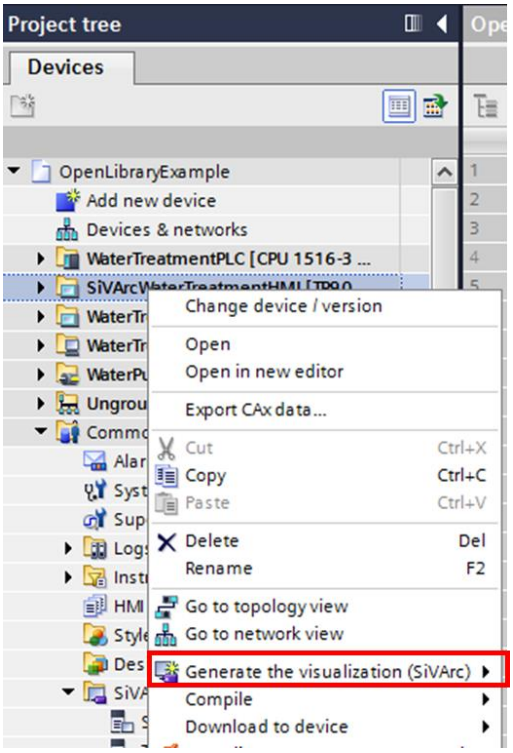


Also check the network title anywhere a function block from the library is called, and make sure it matches the format described for the library object in the document “4-Detailed Block Descriptions”. The network title is used to define titles for pop-ups, as well as to define which icons get populated per the screen rules. In select cases with WinCC Professional, it also determines which pop-up is used. An example of a block call with a corresponding network title:



6.5. Generate the Visualization.

To generate the screens from the control program, right click on the HMI where you would like the visualization to be populated, and select “Generate the visualization (SiVArc)” > “Generate the visualization” (shown below). This can be done many times throughout the development of the program; the screens will be refreshed each time the generation is selected. How the generation will refresh is dependent on the layout defined by the master copy of a screen and layout fields used in rules. For the open library rules, layout fields are only defined for pop-ups, so non-pop-up icons will be refreshed at whatever location they were last moved to.



7. Faceplate SiVArc properties

SiVArc in the library is implemented for all device objects in both WinCC Comfort/Advanced and WinCC Professional, and for some process objects in WinCC Comfort/Advanced. All the faceplates have names, HMI_*Devicetype* tags, and events defined through SiVArc. The pop-up faceplates also have titles defined through SiVArc. This section covers how each of these properties is defined.

7.1. Name

The name of the faceplate is determined from the user assigned name of the instance of the function block it is associated with. Occurences of "ins" and "Ins" are removed from the name of the data block, and the name is preceded with fp. This is important because it allows more than one of each type of faceplate to be populated to a screen.

Example:

insSOL_Overflow → fpSol_Overflow

7.2. HMI_*Devicetype* Tag connection

The HMI tag referenced by the HMI_*Devicetype* property of the faceplate is defined by taking the PLC tag of udtHMI_*Devicetype* used by the function block and replacing the "." with the separator character defined by "TagNaming.SeperatorChar". If this HMI tag doesn't already exist when the visualization is generated, it will be created and connected to PLC tag of udtHMI_*Devicetype*.

Example:

HMIData.SOL_Overflow → HMIData_SOL_Overflow

7.3. Pop-up Title

The title of the pop-up is determined from the first part (i.e. the part before an underscore) of the title of the network where the instance of the function block is called. If the network title isn't defined, a default title related to the type being called is used. Example:

The network is titled "Overflow Valve_Horizontal" → The pop-up title is "Overflow Valve"

The network isn't titled → The popup title is "Solenoid Valve ##"

7.4. Events

Events on screen objects work differently in the WinCC Comfort/Advanced and WinCC Professional implementation, due to the difference in how pop-ups are set up to work in the library for each of these platforms.

7.4.1. WinCC Comfort/Advanced

The event on the icon faceplates opens the associated pop-up when the icon is clicked, and the event on the pop-up closes itself when the close button is released. These events assume the name of the pop-up screen is "popup" followed by the user defined name of the associated instance of the function block. The pop-up screens will auto-generate according to this name convention. Example:

insSOL_Overflow → popupinsSOL_Overflow

7.4.2. WinCC Professional

An event on the icon faceplates loads the pop-up for the corresponding type with tag prefixes for the right instance into the screen window specified on the master copy of the screen. The event uses the LaunchProPopup script, included in the library, with the tag connection specified as in 7.2, and the title specified as in 7.3. The event assumes that the pop-up screen names have not been modified from how they appear in the library.

