DMC

Smart People. Expert Solutions.

SIEMENS OPEN LIBRARY

4 – Detailed Blocks Overview

JUNE 11, 2019

Contents

1. Purpose

This document details the inputs, outputs, user defined types, and functionality of the Siemens Open Library Function Blocks and HMI Faceplates. It should be used as a reference when using or configuring 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

3. Revision History

Version	Date	Author	Comments
1.0	2016-05-23	DMC	Initial Release
1.1	2016-06-20	DMC	Updates to Faceplates, function blocks, and datatypes
1.2	2016-08-23	DMC	Updating information regarding S7-1200 use of the GSeries VFD.
1.3	2016-10-11	DMC	No Changes
2.0	2017-11-3	DMC	Added SiVArc properties to block descroptions.
3.0	2018-12-5	DMC	Added documentation for many new supported blocks.
4.0	2019-6-11	DMC	Added simulation and other block interface updates.

4. Open Library License

5. Section Overview

This section provides a general overview of the library objects and what it contains.

5.1. Function Block

Each library object will have a Function block associated with it. This function block will have various inputs and outputs for the control of the device. Each block has a unique interface which will be outlined below. If the functionality of the block does not meet the requirements of your system, then it can be used as a starting point for developing custom libraries.

5.2. HMI_*Devicetype* User Defined Type

Each function block will have an In/Out UDT for the HMI interface and will be named HMI_*DeviceType*. This UDT serves as the interface with the HMI and can be used to control logic in the PLC. Each function block is connected to a specific version of the UDT, so if items are added to the UDT for custom development, the Function Block and HMI objects will need to be updated to reference the new UDT version.

A Global Data Block variable will need to be created with this UDT for each instance of the device type function block.

5.3. ERROR_*Devicetype* User Defined Type

Each function block that has errors will pass out a UDT named ERROR_*DeviceType*. A Global Data Block variable will need to be created with this UDT for each instance of the device type function block. Error UDT variables are intended to be grouped with other error structures and errors so that the data block can be used by an Excel macro to automatically generate alarms for the HMI. Errors inside this UDT should be used for appropriate PLC logic, and is the easiest way to interface to the alarms from a given device.

5.4. HMI Icon Faceplate

Most objects have an associated Icon Faceplate in the library. The Icon faceplate provides basic information on the main HMI screen and will contain images of objects like valves, pumps, and motors. Many of these objects utilize the built-in WinCC Symbol Library, so if different appearances are required for an application, it is quick to duplicate the type, rename it, and change the visual appearance of the object. Please note that the appearance for new objects will need to be set to 'Shaded' or 'Solid' in order for the color animations to function.

5.5. HMI Pop-up Faceplate

Each object has an associated pop-up faceplate in the library, intended for use with WinCC Comfort/Advanced. Each device in the programmed system will need to be configured with a separate Pop-up to prevent the need to multiplex tags. In addition to pop-up screens, these faceplates can be displayed on any screen desired. Custom versions of the pop-ups can also be created to add or remove functionality or diagnostic information as desired.

5.6. WinCC Professional Pop-up Screen

Each object has an associated pop-up screen in the library, intended for use with WinCC Professional. WinCC Pro doesn't support multiplexing of tags, so to prevent needing to have an instance of a pop-up for every corresponding icon faceplate, screens windows and tag prefixes are used. This allows one pop-up screen to be used for each type, where the correct instance of that type is loaded when the pop-up opens by setting the tag prefix.

5.7. SIMANTIC Visualization Architect (SiVArc) Properties

SiVArc is a plug-in that allows the automatic generation of screens and screen objects from a control program. For more information reference "8 – SIMANTIC Visualization Architect (SiVArc)". Each object has SiVArc properties

associated with the HMI Icon Faceplate and HMI Pop-up Faceplate. These define the name of the faceplates, the interface tag connection, the events to open and close the pop-ups, and title of the pop-up when SiVArc is being used to create screens.

5.8. SiVArc Screen Rules

Most objects also have SiVArc rules associated with them, which control the creation of icons and pop-ups during auto-generation. These rules reference the network title on networks where library objects are called to determine which faceplate should be used. All the rules assume the network title is going to be separated into fields by underscores, giving a network title that has the format:

NetworkTitle[0]_NetworkTitle[1]_NetworkTitle[2]_NetworkTitle[3]

Where NetworkTitle[0],[1], and [2] are any strings that don't contain underscores. NetworkTitle[0] is always the title for the pop-up associated with the instance of the function block. NetworkTitle[1], NetworkTitle[2], and NetworkTitle[3] are used by the screen rules associated with some library objects. How to name network titles to be compatible with SiVArc auto-generation is covered for each block in this document. An explanation of how to implement SiVArc in a project, as well as in-depth explanation of screen rules, can be found in "8 – SIMANTIC Visualization Architect (SiVARC)".

5.9. Device Simulation

Starting in version 4.0, released for TIA Portal V15, many of the device function blocks in the Open Library include a built-in software simulation feature. Open Library simulation allows device blocks to respond in a no-error, running state without controlling physical outputs or requiring physical input feedback. Simulation is not controllable via the HMI interface and must be toggled using the block interface on the PLC. See Document 9 (Device Simulation) for more details.

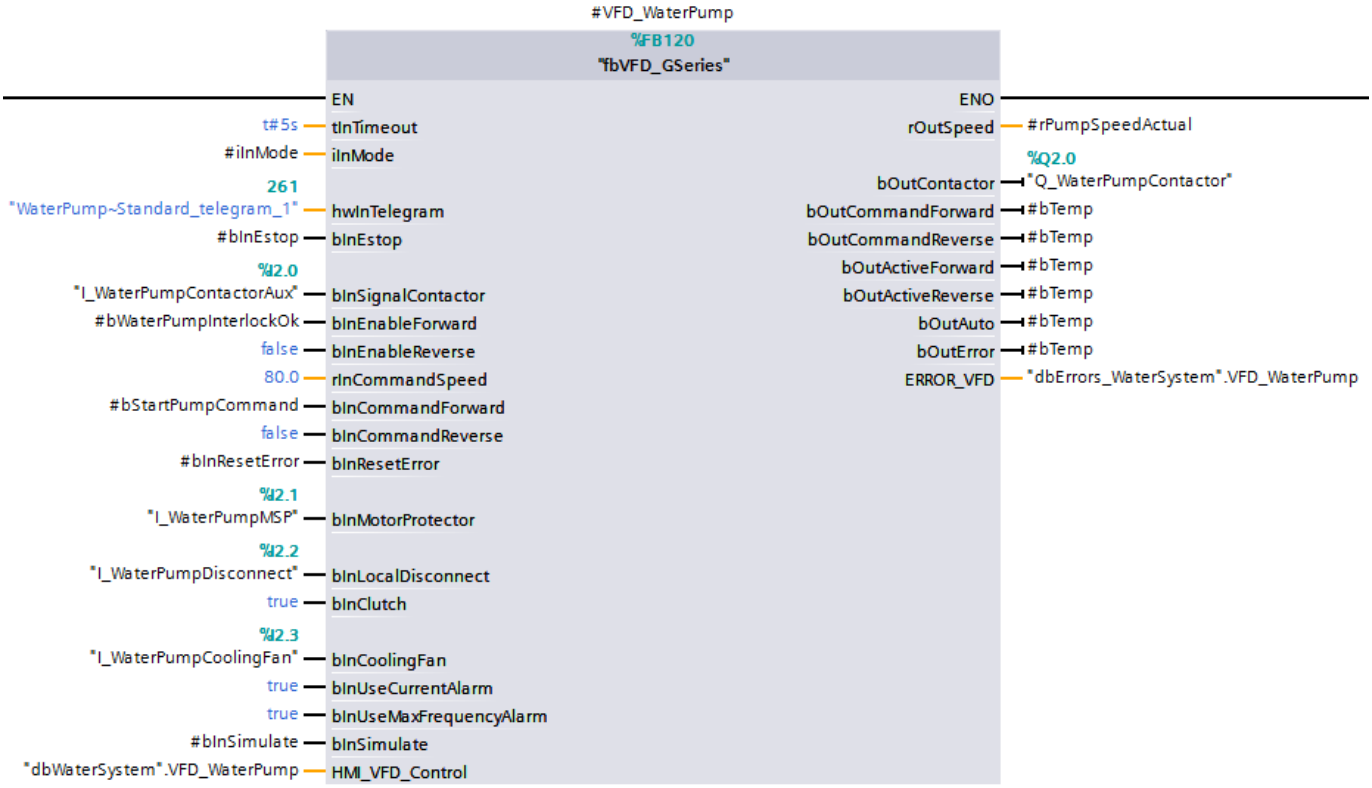
6. Devices

This section covers all objects that are in the Devices Folder of the open library.

6.1. G Series Motor Control – fbVFD_GSeries

6.1.1. Description

The G Series VFD Control Function Block controls Siemens G Series VFD or Micromaster VFDs using Standard Telegram 1. A technology object is not used by this block, which allows for a lower CPU requirement for each drive and avoids technology object count limitations. It utilizes Standard Telegram 1 and has been tested on a G120, however, it will function with any drive using Standard Telegram 1 on either Profibus or Profinet.



6.1.2. Function Block Interface

6.1.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input

hwInTelegram	HW_SUBMODULE	The system constant that connects to the Telegram 1 configuration of the VFD
bInSignalContactor	Bool	Contactor feedback. Map the Output bOutContactor if no Contactor exists.
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
rInCommandSpeed*	Real	Speed set point for automatic mode in percent. (0.0-100.0%)
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will run in this direction until the reverse interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be mapped directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInClutch	Bool	This input should be wired directly into the motor's clutch overload input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInCoolingFan	Bool	This input is used to tell the VFD that the cooling fan is running, if available. If not available, this input should be set to true. When false for 30 seconds after VFD starts, the motor is disabled and an error is triggered
bInUseCurrentAlarm	Bool	This input enables or disables an alarm for drive overcurrent. If an overcurrent alarm should be used, then this input should be true. If the current alarm is to be disabled, then this input should be false, or left unwired
bInUseMaxFrequency	Bool	This input enables or disables an alarm for max frequency. If a frequency alarm should be used, then this input should be true. If the frequency alarm is to be disabled, then this input should be false, or left unwired

bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.1.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_VFD_Control	udtHMI_VFD_Control	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also, inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

6.1.2.3. Out Parameters

Output Variables	Type	Description
bOutContactor	Bool	Contactor output. Map to Output to turn on contactor if used
bOutCommandForward	Bool	Motor commanded forward
bOutCommandReverse	Bool	Motor commanded reverse
bOutActiveForward	Bool	Motor running forward feedback from VFD
bOutActiveReverse	Bool	Motor running reverse feedback from VFD
rOutSpeed	Real	Actual speed from VFD
bOutError	Bool	This output indicates whether there's an error condition in the VFD
bOutAuto	Bool	Output indicating whether the block is in auto mode
ERROR_VFD	udtError_VFD	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

6.1.3. User Defined Types

6.1.3.1. udtHMI_VFD_Control

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
rManualSpeedSP	Real	Speed set point for manual mode
rAutoSpeedSP	Real	Speed set point for automatic mode
rActualSpeed	Real	Actual motor speed

rActualCurrent	Real	Actual motor current
rActualPower	Real	Actual motor power
bPB_ResetError	Bool	Pushbutton Reset errors
bPB_Forward	Bool	Pushbutton Move forward in manual mode
bPB_Reverse	Bool	Move reverse in manual mode pushbutton
bPB_Stop	Bool	Stop in manual mode pushbutton
bPBEN_ResetError	Bool	Reset errors pushbutton enabled
bPBEN_Forward	Bool	Forward pushbutton enabled
bPBEN_Reverse	Bool	Reverse pushbutton enabled
bPBEN_Stop	Bool	Stop pushbutton enabled
bForwardOn	Bool	Run forward command is on
bReverseOn	Bool	Run reverse command is on
bSignalForward	Bool	Forward signal
bSignalReverse	Bool	Reverse signal
bError	Bool	Overall error
bInterlock	Bool	VFD Interlocked

6.1.3.2. udtError_VFD

Errors	Description
MotorProtectorTripped	Motor Protector tripped
LocalDisconnectOff	Motor Local switch OFF
ClutchTripped	Clutch error
NoContactorFeedback	No feedback from motor contactor
ContactorStillOn	Contactor still ON
NoSignalForward	No SIGNAL Forward from VFD
NoSignalReverse	No SIGNAL Reverse from VFD
MotorNotStopped	Motor doesn't stop
MaxFrequencyReached	Max frequency reached
Overcurrent	VFD overcurrent
MotorOverload	Motor overload
VFDOverload	VFD overload

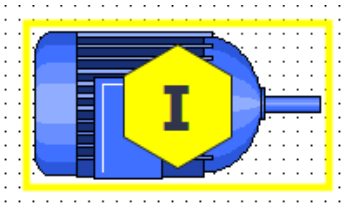
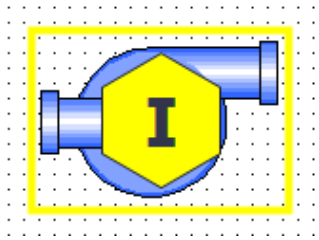
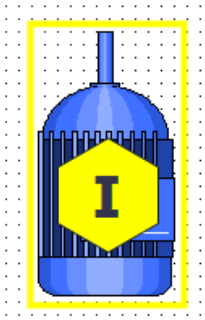
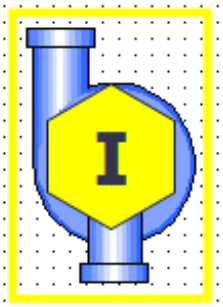
GeneralFault	VFD fault
NoCoolingFanFeedback	Cooling fan did not come on within 30 seconds

6.1.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific user application styles.

6.1.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

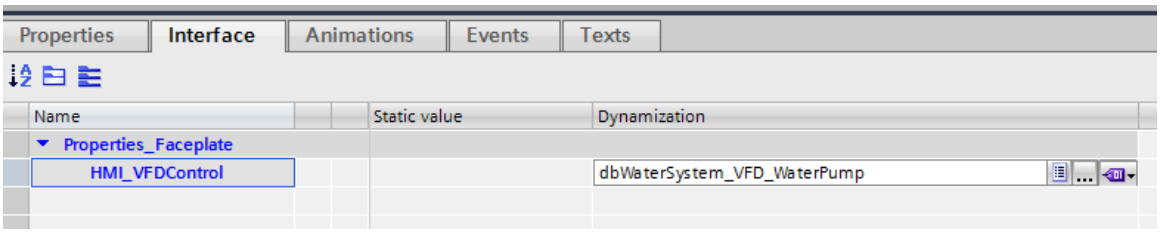
Motor Horizontal	Pump Horizontal	Motor Vertical	Pump Vertical
			

6.1.4.2. HMI Icon Appearance and Functionality

Indicator	Meaning
Device Color	Blue: Stopped Green: Running Yellow: Start Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

6.1.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_VFDControl property. The HMI_VFDControl needs to be mapped to the same 'udtHMI_VFDControl' used by the Function Block.

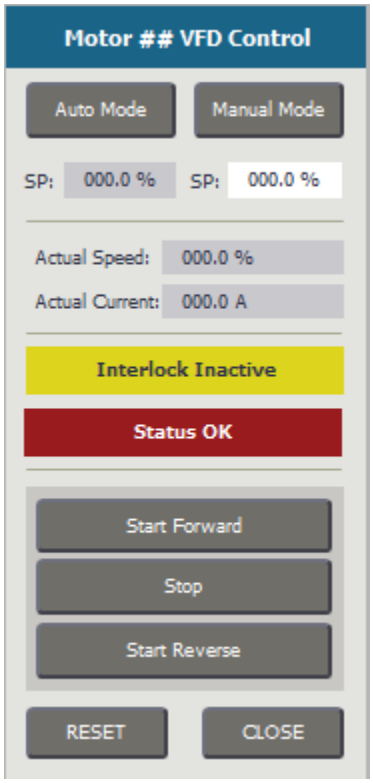


6.1.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the VFD. Additionally, it allows for control on the VFD when the System Mode is Independent or Manual.

6.1.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



6.1.5.2. HMI Pop-up Appearance and Functionality

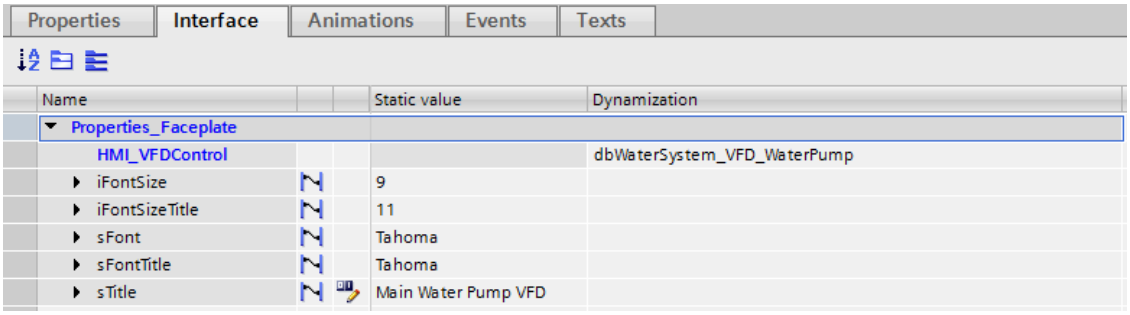
The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode. Button is green when device is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode. Button is green when device is in Manual Mode.
Auto Mode SP	Output	Displays PLC set point to be used in Auto Mode
Manual Mode SP	Input	Adjusts the speed set point when in Manual Mode
Actual Speed	Output	Displays the current speed of the motor (%)
Actual Current	Output	Displays motor current (A)
Interlock Status Field	Output	Displays current status of device interlock (yellow for interlock, grey when not)

Error Status Field	Output	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms.
Start Forward Button	Button	Commands motor forward when in Manual Mode
Stop Button	Button	Stops the device
Start Reverse Button	Button	Commands motor in reverse when in Manual Mode
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

6.1.5.3. HMI Pop-up Interface

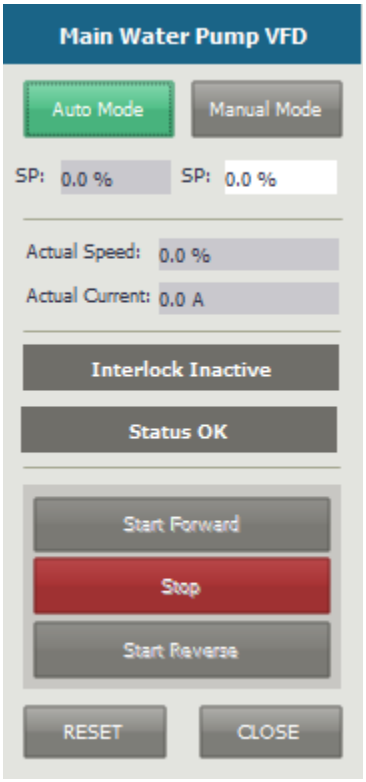
The device interface is shown below. Six properties are linked in this interface: HMI_VFDControl, iFontSize, iFontSizeTitle, sFont, sFontTitle, and sTitle. The HMI_VFDControl needs to be mapped to the same 'udtHMI_VFD_Control' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string by which to identify the device (such as a device name), and this value will display at the top of the Pop-up.



6.1.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

6.1.6.1. Screen



6.1.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_VFD_WaterPump"

2 = VFD_Popup_Pro

3 = The desired title for the pop-up. E.g. "Main Water Pump VFD"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

6.1.7. SiVarc Implementation

Five screen rules are defined for fbVFD_GSeries. One creates a pop-up that will be linked to the instance of the function block and to the created icon. The title on the pop-up is defined by NetworkTitle[0] for the network title on the instance of the function block. The other four rules create icons depending on conditional statements that reference NetworkTitle[1] and NetworkTitle[2] from the network for the instance of the function block. If either

NetworkTitle[1] or NetworkTitle[2] aren't defined, or are defined to something the screen rule doesn't recognize, no icon will be created. NetworkTitle[1] can either be "Pump" or "Motor" and NetworkTitle[2] can either "Horizontal" or "Vertical". The combination of the two determines which faceplate is used for the icon, e.g. *Title*_Pump_Horizontal will produce a horizontal pump icon, or *Title*_Motor_Vertical will produce a vertical motor icon.

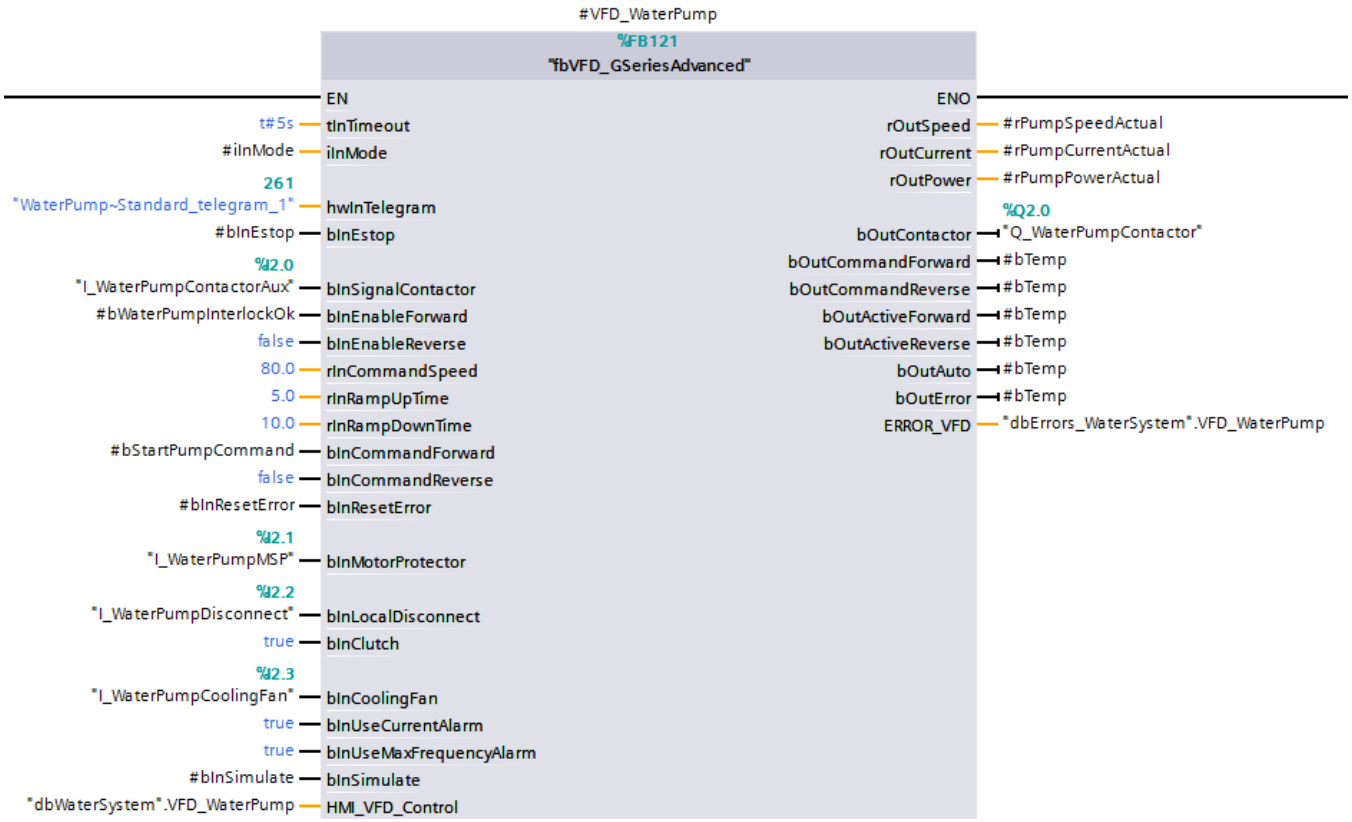
In WinCC Professional, there is no rule to create a pop-up, as there is not a separate pop-up screen for each object in the WinCC Pro implementation. Otherwise, the WinCC Professional rules function exactly as the rules for WinCC Comfort/Advanced.

6.2. G Series Advanced Motor Control – fbVFD_GSeriesAdvanced

6.2.1. Description

This block builds on the functionality of the regular G Series VFD block by adding asynchronous parameter reading and writing. Specifically this block allows the PLC to programmatically set the ramp up and down times as well as periodically read current and power values from the drive.

NOTE: This block is limited to 20 instances per processor, per Siemens' recommendation. Any additional instances may cause unpredictable behavior with asynchronous parameter reading and writing (ramp up/down, current, power).



6.2.2. Function Block Interface

6.2.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
hwInTelegram	HW_SUBMODULE	The system constant that connects to the Telegram 1 configuration of the VFD
bInSignalContactor	Bool	Contactor feedback. Map the Output bOutContactor if no Contactor exists.
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
rInCommandSpeed*	Real	Speed set point for automatic mode in percent. (0.0-100.0%)
rInRampUpTime	Real	The ramp up time for the VFD in seconds
rInRampDownTime	Real	The ramp down time for the VFD in seconds
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will run in this direction until the reverse interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be mapped directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered

bInClutch	Bool	This input should be wired directly into the motor's clutch overload input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInCoolingFan	Bool	This input is used to tell the VFD that the cooling fan is running, if available. If not available, this input should be set to true. When false for 30 seconds after VFD starts, the motor is disabled and an error is triggered
bInUseCurrentAlarm	Bool	This input enables or disables an alarm for drive overcurrent. If an overcurrent alarm should be used, then this input should be true. If the current alarm is to be disabled, then this input should be false, or left unwired
bInUseMaxFrequency	Bool	This input enables or disables an alarm for max frequency. If a frequency alarm should be used, then this input should be true. If the frequency alarm is to be disabled, then this input should be false, or left unwired
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.2.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_VFD_Control	udtHMI_VFD_Control	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also, inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

6.2.2.3. Out Parameters

Output Variables	Type	Description
bOutContactor	Bool	Contactor output. Map to Output to turn on contactor if used
bOutCommandForward	Bool	Motor commanded forward
bOutCommandReverse	Bool	Motor commanded reverse
bOutActiveForward	Bool	Motor running forward feedback from VFD
bOutActiveReverse	Bool	Motor running reverse feedback from VFD
rOutSpeed	Real	Actual speed from VFD
rOutCurrent	Real	Actual current from VFD
rOutPower	Real	Actual power from VFD
bOutError	Bool	This output indicates whether there's an error condition in the VFD
bOutAuto	Bool	Output indicating whether the block is in auto mode

ERROR_VFD	udtError_VFD	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

6.2.3. User Defined Types

See section 6.1.3.

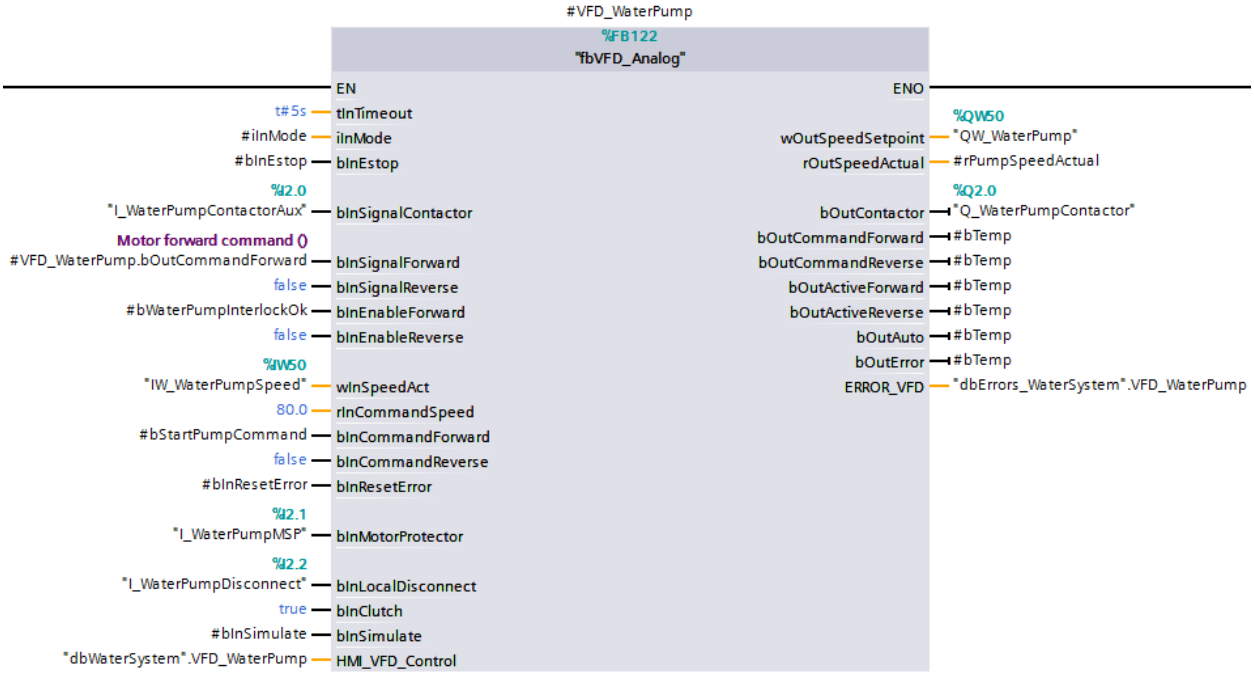
6.2.4. HMI Display

The HMI display for this block is identical those detailed for the regular G Series VFD block, detailed above starting at section 6.1.4.

6.3. Analog VFD Control – fbVFD_Analog

6.3.1. Description

The fbVFD_Analog Motor Control Function Block is utilized for use of a VFD controlled with digital and analog signals. The error UDT is kept identical to the GSeries error UDT, however, some of the errors are not utilized by the block in this revision.



6.3.2. Function Block Interface

6.3.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInSignalContactor	Bool	Contactor feedback

bInSignalForward	Bool	Forward feedback bit. This tells the motor function block whether the motor is going in the forward direction. If no feedback input is available, wire the forward output to the forward feedback input
bInSignalReverse	Bool	Reverse feedback bit. This tells the motor function block whether the motor is going in the reverse direction. If no feedback input is available, wire the forward output to the forward feedback input
wInSpeedAct	Word	Actual speed feedback from an analog speed sensor. If there is no speed feedback, then wOutSpeedSP should be mapped to this input
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
rInCommandSpeed*	Real	Speed set point for automatic mode in percent (0.0-100.0%)
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the reverse interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be wired directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInClutch	Bool	This input should be wired directly into the motor's clutch overload input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.3.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_VFD_Control	udtHMI_VFD_Control	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

6.3.2.3. Out Parameters

Output Variables	Type	Description
bOutContactor	Bool	Contactor output. Map to Output to turn on contactor if used
bOutCommandForward	Bool	Motor commanded forward
bOutCommandReverse	Bool	Motor commanded reverse
bOutActiveForward	Bool	Motor running forward feedback from VFD
bOutActiveReverse	Bool	Motor running reverse feedback from VFD
wOutSpeedSP	Word	Speed value to Map to Output to control speed. Map to analog output.
rOutSpeedActual	Real	Actual Speed based on speed feedback signal
bOutError	Bool	This output indicates whether there's an error condition in the VFD
bOutAuto	Bool	Output indicating whether the block is in auto mode
ERROR_VFD	udtError_VFD	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

6.3.3. User Defined Types

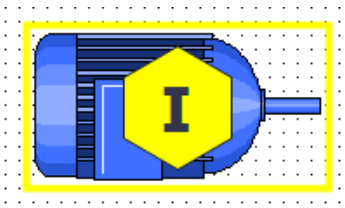
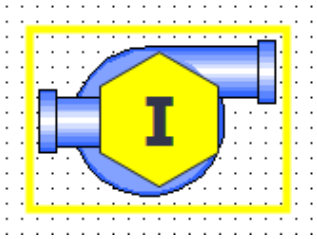
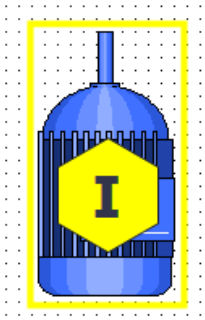
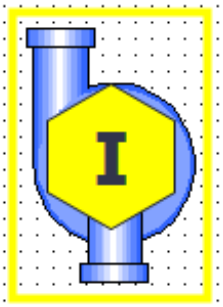
See section 6.1.3.

6.3.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific applications. Note: As the VFD_Analog faceplate uses the same UDT as VFD_GSeries they share the same faceplate.

6.3.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

Motor Horizontal	Pump Horizontal	Motor Vertical	Pump Vertical
			

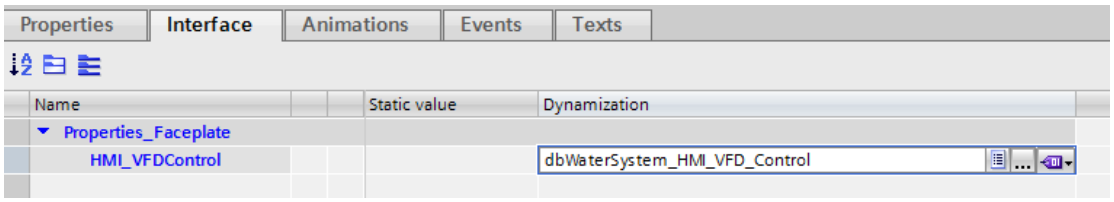
6.3.4.2. HMI Icon Appearance and Functionality

The device appearance changes depending on the state it is in, see table below for description:

Indicator	Meaning
Device Color	Blue: Stopped Green: Running Yellow: Start Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

6.3.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_VFD_Control property. The HMI_VFD_Control needs to be mapped to the same 'udtHMI_VFD_Control' used by the Function Block.



6.3.5. HMI Pop-up Faceplate

See the G-Series Motor Control HMI Pop-up Faceplate section, above. The Analog VFD type uses the same pop-up faceplate as the G-Series.

6.3.6. WinCC Professional Pop-up Screen

Please see the WinCC Pro Pop-up Screen section under G Series Motor control. The Analog VFD type uses the same pop-up scree.

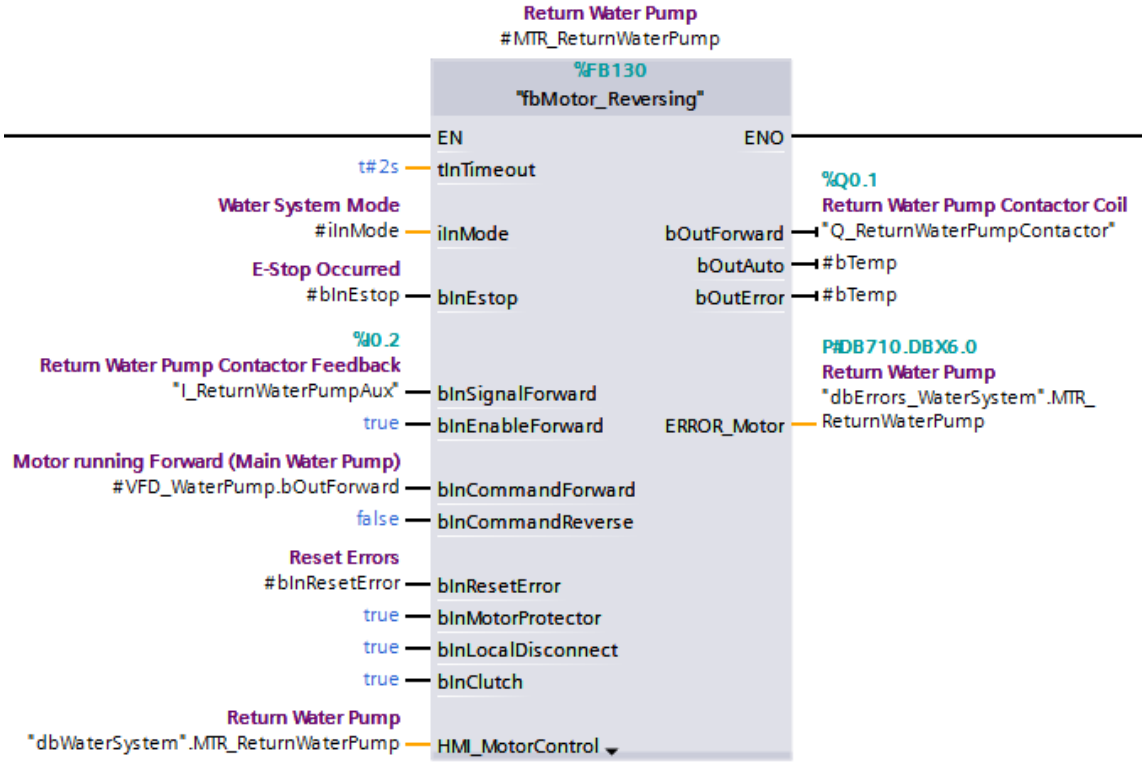
6.3.7. SiVarc Implementation

Please see SiVArc Implementation under G Series Motor control. The implementation for Analog VFD control is the same.

6.4. Digital Single Speed Motor – fbMotor_Reversing

6.4.1. Description

The Reversing Motor Control Function Block is utilized for motors started and stopped using digital outputs. If the Motor is only one direction, then map False into bInEnableReverse, and this will perform as a single direction motor.