---

8. Screen Rules

This section covers the different parts of screen rules and how the rules saved to the library are implemented.

8.1. Components

Each screen rule is named, and includes a Program Block, a Screen Object, and a Master Copy of a Screen. Optionally, a Layout Field and Condition can also be specified.

8.1.1. Program Block

The Program Block field specifies the type that, when used in the control program, will create a visualization automatically. Each program block can have multiple rules associated with it. For example, there might be one rule to create an icon, and another to create a popup screen. Any function block in the project can be specified in this field, but it is recommended to specify the function block type in the project library, as then the rule will run for any instance of that type in the project.

8.1.2. Screen Object

The Screen Object field specifies the visualization to be created. A single screen object can be used in more than one rule, for instance if there is more than one function block that could be used with a single icon (e.g. Analog and Gseries VFD's). Screen object types are specified from the project library, and it is important that they have SiVArc properties associated with them. All the faceplates for S7-1200/1500 on the WinCC Comfort/Advanced in the library have SiVArc properties defined.

8.1.3. Master Copy of a Screen

The Master Copy of a Screen field specifies to which screen a screen object gets populated. For instance, this would specify whether it should go to the screen for that system, or a pop up screen. Layout fields can further specify location. Master copies of screens can be used for any number of rules, and are called from the project library. In the open library, an overview screen and pop-up screens for each type are available. The Overview master screen creates a screen for each block that calls the library function blocks. For example, the Example Project has three system function blocks that then call blocks to which the screen rules apply; each system function block creates a screen when the generation is run.

8.1.4. Layout Field

The Layout Field is an optional component of a screen rule, and specifies where on the screen an object should be populated. Layout fields are required for pop-up screens. These fields are specified on the Master Copies of screens by drawing and labeling a rectangle. If a layout field is defined, the screen object will always be generated into the location specified by the layout field. If a layout field isn't defined, the screen object will be initially generated to grid on the specified screen, then regenerated wherever it gets moved to on that screen by the user. The open library has layout fields defined for each pop-up screen Master Copy, and no layout fields specified on the overview screen.

8.1.5. Condition

The Condition field is an optional part of a screen rule, and allows the user to specify a true/false statement that is evaluated for each instance of the function block and determines whether the screen rule creates a visualization. This is useful if a type of function block can have more than one icon type. For instance, a VFD function block could control a motor or a pump, or you might want the icon to display in either a horizontal or vertical orientation. In the rules defined in the open library, the valve, motor, VFD, PID, and interlock types have conditions specified. For the Interlock

type, which popup is used is determined based on whether more than 8 inputs to the interlock are assigned. For all the other types, the condition looks at the network title where the instance of the function block is called. For the valve and PID types, the second part of the network title (after the first underscore) is looked at, and for the motor and VFD types, both the second and third part of the network title are considered. How these conditions are implemented is covered in more detail below.

8.2. In the Library

The sets of rules for each type of library object can be found under Master copies> SiVarc> Devices> *Devicetype* and Master copies>Sivarc>Process>*Type* folders. These rules can be added to the project by dragging the screen rule item from each *Type* folder into the screen rules page, found under Common Data > SiVArc in portal. Screen rules exist for all the faceplates except System Control. As there is no function block associated with the System Control object, rules cannot be created to make the faceplates autogenerate.

Below is a table of all the rules in the library as the appear in the screen rules screen in Portal. More information about how the rules apply to each type can be found in the detailed block descriptions.