6.4.2. Function Block Interface

6.4.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInSignalForward	Bool	Forward feedback bit. This tells the motor function block whether the motor is going in the forward direction. If no feedback input is available, wire the forward output to the forward feedback input
bInSignalReverse	Bool	Reverse feedback bit. This tells the motor function block whether the motor is going in the reverse direction. If no feedback input is available, wire the forward output to the forward feedback input
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input

		should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the reverse interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be wired directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInClutch	Bool	This input should be wired directly to the motor's clutch input. When false, the motor is disabled and an error is triggered
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.4.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_MotorControl	udtHMI_MotorControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands

6.4.2.3. Out Parameters

Output Variables	Type	Description
bOutCommandForward	Bool	Motor commanded forward
bOutCommandReverse	Bool	Motor commanded reverse
bOutActiveForward	Bool	Motor running forward feedback
bOutActiveReverse	Bool	Motor running reverse feedback
bOutError	Bool	This output indicates whether there's an error condition in the motor
ERROR_Motor	udtError_Motor	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state
bOutAuto	Bool	This output indicates whether the block is in auto mode

6.4.3. User Defined Types

6.4.3.1. udtHMI_MotorControl

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Forward	Bool	Move forward in manual mode pushbutton
bPB_Reverse	Bool	Move Reverse in manual mode pushbutton
bPB_Stop	Bool	Stop in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Errors pushbutton enabled
bPBEN_Forward	Bool	Forward pushbutton enabled
bPBEN_Reverse	Bool	Reverse pushbutton enabled
bPBEN_Stop	Bool	Stop pushbutton enabled
bForwardOn	Bool	Forward command is on
bReverseOn	Bool	Reverse command is on
bSignalForward	Bool	Forward signal
bSignalReverse	Bool	Reverse signal
bError	Bool	Overall error
bInterlock	Bool	Motor interlocked

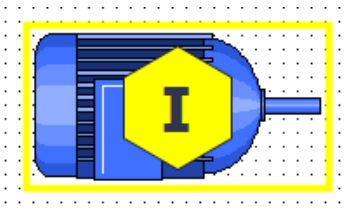
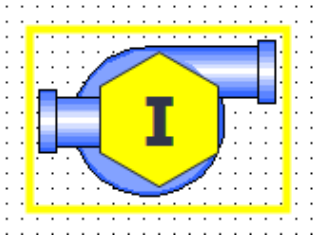
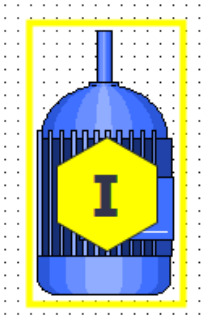
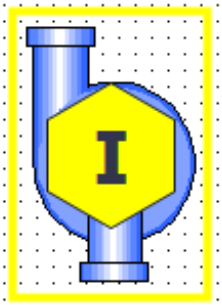
6.4.3.2. udtError_Motor

Name	Description
MotorProtectorTripped	Motor protector tripped
LocalDisconnectOff	Local disconnect off
ClutchTripped	Clutch tripped
NoSignalForward	No signal from forward contactor
NoSignalReverse	No signal from Reverse contactor
MotorNotStopped	Motor doesn't stop

6.4.4. HMI Icon Display

6.4.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

Motor Horizontal	Pump Horizontal	Motor Vertical	Pump Vertical
			

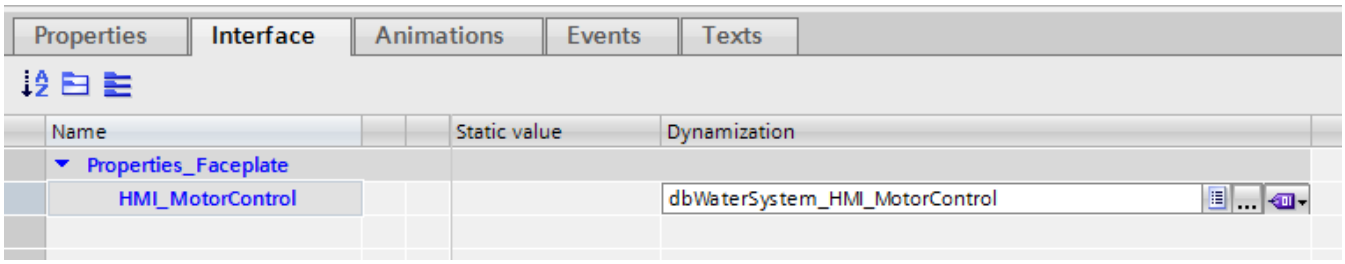
6.4.4.2. HMI Icon Appearance and Functionality

The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Device Color	Blue: Stopped Green: Running Yellow: Start Red Flashing: Error State Red: E-stop pressed
Yellow Border	Manual Mode
Yellow Hexagon with 'I' in center	Interlocked

6.4.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_MotorControl property. The HMI_MotorControl property needs to be mapped to the same 'udtHMI_MotorControl' used by the Function Block.



6.4.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the motor. Additionally, it allows for control on the motor when the System Mode is Independent or Manual modes.

6.4.5.1. HMI Pop-up display

The library contains a single pop-up faceplate shown below:

6.4.5.2. HMI Pop-up Appearance and Functionality

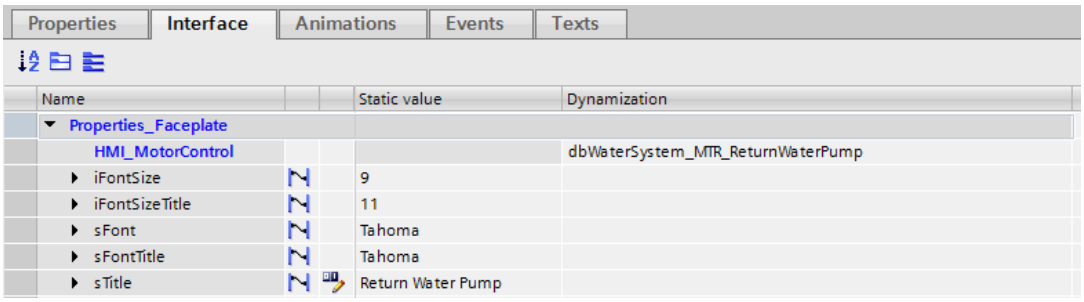
The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms.
Start Forward Button	Button	Drives motor forward when in Manual Mode
Stop Button	Button	Stops the device
Start Reverse Button	Button	Drives motor in reverse when in Manual Mode
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

6.4.5.3. HMI Pop-up Interface

The device interface is shown below. Six properties are linked in this interface: HMI_MotorControl, iFontSize, iFontSizeTitle, sFont, sFontTitle, and sTitle. The HMI_MotorControl needs to be mapped to the same 'udtHMI_Motor_Control' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sFont property controls the font name used

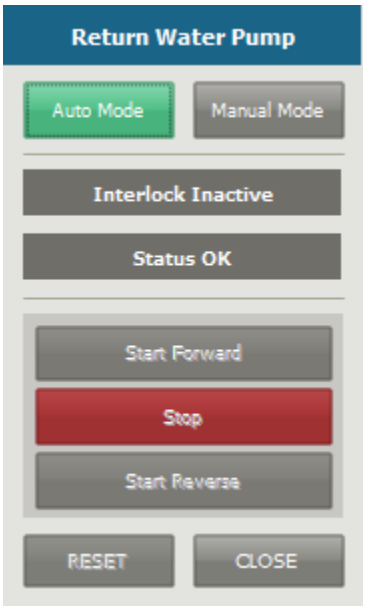
by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



6.4.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

6.4.6.1. Screen



6.4.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_MTR_ReturnWaterPump"

2 = Motor_Reversing_Popup_Pro

3 = The desired title for the pop-up. E.g. "Return Water Pump"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

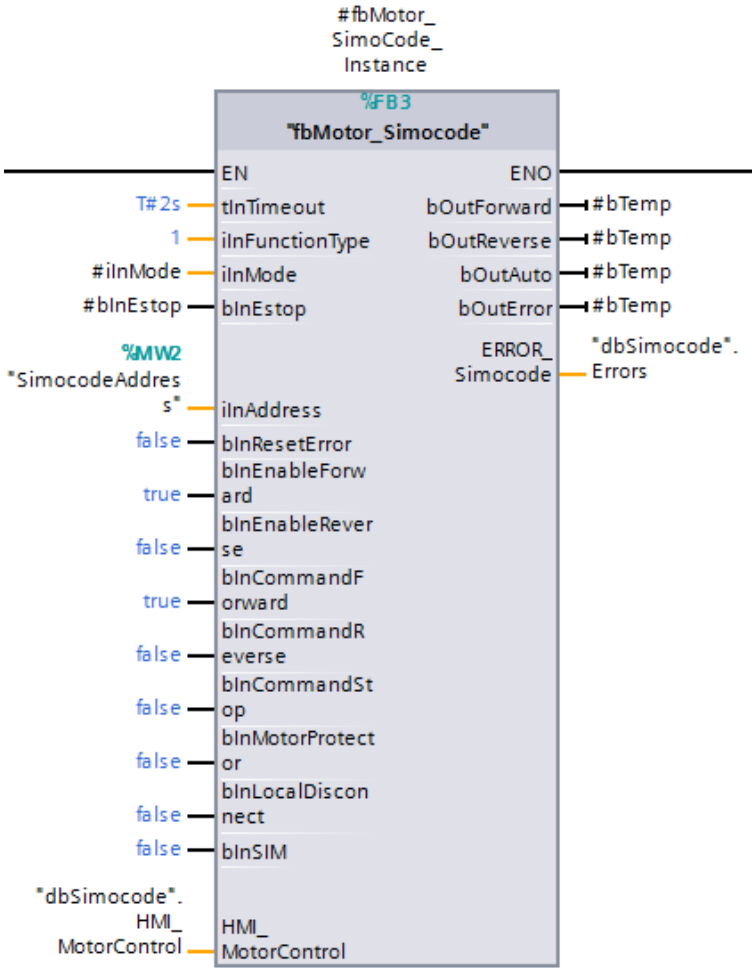
6.4.7. SiVarc Implementation

Please see SiVArc Implementation under G Series Motor control. The implementation for Digital Motor control is the same.

6.5. Simocode Single Speed Motor– fbMotor_Simocode

6.5.1. Description

The Simocode Motor Control Function Block is utilized for controlling motors via a Type 2 Simocode Pro V Module. This block can be configured for an Overload Relay, Direct Starter or Reversing Starter control functions.



6.5.2. Function Block Interface

6.5.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInFunctionType	Int	Declares control function type. Map 0 for overload relay, 1 for direct starter or 2 for reversing starter
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
iInAddress	Int	Address of cyclic receive and send data for control functions on Simocode module. This contains all of the control and feedback data
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false (Direct starter and reversing starter only, wire false for overload relay)
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false (Reversing starter only, wire false for overload relay and direct starter)
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the forward interlock becomes false, or this input is set low (Direct starter and reversing starter only, wire false for overload relay)
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the reverse interlock becomes false, or this input is set low (Reversing starter only, wire false for overload relay and direct starter)
bInCommandStop	Bool	This is the stop command for auto mode. If the input is set high, it will stop the motor.
bInMotorProtector	Bool	When this input is set low, it triggers the Motor Protector Tripped error.
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.5.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_MotorControl	udtHMI_MotorControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands

6.5.2.3. Out Parameters

Output Variables	Type	Description
bOutCommandForward	Bool	Motor commanded forward
bOutCommandReverse	Bool	Motor commanded reverse
bOutActiveForward	Bool	Motor running forward feedback from VFD
bOutActiveReverse	Bool	Motor running reverse feedback from VFD
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the motor
ERROR_Simocode	udtError_Simocode	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

6.5.3. User Defined Types

6.5.3.1. udtHMI_MotorControl

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Forward	Bool	Move forward in manual mode pushbutton
bPB_Reverse	Bool	Move Reverse in manual mode pushbutton
bPB_Stop	Bool	Stop in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Errors pushbutton enabled
bPBEN_Forward	Bool	Forward pushbutton enabled
bPBEN_Reverse	Bool	Reverse pushbutton enabled
bPBEN_Stop	Bool	Stop pushbutton enabled
bForwardOn	Bool	Forward command is on
bReverseOn	Bool	Reverse command is on
bSignalForward	Bool	Forward signal
bSignalReverse	Bool	Reverse signal
bError	Bool	Overall error
bInterlock	Bool	Motor interlocked

6.5.3.2. udtError_Simocode

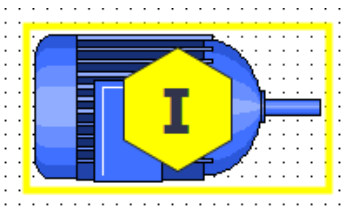
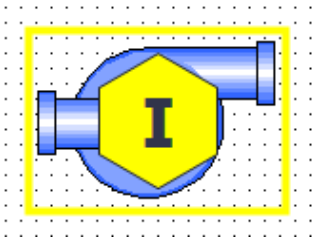
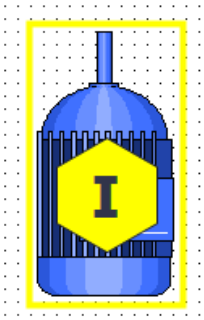
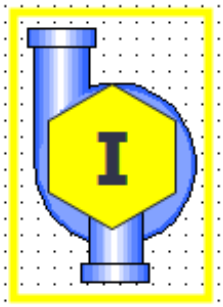
Name	Description
MotorProtectorTripped	Motor protector tripped
NoSignalForward	No signal from forward contactor
NoSignalReverse	No signal from Reverse contactor
MotorOverload	Motor overload
GeneralFault	Simocode fault
NotInRemote	Module not in remote mode
MotorNotStopped	Motor doesn't stop
LocalDisconnectOff	Local disconnect off

6.5.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific user application styles.

6.5.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

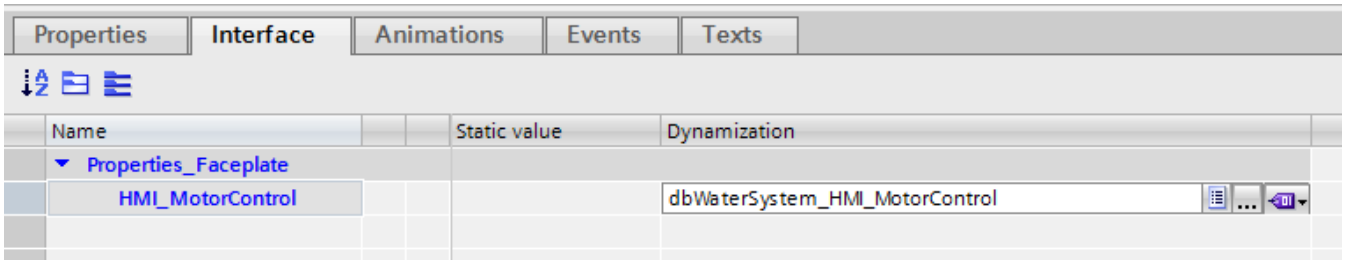
Motor Horizontal	Pump Horizontal	Motor Vertical	Pump Vertical
			

6.5.4.2. HMI Icon Appearance and Functionality

Indicator	Meaning
Device Color	Blue: Stopped Green: Running Yellow: Start Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

6.5.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_MotorControl property. The HMI_MotorControl property needs to be mapped to the same 'udtHMI_MotorControl' used by the Function Block.

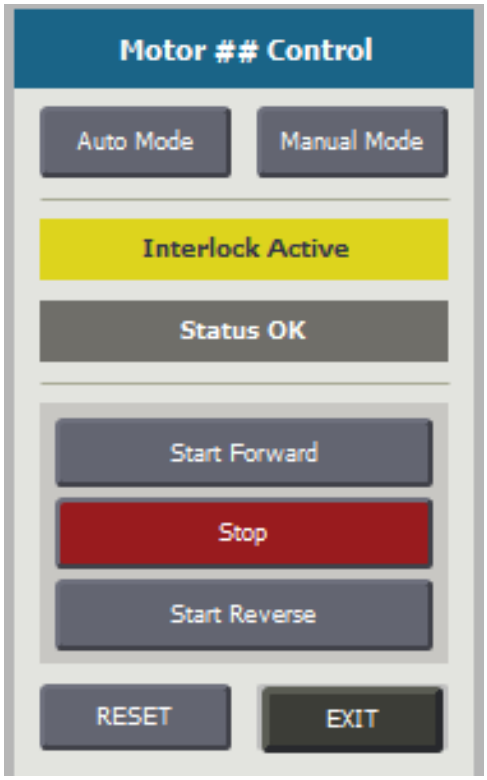


6.5.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the Simocode motor. Additionally, it allows for control on the Simocode motor when the System Mode is Independent or Manual modes.

6.5.5.1. HMI Pop-up display

The library contains a single pop-up faceplate shown below:



6.5.5.2. HMI Pop-up Appearance and Functionality

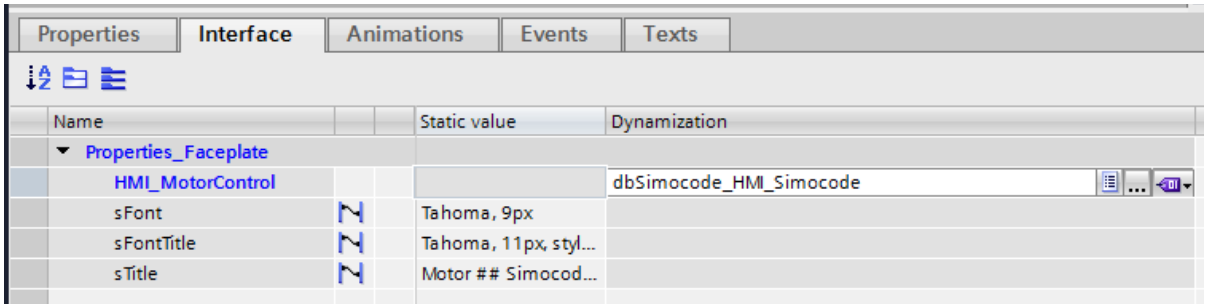
The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.

Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms.
Start Forward Button	Button	Drives motor forward when in Manual Mode (Enabled for Direct Starter and Reversing Starter Only)
Stop Button	Button	Stops the device (Enabled for Direct Starter and Reversing Starter Only)
Start Reverse Button	Button	Drives motor in reverse when in Manual Mode (Enabled for Reversing Starter only)
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

6.5.5.3. HMI Pop-up Interface

The device interface is shown below. Two properties are linked in this interface: HMI_MotorControl and sTitle. The HMI_MotorControl needs to be mapped to the same 'udtHMI_MotorControl' used by the Function Block. The sFont and sFontTitle are strings that define the appearance of the button and title fonts, respectively, and have the format "*desired font*, *desired size*, style=*desired style*", where the style is optional. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



6.5.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

6.5.6.1. Screen

6.5.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_MTR_ReturnWaterPump"

2 = Motor_Simocode_Popup_Pro

3 = The desired title for the pop-up. E.g. "Return Water Pump"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

6.5.7. SiVarc Implementation

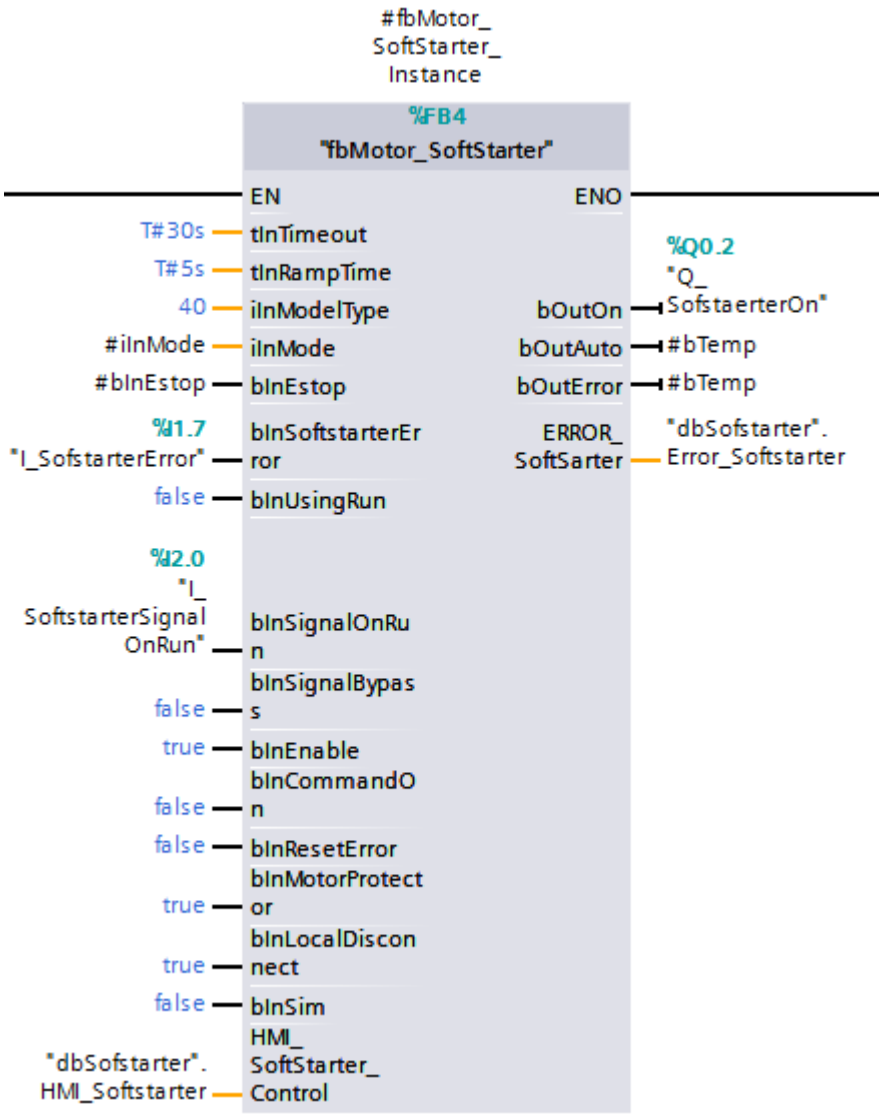
Please see SiVArc Implementation under G Series Motor control. The implementation for Simocode Motor control is the same for WinCC Comfort/Advanced.

In WinCC Professional, there is no rule to create a pop-up, as there is not a separate pop-up screen for each object in the WinCC Pro implementation. Additionally, in order for the correct pop-up to be loaded, NetworkTitle[3] needs to be defined to be "Simocode". This is because the icons for fbMotor_Reversing and fbMotor_Simocode are shared, so to differentiate between the pop-ups, another condition is needed.Otherwise, the WinCC Professional rules function exactly as the rules for WinCC Comfort/Advanced.

6.6. Soft Starter Motor– fbMotor_Softstarter

6.6.1. Description

The Soft Starter Control Function Block is utilized for controlling motors via a SIRIUS 3RW Soft Starter. This block can be configured for the 3RW30 and 3RW40 models.



6.6.2. Function Block Interface

6.6.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error. This time must be greater than the ramp-up and ramp-down times. Default value is 20 seconds
tInRampTime	Time	The amount of time the device will take to ramp up the output voltage to its full value on startup, or ramp down on shutdown. Only applicable for the 3RW40, will be ignored for the 3RW30.
iInModelType	Int	Declares SIRIUS Soft Start model type. Map 30 for 3RW30 models, 40 for 3RW40 models. If no model, or an incorrect model, is declared 3RW30 is set as default

iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInSoftstarterError	Bool	This input from the soft starter module declares an overload error or general failure within the module
bInUsingRun	Bool	Declares the reparameterization from on to run. Wire true if using the run function, wire false otherwise (3RW40 model only, wire false for 3RW30 model)
bInSignalOnRun	Bool	On/run feedback bit. This tells the soft starter function block whether the motor is on/running
bInSignalBypass	Bool	Bypass feedback bit. This tells the soft starter function block whether the motor has started up successfully (3RW40 model only, wire false for 3RW30 model)
bInEnable	Bool	On interlock. The soft starter is allowed to run if this condition has been met. If the soft starter should always be able to run, this input should be set to true. To disable the soft starter from ever running, this input should be set to false
bInCommandOn*	Bool	This is the forward on for auto mode. If this input is set high in auto mode the soft starter will continue to run until the enable interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInMotorProtector	Bool	If this bit goes false, the Motor Protector Tripped error is set.
bInLocalDisconnect	Bool	If this bit goes false, the Local Disconnect Off error is set. If it goes true again, the error is reset.
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.6.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_SoftstarterControl	udtHMI_Softstarter	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including on and off commands

6.6.2.3. Out Parameters

Output Variables	Type	Description
bOutCommandOn	Bool	This output should be wired to the soft starter's on/run contact
bOutActiveOn	Bool	Soft starter has on feedback
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the motor
ERROR_Softstarter	udtError_Softstarter	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

6.6.3. User Defined Types

6.6.3.1. udtHMI_SoftStarter

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_On	Bool	Turn on in manual mode pushbutton
bPB_Off	Bool	Turn off in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Errors pushbutton enabled
bPBEN_On	Bool	On pushbutton enabled
bPBEN_Off	Bool	Off pushbutton enabled
bOn	Bool	On command is on
bSignalOnRun	Bool	On/run signal
bSignalBypass	Bool	Bypass signal
bError	Bool	Overall error
bInterlock	Bool	Soft starter interlocked

6.6.3.2. udtError_Softstarter

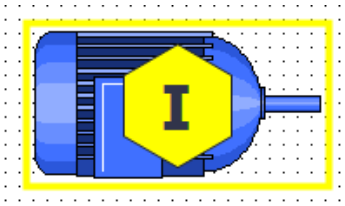
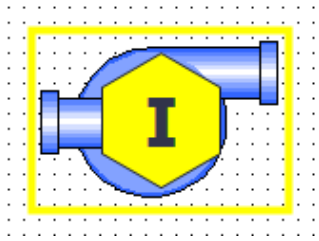
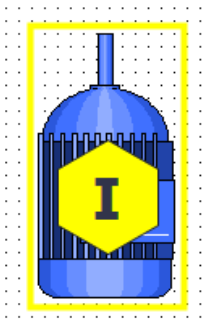
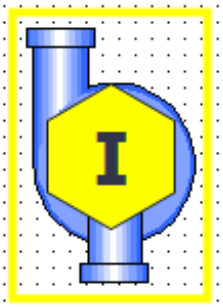
Name	Description
NoSignalOnRun	No signal from on/run contactor
NoSignalBypass	No signal from bypass contactor
MotorNotStopped	Soft starter doesn't stop
SoftstarterFault	Error of fault from soft starter module
SoftStarterWarning	General warning from soft starter module
MotorProtectorTripped	Motor Protector Tripped
LocalDisconnectOff	Local Disconnect Off
DeviceNotReady	No ready signal from soft starter

6.6.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific user application styles.

6.6.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

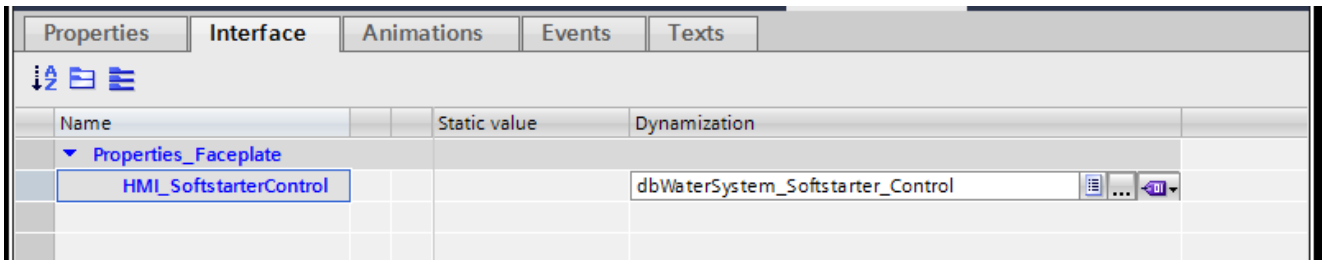
Motor Horizontal	Pump Horizontal	Motor Vertical	Pump Vertical
			

6.6.4.2. HMI Icon Appearance and Functionality

Indicator	Meaning
Device Color	Blue: Stopped Green: Running Yellow: Start Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

6.6.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_SoftstarterControl property. The HMI_ SoftstarterControl property needs to be mapped to the same 'udtHMI_Softstarter' used by the Function Block.

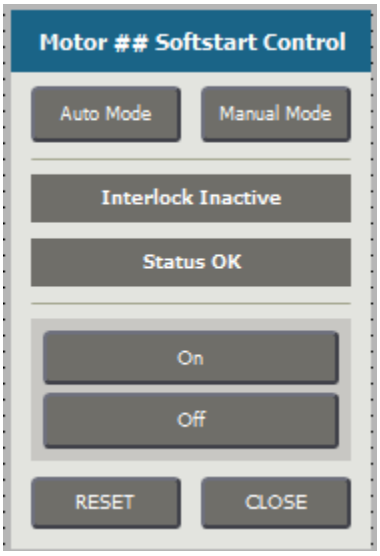


6.6.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the Soft Starter. Additionally, it allows for control on the Soft Starter when the System Mode is Independent or Manual modes.

6.6.5.1. HMI Pop-up display

The library contains a single pop-up faceplate shown below:



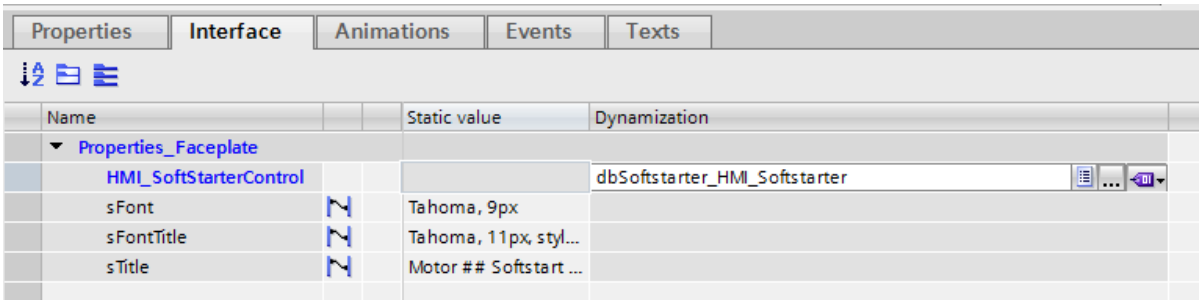
6.6.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms.
Start Forward Button	Button	Drives motor forward when in Manual Mode (Enabled for Direct Starter and Reversing Starter Only)
Stop Button	Button	Stops the device (Enabled for Direct Starter and Reversing Starter Only)
Start Reverse Button	Button	Drives motor in reverse when in Manual Mode (Enabled for Reversing Starter only)
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

6.6.5.3. HMI Pop-up Interface

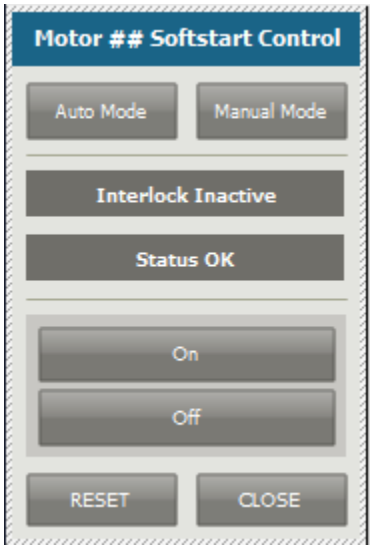
The device interface is shown below. Four properties are linked in this interface: HMI_SoftstarterControl sFont, sFontTitle, and sTitle. The HMI_SoftstarterControl needs to be mapped to the same 'udtHMI_Softstarter' used by the Function Block. The sFont and sFontTitle are strings that define the appearance of the button and title fonts, respectively, and have the format "*desired font*, *desired size*, style=*desired style*", where the style is optional. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



6.6.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

6.6.6.1. Screen



6.6.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_MTR_ReturnWaterPump"

2 = Motor_Softstarter_Popup_Pro

3 = The desired title for the pop-up. E.g. "Return Water Pump"

```
Sub OnClick(ByVal item)
SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"
SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"
ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"
HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"
End Sub
```

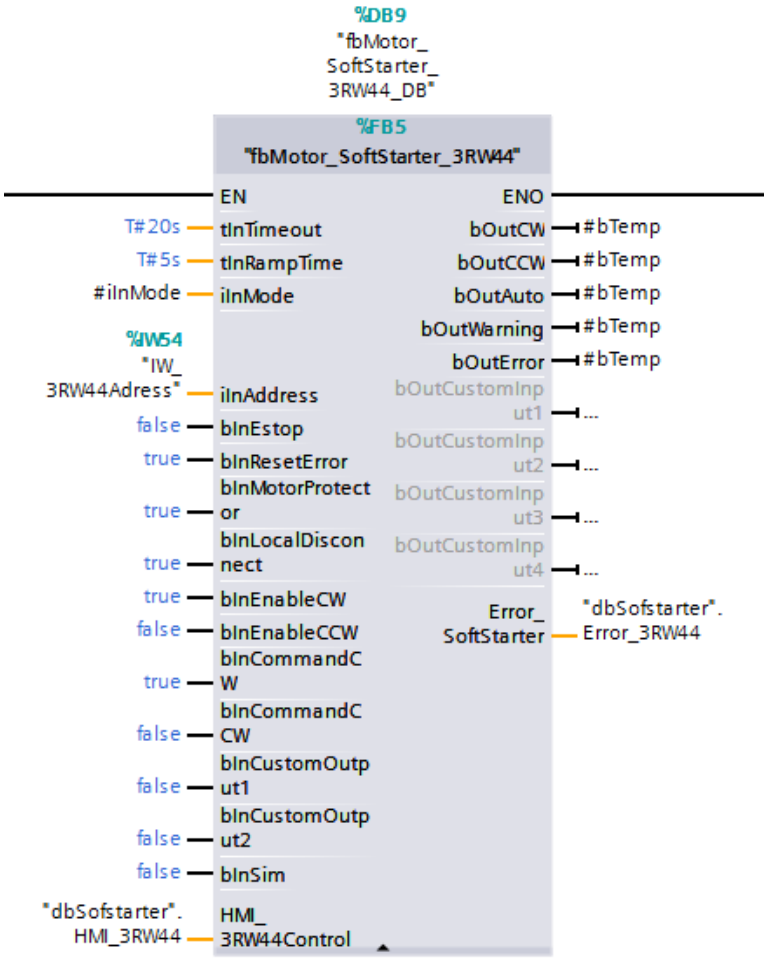
6.6.7. SiVarc Implementation

Please see SiVArc Implementation under G Series Motor control. The implementation for Soft Starter control is the same.

6.7. 3RW44 Soft Starter Motor – fbMotor_Softstarter_3RW44

6.7.1. Description

The Soft Starter Control Function Block is utilized for controlling motors via a SIRIUS 3RW44 Soft Starter.



6.7.2. Function Block Interface

6.7.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error. This time must be greater than the ramp-up and ramp-down times. Default value is 20 seconds
tInRampTime	Time	The amount of time the device will take to ramp up the output voltage to its full value on startup, or ramp down on shutdown. Only applicable for the 3RW40, will be ignored for the 3RW30.
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
iInAddress	Int	Address of cyclic receive and send data for control functions on 3RW44 module. This contains all of the control and feedback data
bInEstop	Bool	Emergency stop input
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInMotorProtector	Bool	If this bit goes false, the Motor Protector Tripped error is set.
bInLocalDisconnect	Bool	If this bit goes false, the Local Disconnect Off error is set. If it goes true again, the error is reset.
bInEnableCW	Bool	On interlock. The soft starter is allowed to run clockwise if this condition has been met. If the soft starter should always be able to run, this input should be set to true. To disable the soft starter from ever running, this input should be set to false
bInEnableCCW	Bool	On interlock. The soft starter is allowed to run counter- clockwise if this condition has been met. If the soft starter should always be able to run, this input should be set to true. To disable the soft starter from ever running, this input should be set to false
bInCommandCW*	Bool	This is the forward on for auto mode. If this input is set high in auto mode the soft starter will continue to run in the clockwise direction until the enable interlock becomes false, or this input is set low
bInCommandCCW*	Bool	This is the forward on for auto mode. If this input is set high in auto mode the soft starter will continue to run in the counter-clockwise direction until the enable interlock becomes false, or this input is set low
bInCustomOutput1	Bool	Bit of the status word sent to the soft starter from the PLC. Controls output 1 on the 3RW44 soft starter.
bInCustomOutput2	Bool	Bit of the status word sent to the soft starter from the PLC. Controls output 2 on the 3RW44 soft starter.
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

6.7.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_3RW44Control	udtHMI_Softstarter_3RW44	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including on and off commands

6.7.2.3. Out Parameters

Output Variables	Type	Description

bOutCommandCW	Bool	This output should be wired to the soft starter's on/run contact for the clockwise direction.
bOutCommandCCW	Bool	This output should be wired to the soft starter's on/run contact for the counter-clockwise direction.
bOutActiveCW	Bool	Soft starter has on feedback in the clockwise direction.
bOutActiveCCW	Bool	Soft starter has on feedback in the counter-clockwise direction.
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutWarning	Bool	This output indicates whether there is a warning condition in the motor.
bOutError	Bool	This output indicates whether there's an error condition in the motor
bOutCustomInput1	Bool	This bit is part of the status word from the soft starter module to the PLC.
bOutCustomInput2	Bool	This bit is part of the status word from the soft starter module to the PLC.
bOutCustomInput3	Bool	This bit is part of the status word from the soft starter module to the PLC.
bOutCustomInput4	Bool	This bit is part of the status word from the soft starter module to the PLC.
ERROR_Softstarter	udtError_Softstarter	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

6.7.3. User Defined Types

6.7.3.1. udtHMI_3RW44Control

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
iParameterSet	Int	Configured on 3RW44: 1,2 or 3, default 1.
bPB_ResetError	Bool	Reset errors pushbutton
bPB_CW	Bool	Turn on moving clockwise in manual mode pushbutton
bPB_CCW	Bool	Turn on moving counter-clockwise in manual mode pushbutton
bPB_Stop	Bool	Turn off in manual mode pushbutton
bPB_EmergencyStart	Bool	
bPBEN_ResetError	Bool	Reset Errors pushbutton enabled
bPBEN_CW	Bool	Clockwise on pushbutton enabled
bPBEN_CCW	Bool	Counter-clockwise on pushbutton enabled

bPBEN_EmergencyStart	Bool	Emergency start pushbutton is enabled.
bSlowSpeedOn	Bool	Slow speed functionality is enabled.
bQuickStopDisabled	Bool	Quick stop functionality is disabled.
bEmergencyStartActive	Bool	Emergency start is currently active
bCWOn	Bool	Currently moving clockwise
bCCWOn	Bool	Currently moving counter-clockwise
bRampOperation	Bool	Currently ramping up or down. This is the inverse of signal bypass on the other softstarter function block.
bSignalRun	Bool	On/run signal
bError	Bool	Overall error
bInterlock	Bool	Soft starter interlocked
rActualCurrent	Real	Current output from 3RW44
bShowOptions	Bool	Used with WinCC Pro Popup to show/hide options.

6.7.3.2. udtError_Softstarter

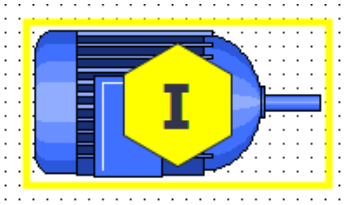
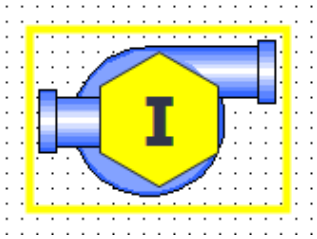
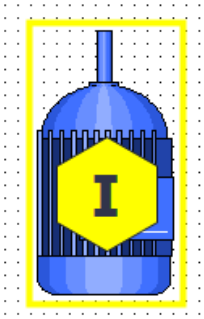
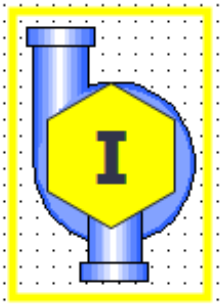
Name	Description
NoSignalOnRun	No signal from on/run contactor
NoSignalBypass	No signal from bypass contactor
MotorNotStopped	Soft starter doesn't stop
SoftstarterFault	Error of fault from soft starter module
SoftStarterWarning	General warning from soft starter module
MotorProtectorTripped	Motor Protector Tripped
LocalDisconnectOff	Local Disconnect Off
DeviceNotReady	No ready signal from soft starter

6.7.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific user application styles.