8.2.1. WinCC Comfort/Advanced

Name	Program block	Screen object	Master copy of a screen	Layout field
Analog_Input				
ruleAnalogInputPopUp	fbIO_AnalogInput	fpIO_AnalogInput_Popup	popUpAnalogInput	analogInput
ruleAnalogInputIcon	fbIO_AnalogInput	fp_AnalogInput_Numeric	Overview	
Analog_Output				
ruleAnalogOutputPopUp	fbIO_AnalogOutput	fpIO_AnalogOutput_Popup	popUpAnalogOutput	analogOutput
ruleAnalogOutputIcon	fbIO_AnalogOutput	fp_AnalogOutput	Overview	
Motor_Reversing				
ruleMotorReversingPopUp	fbMotor_Reversing	fpMotor_Reversing_Popup	popUpMotorReversing	motorReversing
ruleMotorReversingPumpHorizontal	fbMotor_Reversing	fpMotor_Pump_Horizontal	Overview	
ruleMotorReversingPumpVertical	fbMotor_Reversing	fpMotor_Pump_Vertical	Overview	
ruleMotorReversingMotorHorizontal	fbMotor_Reversing	fpMotor_Motor_Horizontal	Overview	
ruleMotorReversingMotorVertical	fbMotor_Reversing	fpMotor_Motor_Vertical	Overview	
Motor_Simocode				
ruleSimocodePopUp	fbMotor_Simocode	fpMotor_Simocode_Popup	popUpMotorSimocode	motorSimocode
ruleSimocodePumpHorizontal	fbMotor_Simocode	fpMotor_Pump_Horizontal	Overview	
ruleSimocodePumpVertical	fbMotor_Simocode	fpMotor_Pump_Vertical	Overview	
ruleSimocodeMotorHorizontal	fbMotor_Simocode	fpMotor_Motor_Horizontal	Overview	
ruleSimocodeMotorVertical	fbMotor_Simocode	fpMotor_Motor_Vertical	Overview	
Motor_Softstarter				
ruleSoftstarterPopUp	fbMotor_SoftStarter	fpMotor_SoftStarter_Popup	popUpMotorSoftstarter	motorSoftstarter
ruleSoftstarterPumpHorizontal	fbMotor_SoftStarter	fpPump_SoftStarter_Horizontal	Overview	
ruleSoftstarterPumpVertical	fbMotor_SoftStarter	fpPump_SoftStarter_Vertical	Overview	
ruleSoftstarterMotorHorizontal	fbMotor_SoftStarter	fpMotor_SoftStarter_Horizontal	Overview	
ruleSoftstarterMotorVertical	fbMotor_SoftStarter	fpPump_SoftStarter_Vertical	Overview	
Motor_Softstarter3RW44				
ruleSoftstarter3RW44PopUp	fbMotor_SoftStarter_3RW44	fpMotor_SoftStarter_3RW44__Popup	popUpMotorSoftstarter3RW44	motorSoftstarter3RW
ruleSoftstarter3RW44PumpHorizontal	fbMotor_SoftStarter_3RW44	fpPump_SoftStarter_3RW44_Horizontal	Overview	
ruleSoftstarter3RW44PumpVertical	fbMotor_SoftStarter_3RW44	fpPump_SoftStarter_3RW44_Vertical	Overview	
ruleSoftstarter3RW44MotorHorizontal	fbMotor_SoftStarter_3RW44	fpMotor_SoftStarter_3RW44_Horizontal	Overview	
ruleSoftstarter3RW44MotorVertical	fbMotor_SoftStarter_3RW44	fpMotor_SoftStarter_3RW44_Vertical	Overview	
Valve_Analog				
ruleValveAnalogPopUp	fbValve_Analog	fpValve_Analog_Popup	popUpValveAnalog	valveAnalog
ruleValveAnalogVertical	fbValve_Analog	fpValve_Analog_Vertical	Overview	
ruleValveAnalogHorizontal	fbValve_Analog	fpValve_Analog_Horizontal	Overview	
ValveSolenoid				
ruleValveSolenoidPopUp	fbValve_Solenoid	fpValve_Solenoid_Popup	popUpSolenoidValve	fpSol
ruleValveSolenoidVertical	fbValve_Solenoid	fpValve_Solenoid_Vertical	Overview	
ruleValveSolenoidHorizontal	fbValve_Solenoid	fpValve_Solenoid_Horizontal	Overview	
VFD_Gseries				
ruleVFDGSeriesPopUp	fbVFD_GSeries	fpVFD_Popup	popUpVFDGSeries	vfdGSeries
ruleVFDGSeriesPumpHorizontal	fbVFD_GSeries	fpVFD_Pump_Horizontal	Overview	
ruleVFDGSeriesPumpVertical	fbVFD_GSeries	fpVFD_Pump_Vertical	Overview	
ruleVFDGSeriesMotorHorizontal	fbVFD_GSeries	fpVFD_Motor_Horizontal	Overview	
ruleVFDGSeriesMotorVertical	fbVFD_GSeries	fpVFD_Motor_Vertical	Overview	
VFD_Analog				
ruleVFDAnalogPopUp	fbVFD_Analog	fpVFD_Popup	popUpVFDGSeries	vfdGSeries
ruleVFDAnalogPumpHorizontal	fbVFD_Analog	fpVFD_Pump_Horizontal	Overview	
ruleVFDAnalogPumpVertical	fbVFD_Analog	fpVFD_Pump_Vertical	Overview	
ruleVFDAnalogMotorHorizontal	fbVFD_Analog	fpVFD_Motor_Horizontal	Overview	
ruleVFDAnalogMotorVertical	fbVFD_Analog	fpVFD_Motor_Vertical	Overview	
Interlock				
ruleInterlock8PopUp	fbInterlock	fpInterlock8_Popup	popUpInterlock8	interlock
ruleInterlock16PopUp	fbInterlock	fpInterlock16_Popup	popUpInterlock16	interlock
ruleInterlockIcon	fbInterlock	fpInterlock	Overview	
PID_Compact				
rulePIDwTrend	fbPID_Compact	fpPID_Compact_Popup_Graph	popUpPidCompactTrend	PID_Compact_Trend
rulePID	fbPID_Compact	fpPID_Compact_Popup	popUpPidCompact	POD_Compact