6.7.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

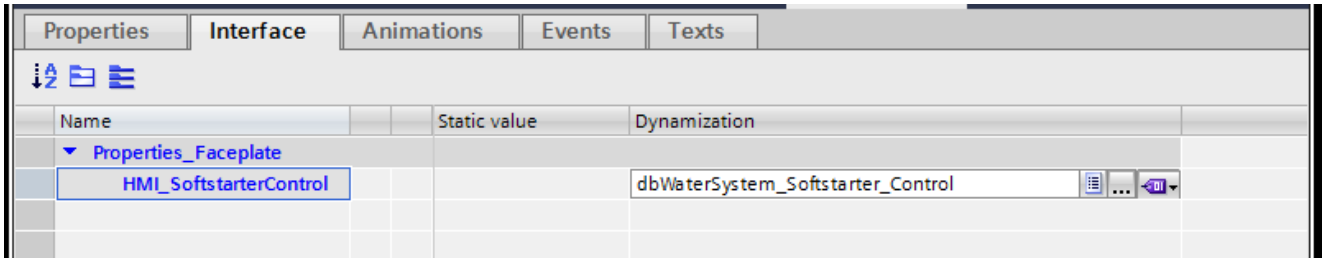
Motor Horizontal	Pump Horizontal	Motor Vertical	Pump Vertical
			

6.7.4.2. HMI Icon Appearance and Functionality

Indicator	Meaning
Device Color	Blue: Stopped Green: Running Yellow: Start Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

6.7.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_SoftstarterControl property. The HMI_ SoftstarterControl property needs to be mapped to the same 'udtHMI_Softstarter' used by the Function Block.

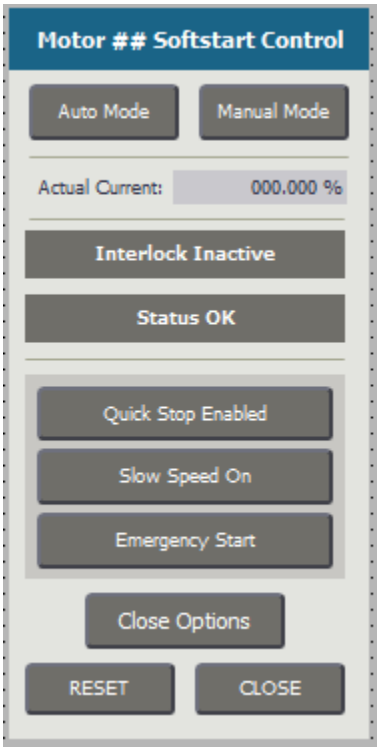


6.7.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the Soft Starter. Additionally, it allows for control on the Soft Starter when the System Mode is Independent or Manual modes.

6.7.5.1. HMI Pop-up display

The library contains a single pop-up faceplate shown below:



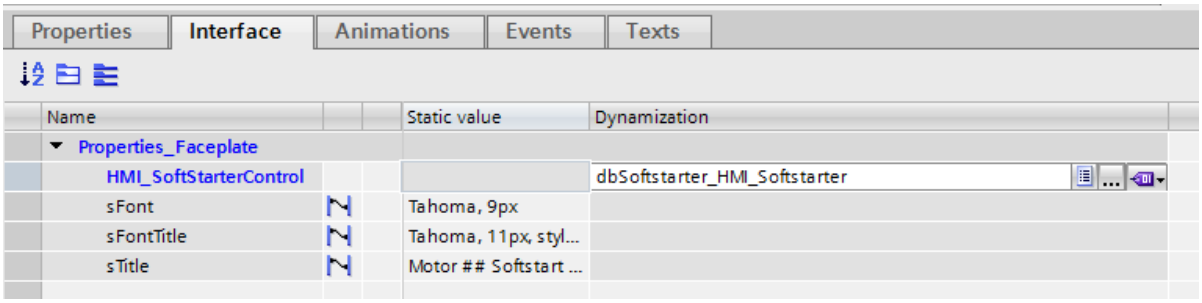
6.7.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms.
Quick Stop Enabled/Disabled	Button	Toggles the quick stop disabled bit. Text reflects current status.
Slow Speed On/Off	Button	Toggles the slow speed on bit. Text reflects current status.
Emergency Start	Button	
Close Options	Button	Toggles the bShowOptions bit.
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

6.7.5.3. HMI Pop-up Interface

The device interface is shown below. Four properties are linked in this interface: HMI_SoftstarterControl sFont, sFontTitle, and sTitle. The HMI_SoftstarterControl needs to be mapped to the same 'udtHMI_Softstarter' used by the Function Block. The sFont and sFontTitle are strings that define the appearance of the button and title fonts, respectively, and have the format "*desired font*, *desired size*, style=*desired style*", where the style is optional. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



6.7.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

6.7.6.1. Screen

6.7.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_MTR_ReturnWaterPump"

2 = Motor_Starter_3RW44_Popup_Pro

3 = The desired title for the pop-up. E.g. "Return Water Pump"

```
Sub OnClick(ByVal item)
SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"
SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"
ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"
HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"
End Sub
```

6.7.7. SiVarc Implementation

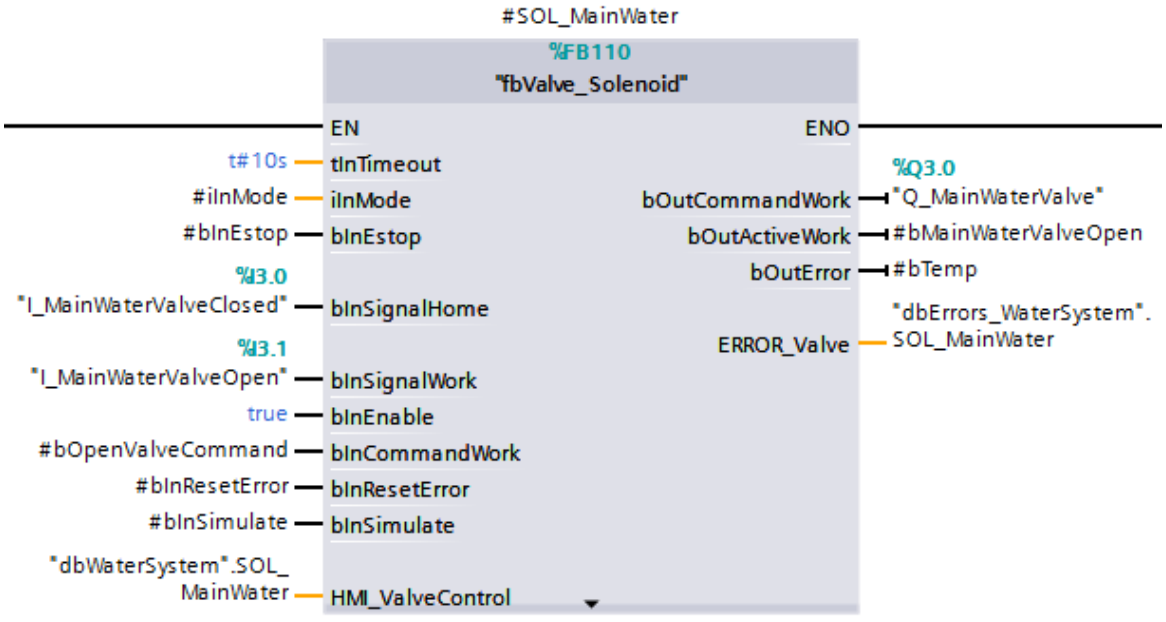
Please see SiVArc Implementation under G Series Motor control. The implementation for Soft Starter 3RW44 control is the same.

7. Valve Control

7.1. Two State Solenoid Valve – fbValve_Solenoid

7.1.1. Description

This library item is to be utilized with two state solenoid valve applications. It can be used for spring close, spring open, or double acting solenoids.



7.1.2. Function Block Interface

7.1.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInSignalHome	Bool	Home position feedback
bInSignalWork	Bool	Work position feedback
bInEnable	Bool	Valve interlock. The valve is allowed to operate if this condition has been met
bInCommandWork*	Bool	Move to work position in automatic mode
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

7.1.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_ValveControl	udtHMI_ValveControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the cylinder in manual mode

7.1.2.3. Out Parameters

Output Variables	Type	Description
bOutCommandHome	Bool	Output for home position
bOutCommandWork	Bool	Output for work position
bOutActiveHome	Bool	Valve is actively in home position
bOutActiveWork	Bool	Valve is actively in work position
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the cylinder
ERROR_Valve	udtError_Valve	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

7.1.3. User Defined Types

7.1.3.1. udtError_Valve

Errors	Description
NoHomeFeedback	Home position feedback not active
NoWorkFeedback	Work position feedback not active
HomeFeedbackStillActive	Home position feedback still active when commanded to Work Position
WorkFeedbackStillActive	Work position feedback still active when commanded to Home Position

7.1.3.2. udtHMI_ValveControl

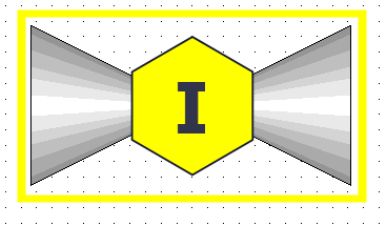
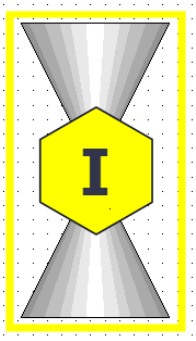
Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Home	Bool	Move to home in manual mode pushbutton
bPB_Work	Bool	Move to work in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Error pushbutton enabled
bPBEN_Home	Bool	Home pushbutton enabled
bPBEN_Work	Bool	Work pushbutton enabled
bHomeOn	Bool	Home command is on
bWorkOn	Bool	Work command is on
bSignalHome	Bool	Home feedback
bSignalWork	Bool	Work feedback
bError	Bool	Error status
bInterlock	Bool	Valve interlocked

7.1.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific applications.

7.1.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

Valve Horizontal	Valve Vertical
	

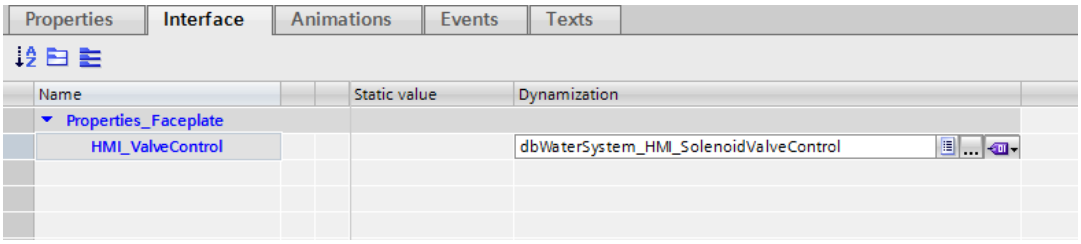
7.1.4.2. HMI Icon Appearance and Functionality

The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Device Color	Green: Open Yellow: Valve Opening/Closing Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

7.1.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_ValveControl property. The HMI_ValveControl property requires a datablock of the 'udtHMI_ValveControl' type to be linked: in the example below it is dbWaterSystem_HMI_SolenoidValveControl.

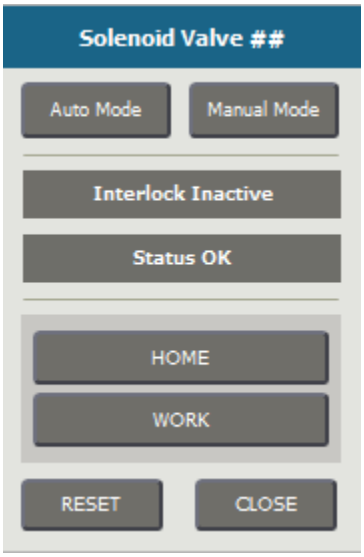


7.1.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the valve. Additionally, it allows for control on the valve when the System Mode is Independent or Manual.

7.1.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



7.1.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when device is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms
HOME Button	Button	Sets valve to HOME position
WORK Button	Button	Sets valve to WORK position
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

7.1.5.3. HMI Pop-up Interface

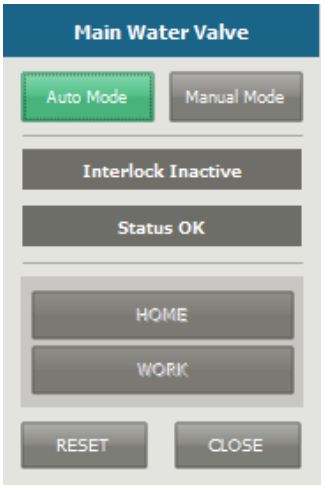
The device interface is shown below. Eight properties are linked in this interface: HMI_ValveControl, iFontSize, iFontSizeTitle, sButtonButtonText, sFont, sFontTitle, sTitle, and sTopButtonText. The HMI_ValveControl needs to be mapped to the same 'udtHMI_Valve_Control' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sBottomButtonText controls the text displayed on the Work button. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sTopButtonText controls the text displayed on the Home button.

Name	Static value	Dynamization
Properties_Faceplate		
HMI_ValveControl		dbWaterSystem_SOL_MainWater
iFontSize	9	
iFontSizeTitle	11	
sBottomButtonText	WORK	
sFont	Tahoma	
sFontTitle	Tahoma	
sTitle	Main Water Valve	
sTopButtonText	HOME	

7.1.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

7.1.6.1. Screen



7.1.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_SOL_MainWaterValve"

2 = Valve_Solenoid_Popup_Pro

3 = The desired title for the pop-up. E.g. "Main Water Valve"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

7.1.7. SiVArc Implementation

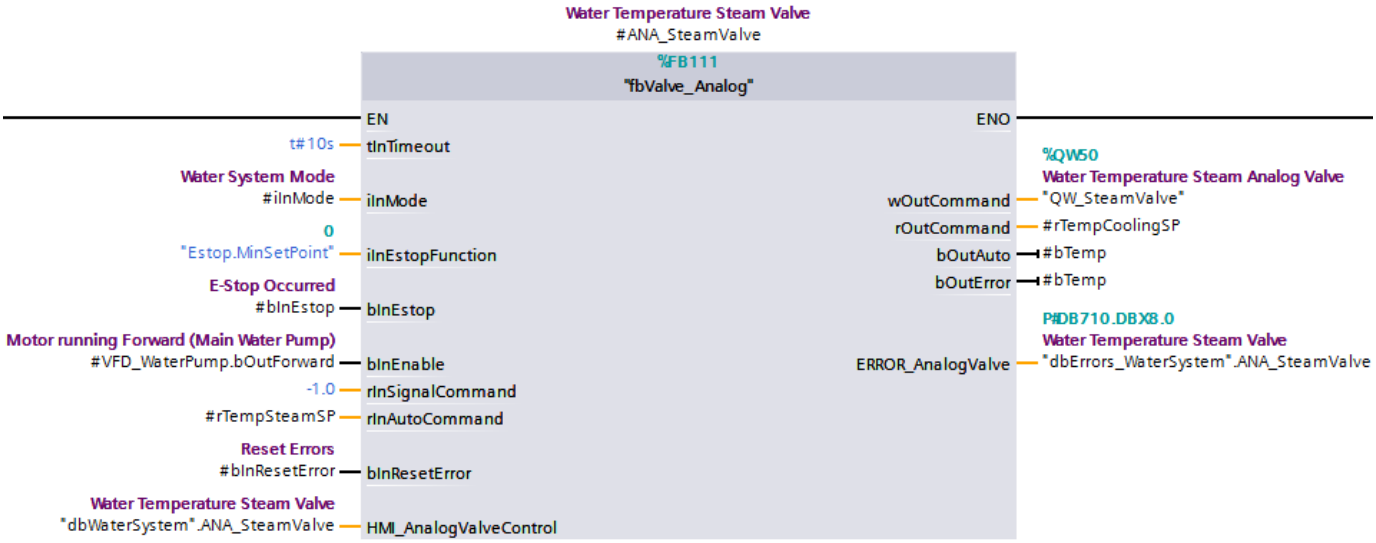
Three screen rules are defined for fbValve_Analog. One creates a pop-up that will be linked to the instance of the function block and to the created icon. The title on the pop-up is defined by NetworkTitle[0] for the network title on the instance of the function block. The other two rules create icons depending on conditional statements that reference NetworkTitle[1] from the network for the instance of the function block. If NetworkTitle[1] isn't defined, or is defined to something the screen rule doesn't recognize, both horizontal and vertical valve icons will be generated. NetworkTitle[1] can either be "Horizontal" or "Vertical", and will produce either a horizontal or vertical valve icon, respectively.

In WinCC Professional, there is no rule to create a pop-up, as there is not a separate pop-up screen for each object in the WinCC Pro implementation. Otherwise, the WinCC Professional rules function exactly as the rules for WinCC Comfort/Advanced.

7.2. Analog Valve – fbValve_Analog

7.2.1. Description

This library item is to be utilized with a valve controlled using an analog output. This block will work with any valve that is controlled through an analog output from the PLC.



7.2.2. Function Block Interface

7.2.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
iInEstopFunction	Int	Enumerated value of E-Stop response constant. 'Estop.MinSetPoint' sets rOutCommand to the minimum set point (0%), 'Estop.MaxSetPoint' sets to the maximum set point (100%), and 'Estop.HMISetPoint' sets to an HMI-specified value that can be in between the min and max.
bInEstop	Bool	Emergency stop input
bInEnable	Bool	Valve interlock. The valve is allowed to operate if this condition has been met
rInSignalCommand	Real	Actual % open from valve (-1 to disable)
rInAutoCommand	Real	Auto mode set point command
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInSimulate	Bool	Enables software simulation of device.

7.2.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_AnalogValveControl	udtHMI_AnalogValveControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the cylinder in manual mode and provide status for the HMI

7.2.2.3. Out Parameters

Output Variables	Type	Description
wOutCommand	Word	Value to be mapped to output used to control cylinder
rOutCommand	Real	Output value from 0-100% used for other PLC logic
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the cylinder
ERROR_AnaValve	udtError_AnaValve	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

7.2.3. User Defined Types

7.2.3.1. udtHMI_AnalogValveControl

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Color status for HMI. See Section 12.3.1 of the Library Overview and Architecture document for further details
rManualSP	Real	Set point for manual mode
rAutoSP	Real	Set point for automatic mode
rEstopSP	Real	Set point for an e-stop condition, if selected
rActual	Real	Actual value
bPB_ResetError	Bool	Reset errors pushbutton
bPBEN_ResetError	Bool	Reset errors pushbutton enable
bError	Bool	Overall error
bInterlock	Bool	Valve Interlocked
iEstopFunction	Int	Enumerated value of E-Stop response constant.

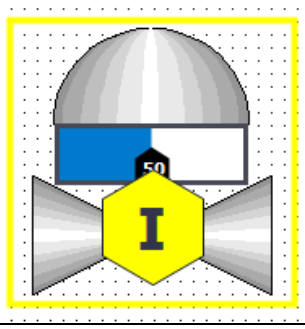
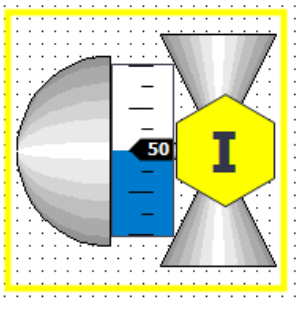
7.2.3.2. udtError_AnalogValve

Name	Data Type	Description
Invalid	Bool	Invalid input given

7.2.4. HMI Icon Display

7.2.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

Analog Valve Horizontal	Analog Valve Vertical
	

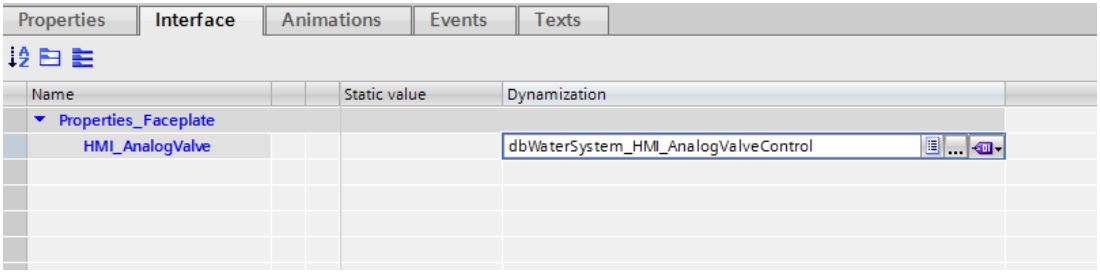
7.2.4.2. HMI Icon Appearance and Functionality

The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Device Color	Green: Opening Red: Error state
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

7.2.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_AnalogValveControl property. The HMI_AnalogValveControl needs to be mapped to the same 'udtHMI_AnalogValveControl' used by the Function Block.

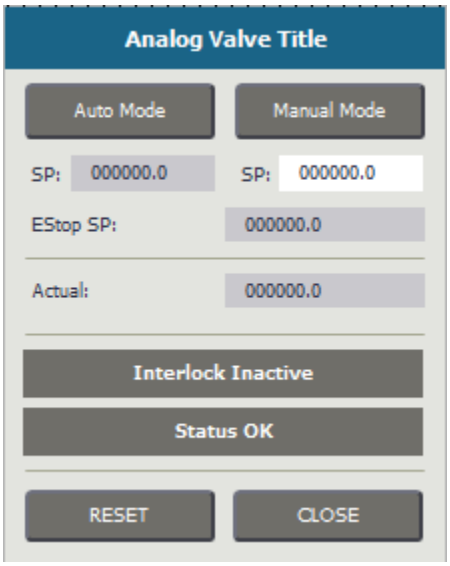


7.2.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the valve. Additionally, it allows for control on the valve when the System Mode is Independent or Manual.

7.2.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



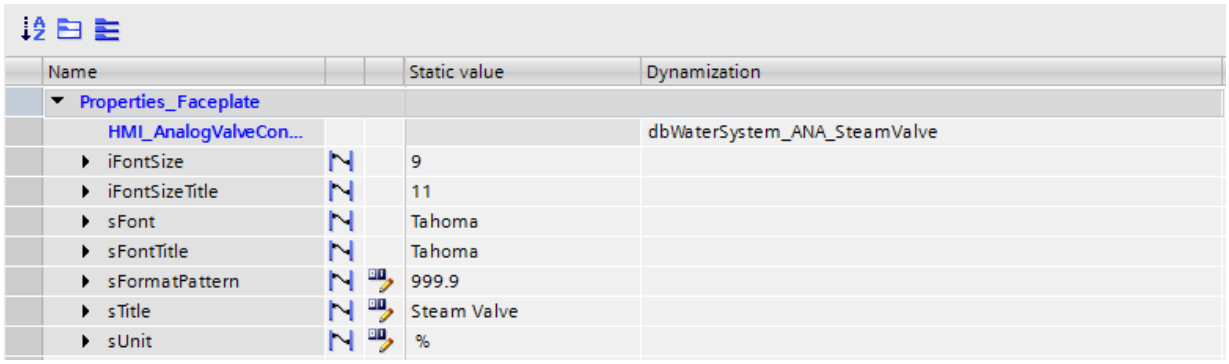
7.2.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Auto Mode SP	Output	Displays PLC setpoint to be used in Auto Mode.
Manual Mode SP	Input	Adjusts the speed set point when in Manual Mode
Estop SP	Output	Displays the current speed of the motor
Actual	Output	Displays ratio of valve opened
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

7.2.5.3. HMI Pop-up Interface

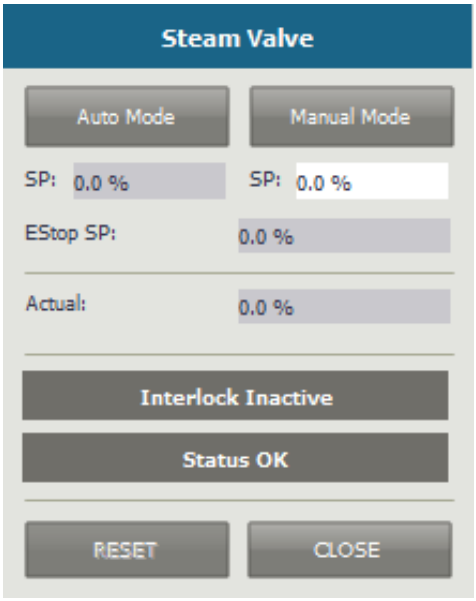
The device interface is shown below. Eight properties are linked in this interface: HMI_AnalogValveControl, iFontSize, iFontSizeTitle, sFont, sFontTitle, sFormatPattern, sTitle, and sUnit. The HMI_MotorControl needs to be mapped to the same 'udtHMI_Motor_Control' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sFormatPattern property determines how the setpoint and actual values are displayed. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sUnit property controls the unit that is displayed on the setpoint and actual values.



7.2.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

7.2.6.1. Screen



7.2.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_ANA_SteamValve"

2 = Solenoid Analog_Popup_Pro

3 = The desired title for the pop-up. E.g. "Steam Valve"

```
Sub OnClick(ByVal item)
SetPropertyByConstant AccessPath, "Screen window_1", "TagPrefix", "**1**"
SetPropertyByConstant AccessPath, "Screen window_1", "Visible", "True"
ActivateScreenInScreenWindow AccessPath, "Screen window_1", "**2**"
HMIRuntime.Screens(AccessPath & ".Screen window_1:**2**").ScreenItems("Title").Text = "**3**"
End Sub
```

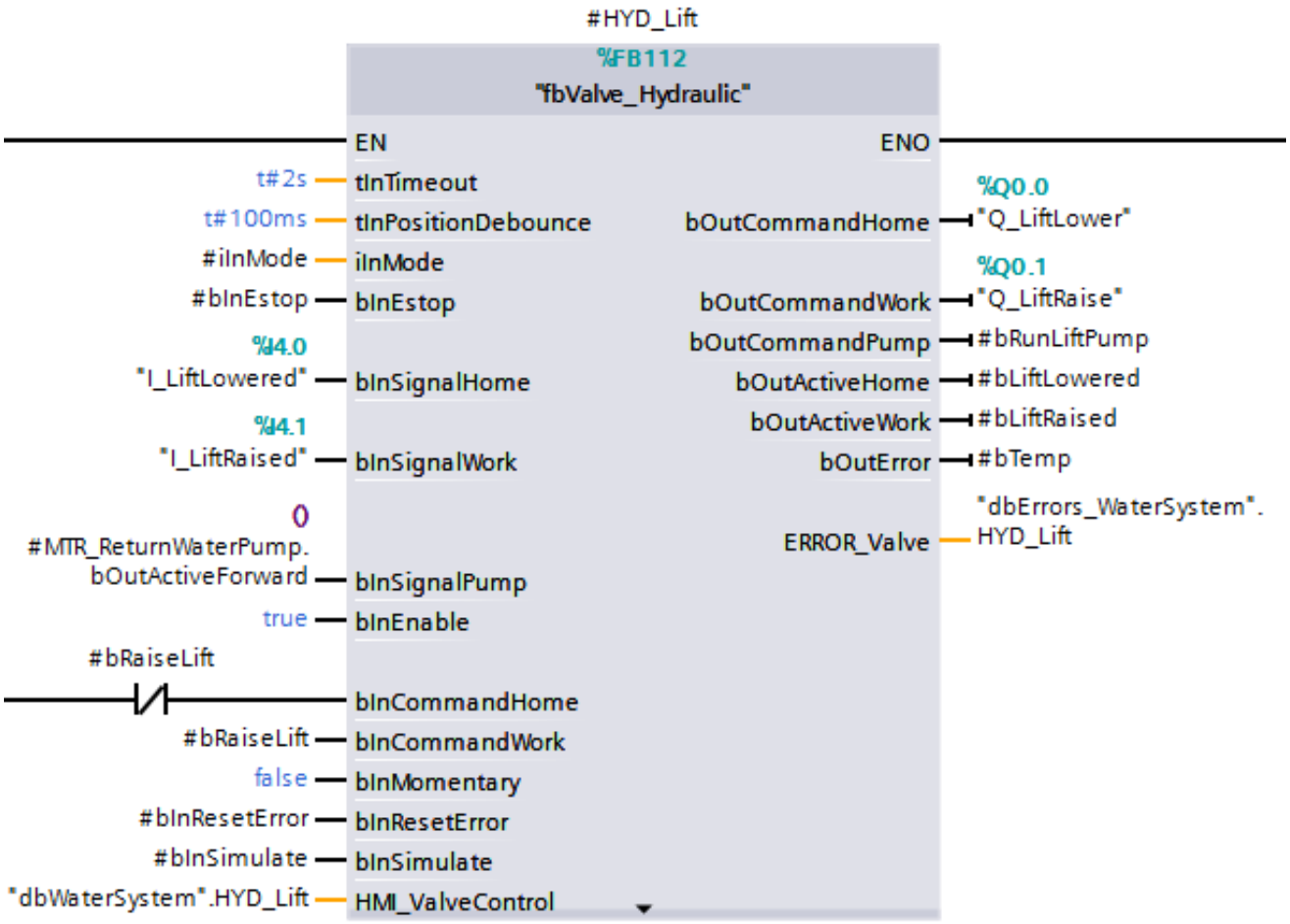
7.2.7. SiVarc Implementation

Please see SiVArc Implementation under Two State Solenoid Valve. The implementation for Analog Valve is the same.

7.3. Hydraulic Valve – fbValve_Hydraulic

7.3.1. Description

This Valve Hydraulic Function block is to be utilized with two state hydraulic valve applications. It can be used for momentary or non-momentary valves, with or without directly controlling the system's pump.



7.3.2. Function Block Interface

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
tInPositionDebounce	Time	Debounce time between when signal work or home is received, and work or home output is set low, assuming bInMomentary is false. Set to 0 to disable.
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInSignalHome	Bool	Home position feedback
bInSignalWork	Bool	Work position feedback
bInSignalPump	Bool	Feedback that the hydraulic pump is running.

bInEnable	Bool	Valve interlock. The valve is allowed to operate if this condition has been met
bInCommandHome*	Bool	Move to work position in automatic mode
bInCommandWork*	Bool	Move to work position in automatic mode
bInMomentary	Bool	Allows the cylinder to continue to move past the home or work position.
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

7.3.2.1. InOut Parameters

In/Out Variables	Type	Description
HMI_ValveControl	udtHMI_ValveControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the cylinder in manual mode

7.3.2.2. Out Parameters

Output Variables	Type	Description
bOutCommandHome	Bool	Output for home position
bOutCommandWork	Bool	Output for work position
bOutActiveHome	Bool	Home position is active
bOutActiveWork	Bool	Work position is active
bOutPump	Bool	Output for the pump that drives the valve, if it is not already always on.
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the cylinder
ERROR_Valve	udtError_Valve	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

7.3.3. User Defined Types

7.3.3.1. udtError_Valve

Errors	Description
NoHomeFeedback	Home position feedback not active
NoWorkFeedback	Work position feedback not active

HomeFeedbackStillActive	Home position feedback still active when commanded to Work Position
WorkFeedbackStillActive	Work position feedback still active when commanded to Home Position

7.3.3.2. udtHMI_ValveControl

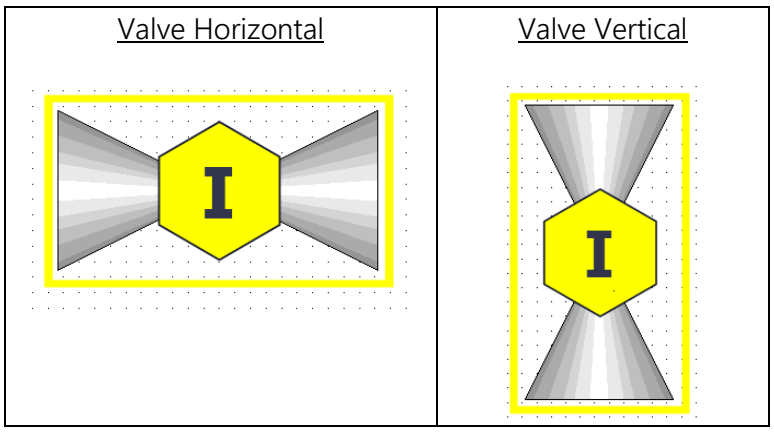
Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Home	Bool	Move to home in manual mode pushbutton
bPB_Work	Bool	Move to work in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Error pushbutton enabled
bPBEN_Home	Bool	Home pushbutton enabled
bPBEN_Work	Bool	Work pushbutton enabled
bHomeOn	Bool	Home command is on
bWorkOn	Bool	Work command is on
bSignalHome	Bool	Home feedback
bSignalWork	Bool	Work feedback
bError	Bool	Error status
bInterlock	Bool	Valve interlocked

7.3.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific applications.

7.3.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:



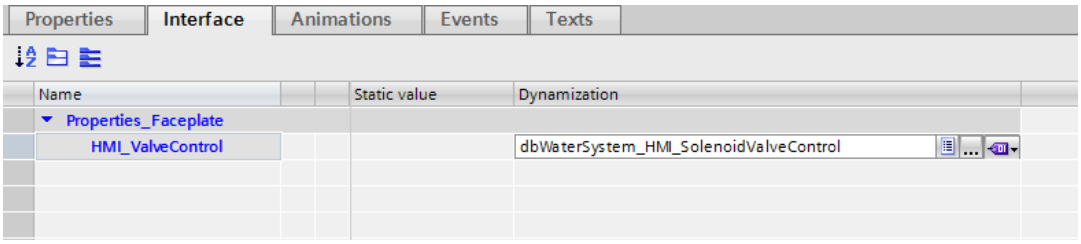
7.3.4.2. HMI Icon Appearance and Functionality

The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Device Color	Green: Open Yellow: Valve Opening/Closing Red Flashing: Error state Red: E-stop pressed
Yellow Border	Manual mode
Yellow Hexagon with 'I' in center	Interlocked

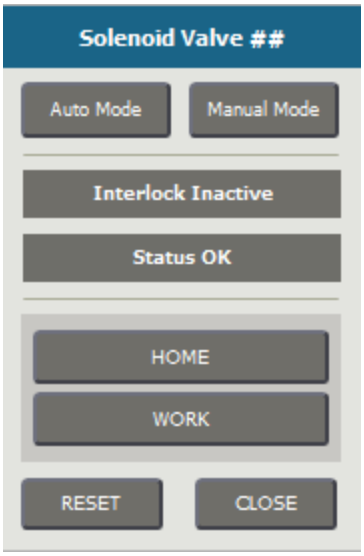
7.3.4.3. HMI Icon Interface

The device interface is shown below. It requires only one property linked, the HMI_ValveControl property. The HMI_ValveControl property requires a datablock of the 'udtHMI_ValveControl' type to be linked: in the example below it is dbWaterSystem_HMI_SolenoidValveControl.



7.3.5. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



7.3.5.1. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when device is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms
HOME Button	Button	Sets valve to HOME position
WORK Button	Button	Sets valve to WORK position
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

7.3.5.2. HMI Pop-up Interface

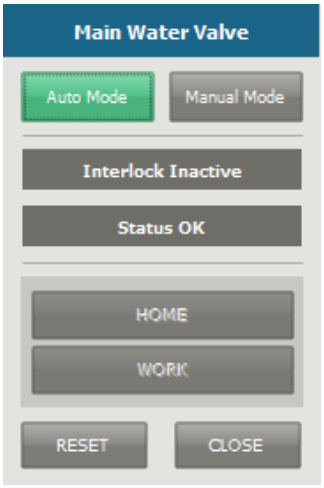
The device interface is shown below. Eight properties are linked in this interface: HMI_ValveControl, iFontSize, iFontSizeTitle, sButtonButtonText, sFont, sFontTitle, sTitle, and sTopButtonText. The HMI_ValveControl needs to be mapped to the same 'udtHMI_Valve_Control' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sBottomButtonText controls the text displayed on the Work button. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sTopButtonText controls the text displayed on the Home button.

Name	Static value	Dynamization
Properties_Faceplate		
HMI_ValveControl		dbWaterSystem_SOL_MainWater
iFontSize	9	
iFontSizeTitle	11	
sBottomButtonText	WORK	
sFont	Tahoma	
sFontTitle	Tahoma	
sTitle	Main Water Valve	
sTopButtonText	HOME	

7.3.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

7.3.6.1. Screen



7.3.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_SOL_MainWaterValve"

2 = Valve_Solenoid_Popup_Pro

3 = The desired title for the pop-up. E.g. "Main Water Valve"