	Condition
	Analog_Input
AND	
AND	
	Analog_Output
AND	
AND	
	Motor_Reversing
AND	
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Motor_Simocode
AND	
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Motor_Softstarter
AND	
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Motor_Softstarter3RW44
AND	
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Valve_Analog
AND	
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Vertical", True, False), True)
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Horizontal", True, False), True)
	Valve_Solenoid
AND	
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Vertical", True, False), True)
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Horizontal", True, False), True)
	VFD_Gseries
AND	
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	VFD_Analog
AND	
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Interlock
AND	Not Block.Parameters("bInInterlock09").Assigned
AND	Block.Parameters("bInInterlock09").Assigned
AND	
	PID_Compact
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "TREND", True, False), False)
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "TREND", False, True), True)

8.2.2. WinCC Professional

Name	Program block	Screen object	Master copy of a screen	Layout field
AnalogInput_Pro				
ruleAnalogInput_Pro	fbIO_AnalogInput	fp_AnalogInput_Numeric_Pro	Pro_Overview	
AnalogOutput_Pro				
ruleAnalogOutputIcon_Pro	fbIO_AnalogOutput	fp_AnalogOutput_Pro	Pro_Overview	
Motor_Reversing_Pro				
ruleMotorReversingPumpHorizontal_Pro	fbMotor_Reversing	fpMotor_Pump_Horizontal_Pro	Pro_Overview	
ruleMotorReversingPumpVertical_Pro	fbMotor_Reversing	fpMotor_Pump_Vertical_Pro	Pro_Overview	
ruleMotorReversingMotorHorizontal_Pro	fbMotor_Reversing	fpVFD_Motor_Horizontal_Pro	Pro_Overview	
ruleMotorReversingMotorVertical_Pro	fbMotor_Reversing	fpMotor_Motor_Vertical_Pro	Pro_Overview	
Motor_Simocode_Pro				
ruleSimocodePumpHorizontal_Pro	fbMotor_Simocode	fpMotor_Pump_Horizontal_Pro	Pro_Overview	
ruleSimocodePumpVertical_Pro	fbMotor_Simocode	fpMotor_Pump_Vertical_Pro	Pro_Overview	
ruleSimocodeMotorHorizontal_Pro	fbMotor_Simocode	fpVFD_Motor_Horizontal_Pro	Pro_Overview	
ruleSimocodeMotorVertical_Pro	fbMotor_Simocode	fpMotor_Motor_Vertical_Pro	Pro_Overview	
Motor_Softstarter_Pro				
ruleSoftstarterPumpHorizontal_Pro	fbMotor_SoftStarter	fpPump_SoftStarter_Horizontal_Pro	Pro_Overview	
ruleSoftstarterPumpVertical_Pro	fbMotor_SoftStarter	fpPump_SoftStarter_Vertical_Pro	Pro_Overview	
ruleSoftstarterMotorHorizontal_Pro	fbMotor_SoftStarter	fpMotor_SoftStarter_Horizontal_Pro	Pro_Overview	
ruleSoftstarterMotorVertical_Pro	fbMotor_SoftStarter	fpMotor_SoftStarter_Vertical_Pro	Pro_Overview	
Motor_Softstarter3RW44_Pro				
ruleSoftstarter3RW44PumpHorizontal_Pro	fbMotor_SoftStarter_3RW44	fpPump_SoftStarter_3RW44_Horizontal_Pro	Pro_Overview	