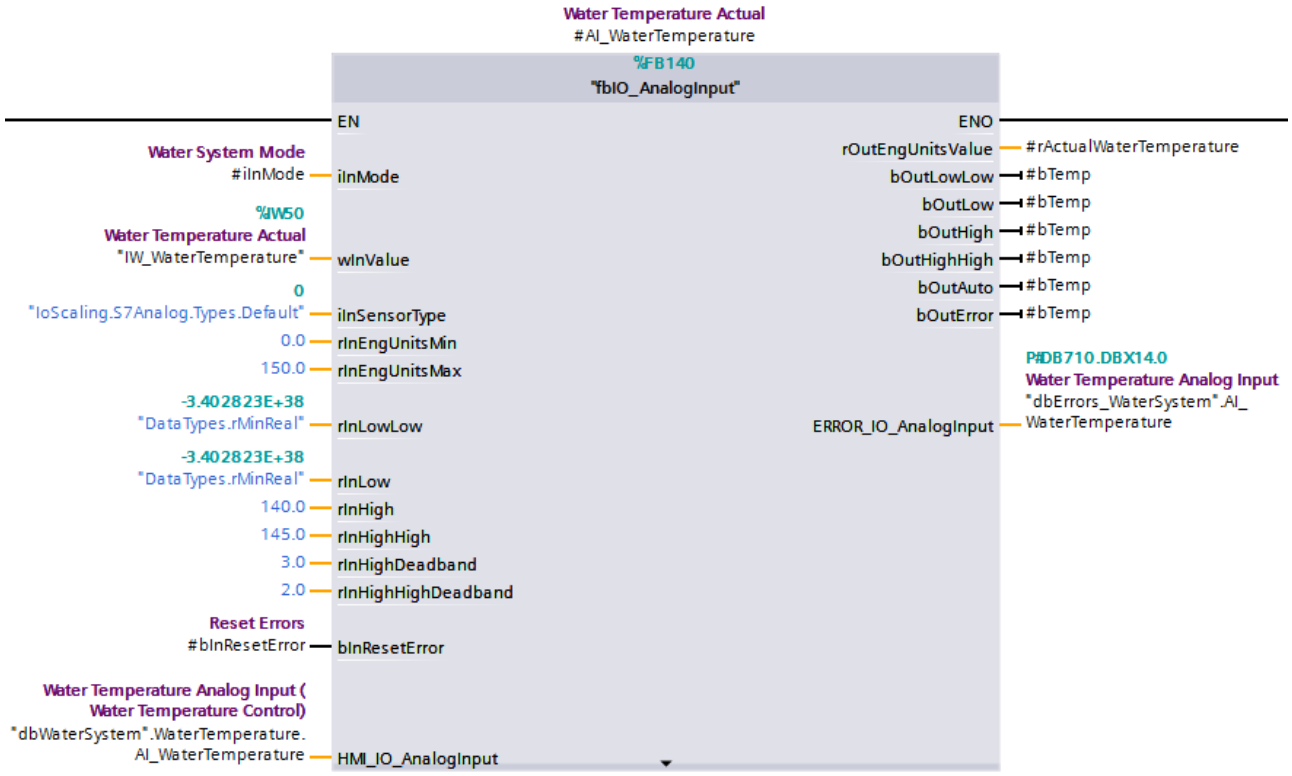
```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

8. Input/Output Control

8.1. Analog Input with Scaling and Alarms – fbIO_AnalogInput

8.1.1. Description

This library object scales analog inputs and provides setups for alarms.



8.1.2. Function Block Interface

8.1.2.1. Input Parameters

Input Variables	Type	Description
iInMode	Int	Mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
wInValue	Word	Raw input signal from hardware input
iInSensorType	Int	Sensor Type determines how the input should be scaled based on the type of sensor.
rInEngUnitsMin	Real	Minimum Value for Engineering Units
rInEngUnitsMax	Real	Maximum Value for Engineering Units
rInLowLow	Real	Value for Low Alarm to trigger. Use IoScaling.S7Analog.iRawMin to disable
rInLow	Real	Value for Low Warning to trigger. Use IoScaling.S7Analog.iRawMin to disable
rInHigh	Real	Value for High Warning to trigger. Use IoScaling.S7Analog.iRawMax to disable

rInHighHigh	Real	Value for High Alarm to trigger. Use IoScaling.S7Analog.iRawMax to disable
rInLowLowDeadband	Real	Deadband for Low Low Alarm. Use 0 to disable
rInLowDeadband	Real	Deadband for Low Warning. Use 0 to disable
rInHighDeadband	Real	Deadband for High Warning. Use 0 to disable
rInHighHighDeadband	Real	Deadband for High High Alarm. Use 0 to disable
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error

*Only valid in Automatic Mode

8.1.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_IO_AnalogInput	udtHMI_AnalogInput	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

8.1.2.3. Out Parameters

Output Variables	Type	Description
rOutEngUnitsValue	Real	Engineering units scaled value
bOutLowLow	Bool	Low low alarm is active
bOutLow	Bool	Low warning is active
bOutHigh	Bool	High warning is active
bOutHighHigh	Bool	High high alarm is active
bOutAuto	Bool	Input is in auto mode
bOutError	Bool	Error Exists
ERROR_IO_AnalogIn	udtError_AnalogInput	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

8.1.3. User Defined Types

8.1.3.1. udtHMI_AnalogInput

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details

iStatus	Int	Overall HMI Status. See Section 12.3.1 of the Library Overview and Architecture document for further details
rActiveValue	Real	Used Value, Input or Override
rInputValue	Real	Input Value
rManualValue	Real	Manual mode simulated value
bPB_ResetError	Bool	Reset errors pushbutton
bPBEN_ResetError	Bool	Reset error pushbutton enabled
bError	Bool	Overall error

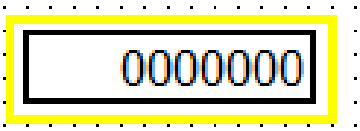
8.1.3.2. udtError_AnalogInput

Errors	Description
LowLowAlarm	Value below low low set point
HighHighAlarm	Value above high high set point
Invalid	Invalid input

8.1.4. HMI Icon Display

8.1.4.1. HMI Icon Display Object

The library contains the following display to be used on the HMI screen:



8.1.4.2. HMI Icon Appearance and Functionality

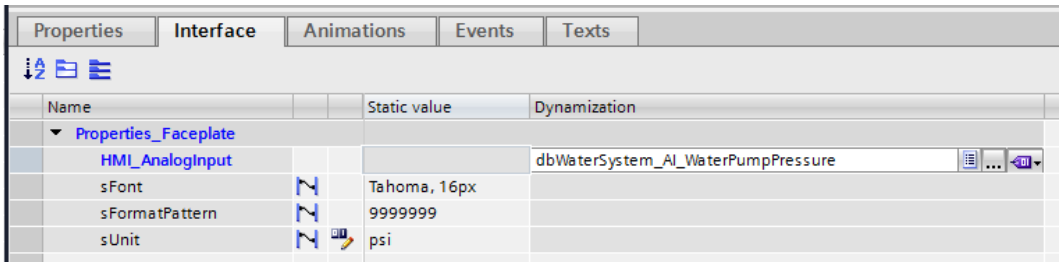
The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
I/O Field Value	Current value received from sensor
Yellow Rectangle	Manual Mode Active

8.1.4.3. HMI Icon Interface

The device interface is shown below. The following three properties need to be assigned to the object:

- HMI_AnalogInput: Control needs to be mapped to the same 'udtHMI_AnalogInput used by the Function Blocks
- sFont: Determines the font that data will be displayed in
- sFormatPattern: Sets the amount of leading/trailing zeros of the display value
- sUnit: Sets the unit the value is displayed in (i.e. amps, centimeters etc.)

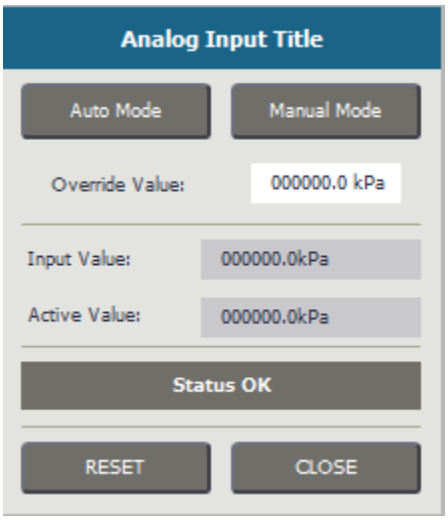


8.1.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the Analog Input device. Additionally, it allows for override control of the device reading when the System Mode is Independent or Manual.

8.1.5.1. Pop Up Object

The library contains a single pop-up faceplate shown below:



8.1.5.2. HMI Pop-up Appearance and Functionality

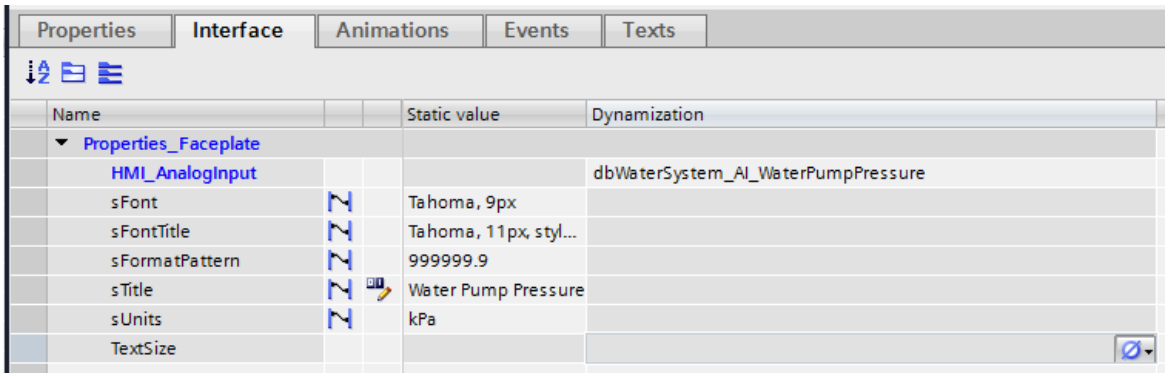
The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Override Value	Input	Adjusts the current value of the device if in manual mode
Input Value	Output	Displays the current reading of the device
Active Value	Output	Displays the value being currently used – the Override Value in Manual Mode, or Input Value in Auto Mode
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms

RESET Button	Button	Resets errors
EXIT Button	Button	Exits the pop up screen

8.1.5.3. Interface

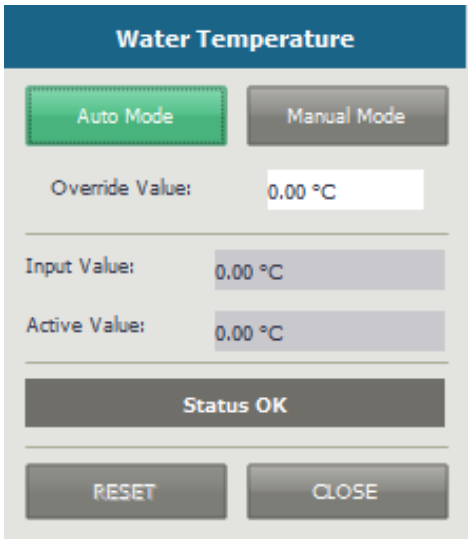
The device interface is shown below. Seven properties are linked in this interface: HMI_AnalogInput, sFont, sFontTitle, sFormatPattern, sTitle, sUnits, and Textsize. The HMI_AnalogInput needs to be mapped to the same 'udtHMI_AnalogInput_Control' used by the Function Block. The sFont property controls the font name and size used by text elements. The sFontTitle property controls the font name and size used by the title text. The sFormatPattern property determines how the values are displayed. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sUnits property controls the displayed unit. TeztSize does not do anything size font size is set in the sFont and sFontTitle properties.



8.1.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

8.1.6.1. Screen



8.1.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_AI_WaterTemperature"

2 = IO_AnalogInput_Popup_Pro

3 = The desired title for the pop-up. E.g. "Water Temperature"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

8.1.7. SiVArc Implementation

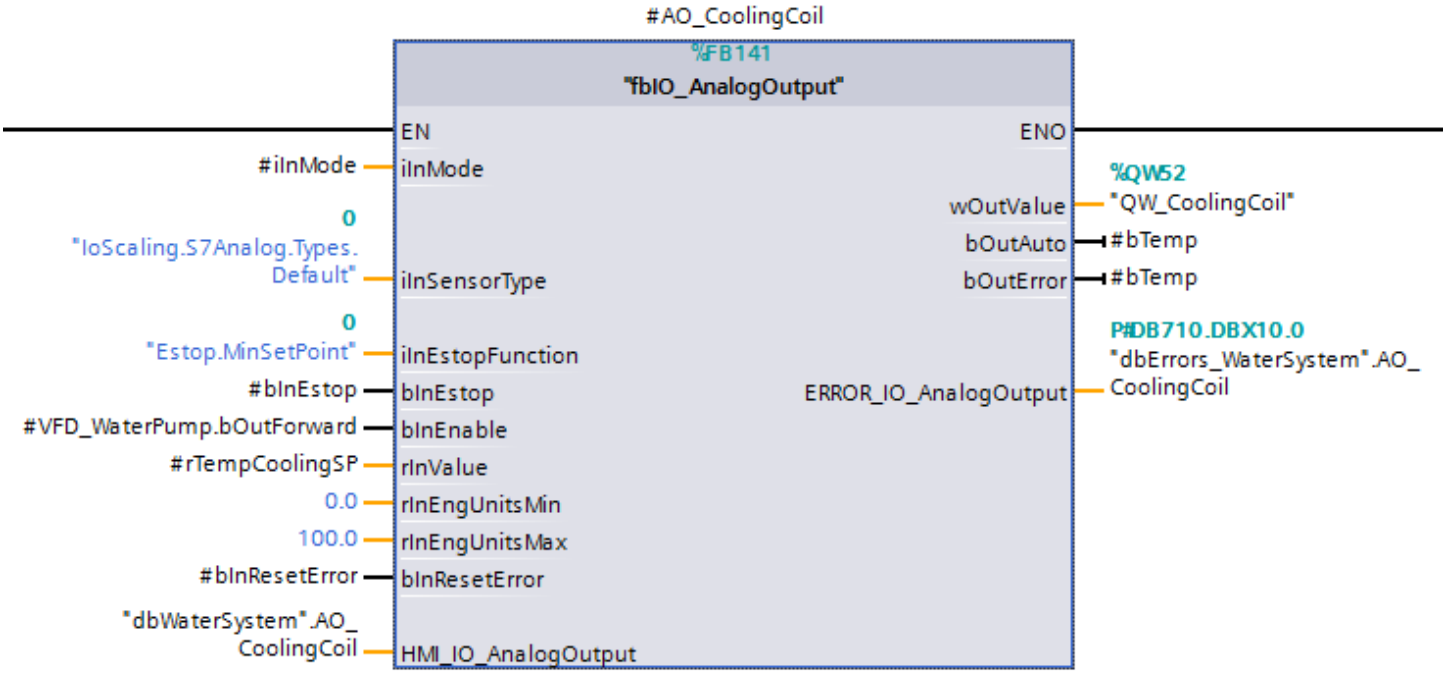
Two screen rules are defined for fbValve_Analog. One creates a pop-up that will be linked to the instance of the function block and to the created icon. The title on the pop-up is defined by NetworkTitle[0] for the network title on the instance of the function block. The other creates an icon linked to the function block. No other fields of the network title are used.

In WinCC Professional, there is no rule to create a pop-up, as there is not a separate pop-up screen for each object in the WinCC Pro implementation. Otherwise, the WinCC Professional rules function exactly as the rules for WinCC Comfort/Advanced.

8.2. Analog Output with Scaling – fbIO_AnalogOutput

8.2.1. Description

This library object scales process values to be used for Analog Outputs.



8.2.2. Function Block Interface

8.2.2.1. Input Parameters

Input Variables	Type	Description
iInMode	Int	Mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
iInSensorType	Int	Enumerated value of sensor types (IoScaling.S7Analog.Types)
iInEstopFunction	Int	Enumerated value of E-Stop response constant. 'Estop.MinSetPoint' sets wOutValue to the minimum set point (rInEngUnitsMax), 'Estop.MaxSetPoint' sets to the maximum set point (rInEngUnitsMin), and 'Estop.HMISetPoint' sets to an HMI-specified value that can be in between the min and max.
bInEstop	Bool	Emergency stop input
bInEnable	Bool	Interlock for the Analog Output. If this bit is false, then output will be 0.
rInValue*	Real	Auto mode value of output
rInEngUnitsMin	Real	Engineering units scale min
rInEngUnitsMax	Real	Engineering units scale max
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error

*Only valid in Automatic Mode

8.2.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_IO_AnalogOutput	udtHMI_AnalogOutput	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

8.2.2.3. Out Parameters

Output Variables	Type	Description
bOutAuto	Bool	Block in auto mode
bOutError	Bool	Error exists
wOutValue	Word	Value to output to Analog Output
ERROR_IO_AnalogOut	udtError_AnalogOut	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

8.2.3. User Defined Types

8.2.3.1. udtHMI_AnalogOutput

Name	Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
rValue	Real	Output Value
rAutoValue	Real	Auto mode value
rManualValue	Real	Manual mode value
rEstopValue	Real	Estop condition custom value
bPB_ResetError	Bool	Reset errors pushbutton
bPBEN_ResetError	Bool	Reset error pushbutton enabled
bError	Bool	Overall error
bInterlock	Bool	Analog output is interlocked
iEstopFunction	Int	Enumerated value of E-Stop response constant.

8.2.3.2. udtError_AnalogOutput

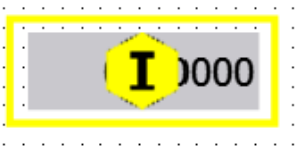
Errors	Description
Out_Of_Range	Input Value is outside range of Engineering Units
Configuration_Error	Engineering Units are not properly configured

8.2.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific applications.

8.2.4.1. HMI Icon Display Objects

The library contains the following object to be used on the HMI screen:



8.2.4.2. HMI Icon Appearance and Functionality

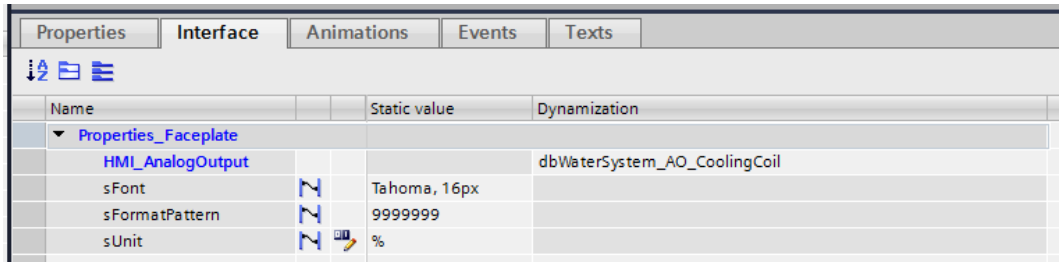
The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
I/O Field Value	Current value sent to device
Yellow Rectangle	Manual Mode Active
Yellow Hexagon	Device interlocked

8.2.4.3. HMI Icon Interface

The device interface is shown below. It requires three properties to be linked:

- HMI_Analog_Output: Needs to be mapped to the same 'udtHMI_AnalogOutput' used by the Function Block
- sFont: Determines font properties for the displayed output
- sFormatPattern: Changes data format of values, setting amount of leading and trailing zeros of outputs. Adjusted directly in table
- sUnit: Sets the units used in the display

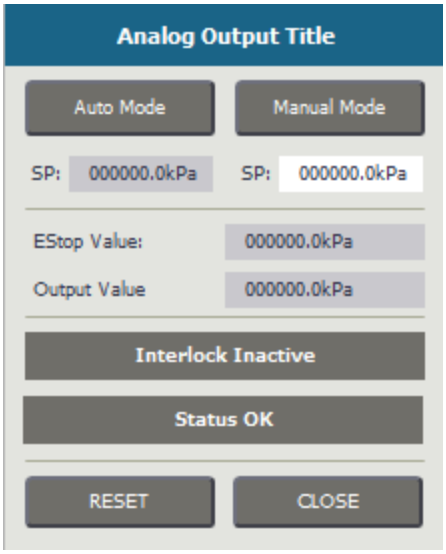


8.2.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the Analog Output device. Additionally, it allows for control of the device when the System Mode is Independent or Manual.

8.2.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



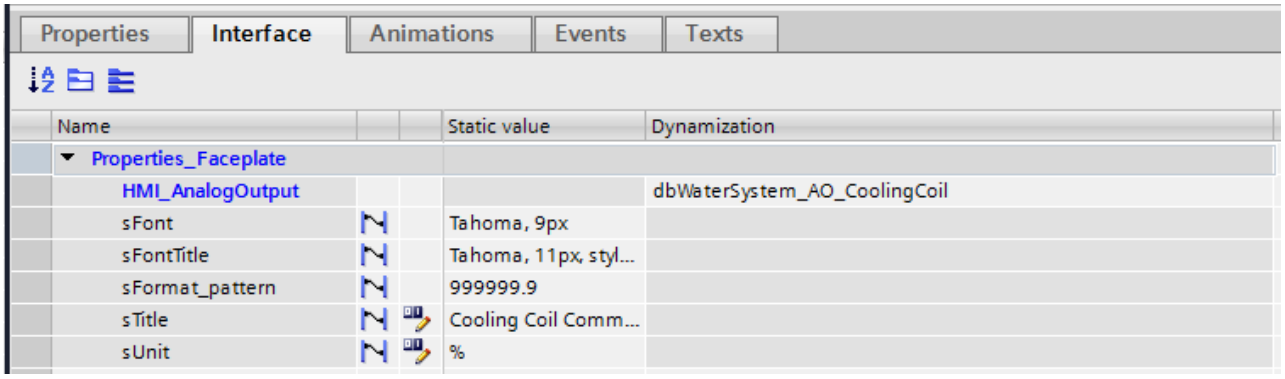
8.2.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Auto Mode SP	Output	Displays PLC set point to be used in Auto Mode.
Manual Mode SP	Input	Adjusts the speed set point when in Manual Mode
EStop Value	I/O Field	Displays the current value of the Estop reading
Interlock Status Field	I/O Field	Displays current status of device interlock (yellow for interlock, grey when not)
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

8.2.5.3. HMI Pop-up Interface

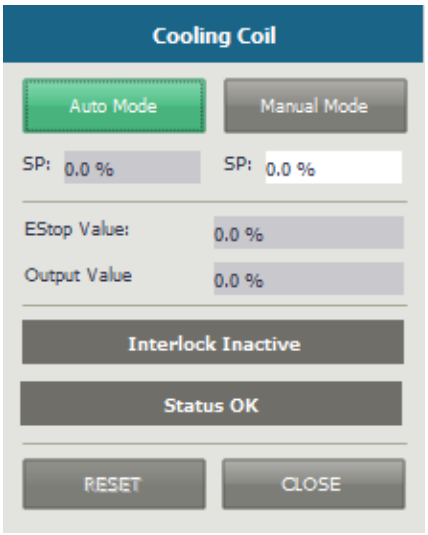
The device interface is shown below. Six properties are linked in this interface: HMI_AnalogInput, sFont, sFontTitle, sFormatPattern, sTitle, and sUnits. The HMI_AnalogInput needs to be mapped to the same 'udtHMI_AnalogInput_Control' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The s FormatPattern property determines how the values are displayed. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sUnits property controls the displayed unit.



8.2.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

8.2.6.1. Screen



8.2.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_AO_CoolingCoil"

2 = IO_AnalogOutput_Popup_Pro

3 = The desired title for the pop-up. E.g. "Cooling Coil"