ruleSoftstarter3RW44PumpVertical_Pro	fbMotor_SoftStarter_3RW44	fpPump_SoftStarter_3RW44_Vertical_Pro	Pro_Overview	
ruleSoftstarter3RW44MotorHorizontal_Pro	fbMotor_SoftStarter_3RW44	fpMotor_SoftStarter_3RW44_Horizontal_Pro	Pro_Overview	
ruleSoftstarter3RW44MotorVertical_Pro	fbMotor_SoftStarter_3RW44	fpMotor_SoftStarter_3RW44_Vertical_Pro	Pro_Overview	
Valve_Analog_Pro				
ruleValveAnalogVertical_Pro	fbValve_Analog	fpValve_Analog_Vertical_Pro	Pro_Overview	
ruleValveAnalogHorizontal_Pro	fbValve_Analog	fpValve_Analog_Horizontal_Pro	Pro_Overview	
Valve_Solenoid_Pro				
ruleValveSolenoidVertical_Pro	fbValve_Solenoid	fpValve_Solenoid_Vertical_Pro	Pro_Overview	
ruleValveSolenoidHorizontal_Pro	fbValve_Solenoid	fpValve_Solenoid_Horizontal_Pro	Pro_Overview	
VFD_Analog_Pro				
ruleVFDAnanlogPumpHorizontal_Pro	fbVFD_Analog	fpVFD_Pump_Horizontal_Pro	Pro_Overview	
ruleVFDAnalogPumpVertical_Pro	fbVFD_Analog	fpVFD_Pump_Vertical_Pro	Pro_Overview	
ruleVFDAnalogMotorHorizontal_Pro	fbVFD_Analog	fpVFD_Motor_Horizontal_Pro	Pro_Overview	
ruleVFDAnalogMotorVertical_Pro	fbVFD_Analog	fpVFD_Motor_Vertical_Pro	Pro_Overview	
VFD_GSeries_Pro				
ruleVFDGSeriesPumpHorizontal_Pro	fbVFD_GSeries	fpVFD_Pump_Horizontal_Pro	Pro_Overview	
ruleVFDGSeriesPumpVertical_Pro	fbVFD_GSeries	fpVFD_Pump_Vertical_Pro	Pro_Overview	
ruleVFDGSeriesMotorHorizontal_Pro	fbVFD_GSeries	fpVFD_Motor_Horizontal_Pro	Pro_Overview	
ruleVFDGSeriesMotorVertical_Pro	fbVFD_GSeries	fpVFD_Motor_Vertical_Pro	Pro_Overview	

	Condition
	AnalogInput_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Analog") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Input")
	AnalogOutput_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Output") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Icon")
	Motor_Reversing_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Motor_Simocode_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Motor_Softstarter_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Motor_Softstarter3RW44_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	Valve_Analog_Pro
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Vertical", True, False), True)
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Horizontal", True, False), True)
	Valve_Solenoid_Pro
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Vertical", True, False), True)
AND	If(IsDefined("Block.NetworkTitle[1]"), If(Block.NetworkTitle[1] = "Horizontal", True, False), True)
	VFD_Analog_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
	VFD_GSeries_Pro
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Pump") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Horizontal")
AND	IsDefined("Block.NetworkTitle[1]") And (Block.NetworkTitle[1] = "Motor") And IsDefined("Block.NetworkTitle[2]") And (Block.NetworkTitle[2] = "Vertical")