```
Sub OnClick(ByVal item)
SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"
SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"
ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"
HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"
End Sub
```

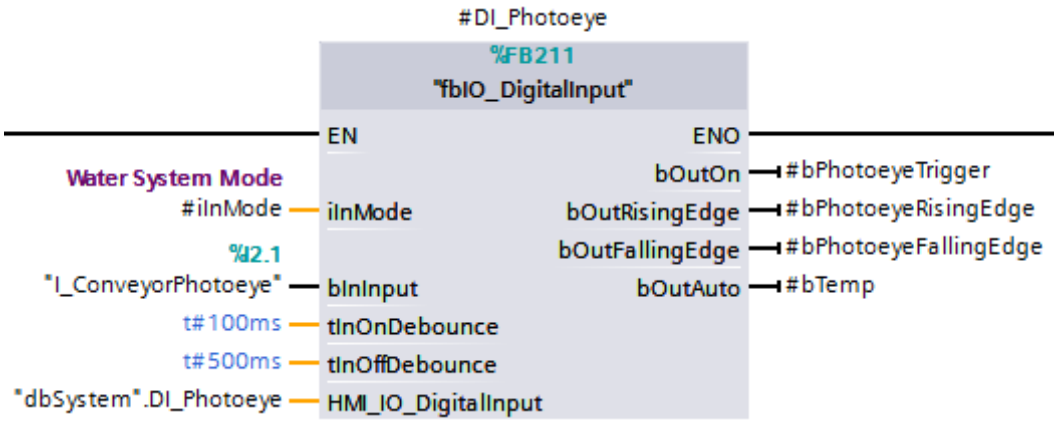
8.2.7. SiVarc Implementation

See section 8.1.7.

8.3. Digital Input – fbIO_DigitalInput

8.3.1. Description

This library object reads a digital input and allows manual control from the HMI.



8.3.2. Function Block Interface

8.3.2.1. Input Parameters

Input Variables	Type	Description
iInMode	Int	Mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInInput	Bool	The raw digital input signal
bInEnable	Bool	Interlock for the Analog Output. If this bit is false, then output will be 0.
tInOnDebounce	Time	Optional debounce timer for rising edge of input (T#0ms to disable).

tInOffDebounce	Time	Optional debounce timer for falling edge of input (T#0ms to disable).

8.3.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_IO_DigitalInput	udtHMI_Digital Input	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

8.3.2.3. Out Parameters

Output Variables	Type	Description
bOutAuto	Bool	Block in auto mode
bOutOn	Bool	The status of the digital input
bOutRisingEdge	Bool	The status of the input transitioned from low to high. True for exactly 1 scan
bOutFallingEdge	Bool	The status of the input transitioned from high to low. True for exactly 1 scan

8.3.3. User Defined Types

8.3.3.1. udtHMI_DigitalInput

Name	Type	Description
iMode	Int	Current mode
bOn	Bool	The status of the digital input block
bOnActual	Bool	The status of the physical digital input, ignoring manual overrides
bPB_On	Bool	Command on override pushbutton
bPB_Off	Bool	Command off override pushbutton
bPBEN_On	Bool	Command on override pushbutton enabled
bPBEN_Off	Bool	Command off override pushbutton enabled

8.3.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific applications.

8.3.4.1. HMI Icon Display Objects

The library contains the following object to be used on the HMI screen:

8.3.4.2. HMI Icon Appearance and Functionality

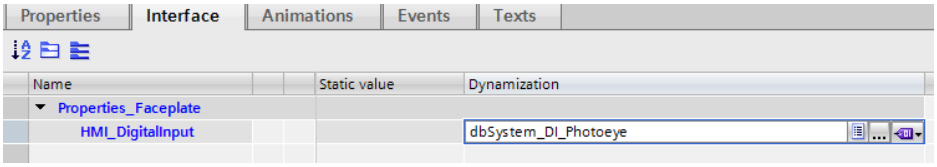
The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Text Field Visible	Current status of input
Yellow Border	Device in manual mode

8.3.4.3. HMI Icon Interface

The device interface is shown below. It requires three properties to be linked:

- HMI_DigitalInput: Needs to be mapped to the same 'udtHMI_DigitalInput' used by the Function Block



8.3.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the digital input device. Additionally, it allows for control of the device when the System Mode is Independent or Manual.

8.3.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:

8.3.5.2. HMI Pop-up Appearance and Functionality

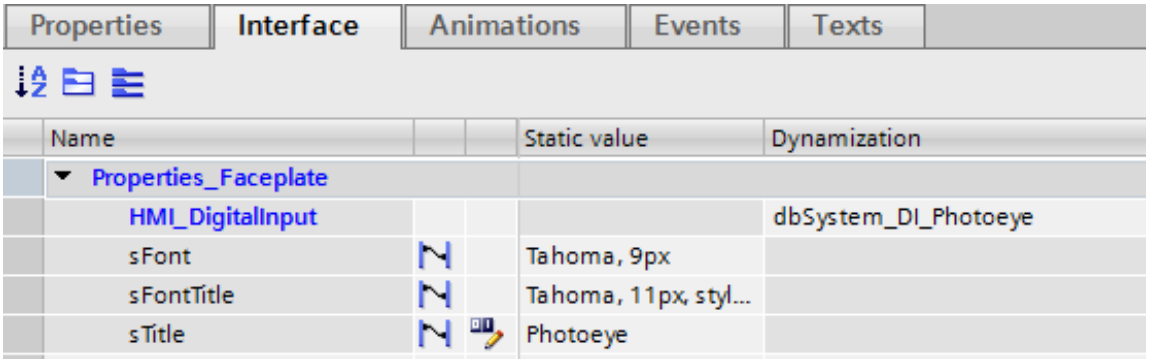
The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Override Buttons	Input	Sets the status of the digital input when in Manual Mode.

Active Value	Output	The status of the digital input block.
Raw Value	Output	The value of the raw physical digital input ignoring debouncing and Manual Mode overrides.
CLOSE Button	Button	Exits the pop up screen

8.3.5.3. HMI Pop-up Interface

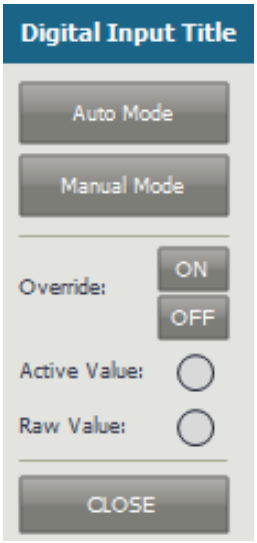
The device interface is shown below. Four properties are linked in this interface: HMI_DigitalInput, sFont, sFontTitle, and sTitle. The HMI_DigitalInput needs to be mapped to the same 'udtHMI_DigitalInput' used by the Function Block. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



8.3.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

8.3.6.1. Screen



8.3.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbSystem_DI_Photoeye"

2 = IO_DigitalInput_Popup_Pro

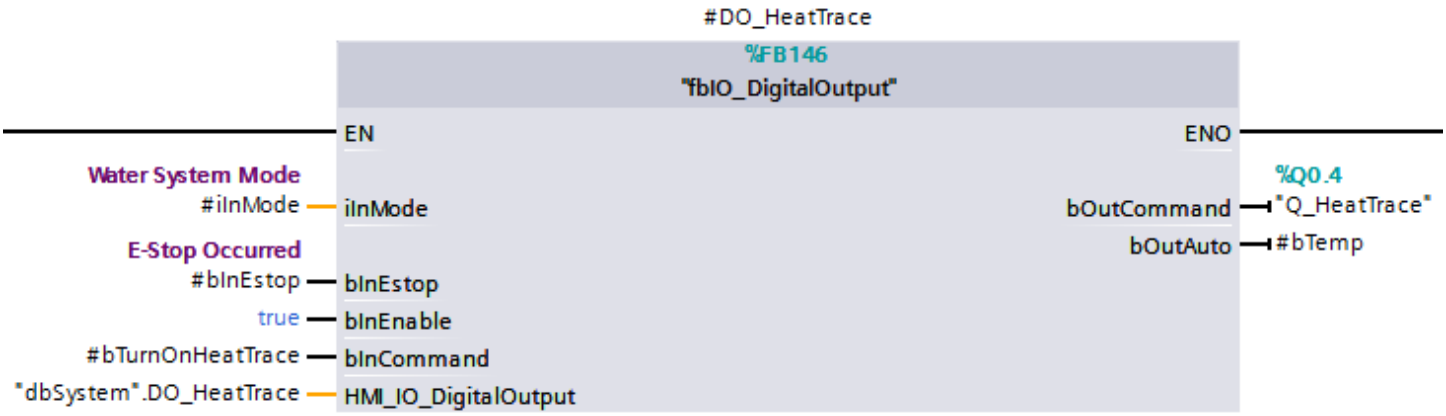
3 = The desired title for the pop-up. E.g. "Photoeye"

```
Sub OnClick(ByVal item)
SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"
SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"
ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"
HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"
End Sub
```

8.4. Digital Output – fbIO_DigitalOutput

8.4.1. Description

This library object controls a digital output and allows manual control from the HMI.



8.4.2. Function Block Interface

8.4.2.1. Input Parameters

Input Variables	Type	Description
iInMode	Int	Mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInEnable	Bool	Interlock for the digital output. If this bit is false, then output will be 0.
bInCommand*	Bool	The command to the digital output

*Only valid in Automatic Mode

8.4.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_IO_DigitalOutput	udtHMI_Digital Output	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

8.4.2.3. Out Parameters

Output Variables	Type	Description
bOutAuto	Bool	Block in auto mode
bOutCommand	Bool	Command to digital output

8.4.3. User Defined Types

8.4.3.1. udtHMI_DigitalOutput

Name	Type	Description
iMode	Int	Current mode
bCommand	Bool	Output command status
bCommandAuto	Bool	The output command from the PLC, ignoring manual overrides
bInterlock	Bool	Digital output is interlocked
bPB_On	Bool	Command on override pushbutton
bPB_Off	Bool	Command off override pushbutton
bPBEN_On	Bool	Command on override pushbutton enabled
bPBEN_Off	Bool	Command off override pushbutton enabled

8.4.4. HMI Icon Display

The HMI Icon is used on the main HMI screen to provide a brief overview of the status of the device. Custom devices can be made to match specific applications.

8.4.4.1. HMI Icon Display Objects

The library contains the following object to be used on the HMI screen:

8.4.4.2. HMI Icon Appearance and Functionality

The device appearance's changes depending on the state it is in, see table below for description:

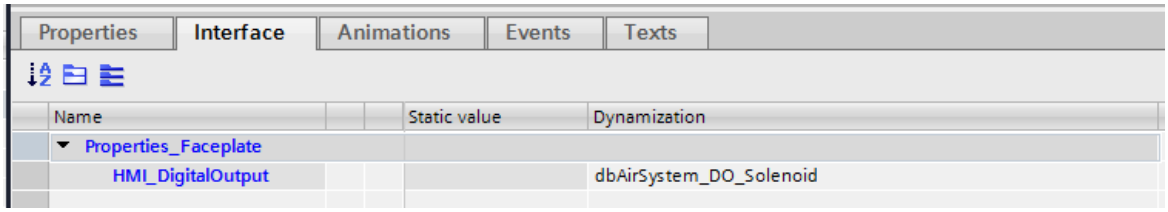
Indicator	Meaning
Text Field Visible	Current status of output

Yellow Border	Device in manual mode

8.4.4.3. HMI Icon Interface

The device interface is shown below. It requires three properties to be linked:

- HMI_DigitalOutput: Needs to be mapped to the same 'udtHMI_DigitalOutput' used by the Function Block

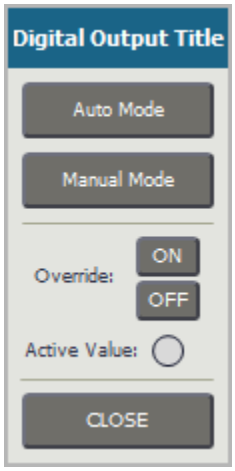


8.4.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the digital input device. Additionally, it allows for control of the device when the System Mode is Independent or Manual.

8.4.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



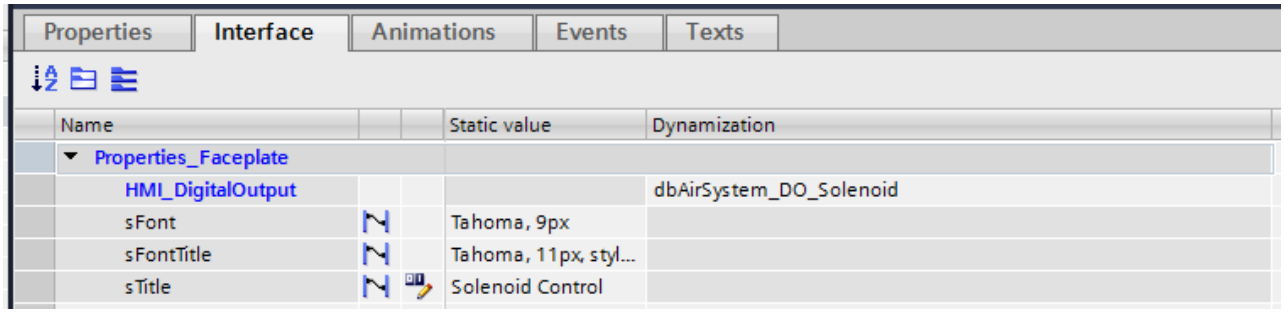
8.4.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Override Buttons	Input	Sets the status of the digital output when in Manual Mode.
Active Value	Output	The status of the digital output block.
CLOSE Button	Button	Exits the pop up screen

8.4.5.3. HMI Pop-up Interface

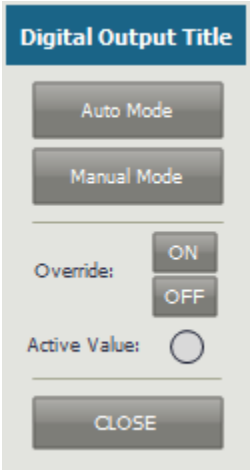
The device interface is shown below. Four properties are linked in this interface: HMI_DigitalInput, sFont, sFontTitle, and sTitle. The HMI_DigitalOutput needs to be mapped to the same 'udtHMI_DigitalOutput' used by the Function Block. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



8.4.6. WinCC Professional Pop-up Screen

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

8.4.6.1. Screen



8.4.6.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbSystem_DO_Solenoid"

2 = IO_DigitalOutput_Popup_Pro

3 = The desired title for the pop-up. E.g. "Solenoid"

9. General Control

9.1. System Control

9.1.1. Description

The System Control Library Object consists of a UDT, HMI Icon, and HMI Faceplate.

9.1.2. User Defined Types

9.1.2.1. udtHMI_SystemControl

Name	Data Type	Description
iMode	Int	System mode
bEStop	Bool	System hardware E-Stop
bError	Bool	Actuators error present
bWarning	Bool	Warning present
bAuto	Bool	All actuators in auto mode
bPB_ResetError	Bool	Reset error push button

9.1.3. HMI Icon Display

9.1.3.1. HMI Icon Display Objects

The library contains the following object to be used on the HMI screen:

9.1.3.2. HMI Icon Appearance and Functionality

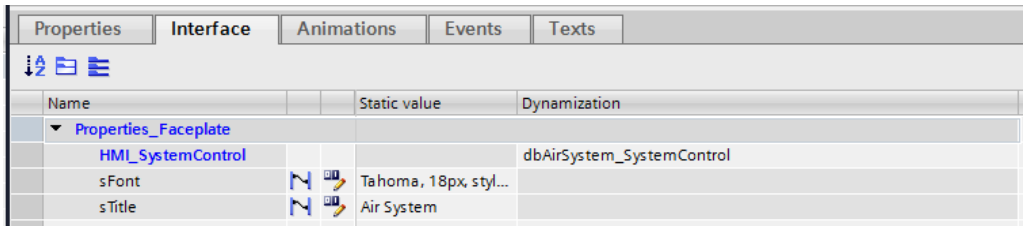
The status text and color updates according to the current running state of the device:

Device Mode	System Color
Stop	Grey
Auto	Green
Manual	Blue
Independent	Yellow

9.1.3.3. HMI Icon Interface

The device interface is shown below. The following properties are required to be linked:

- HMI_SystemMode: Needs to be mapped to the 'udt_HMI_SystemControl' for the given system.
- sFont: Determines the font and font size of the icon.
- sTitle: Sets title of the pop up display. Adjusted directly in table.

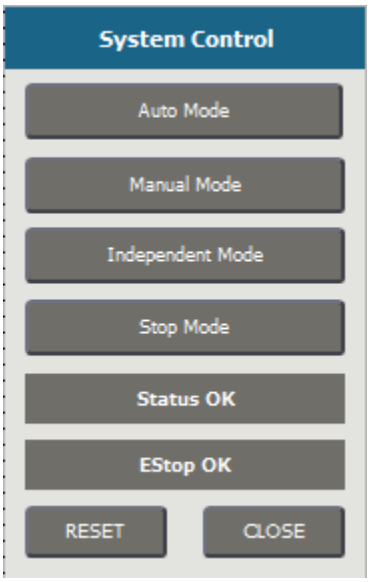


9.1.4. HMI Pop-up Faceplate

The HMI pop up provides a display of the system's status. Additionally, it provides operator control for the overall system mode.

9.1.4.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



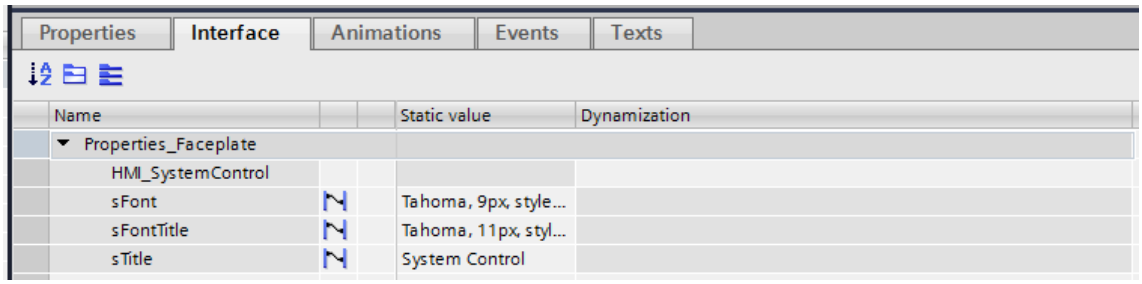
9.1.4.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode	Button	Sets system to Auto Mode
Manual Mode	Button	Sets system to Manual Mode
Independent Mode	Button	Sets system to Independent Mode
Stop Mode	Button	Sets system to Stop Mode
Interlock Status Field	I/O Field	Displays current system alarm status (red for alarm, grey for System Ok)
E-Stop Status Field	I/O Field	Displays current E-stop status. Field box turns red if an E-stop error is present, and displays active error text.
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

9.1.4.3. HMI Icon Interface

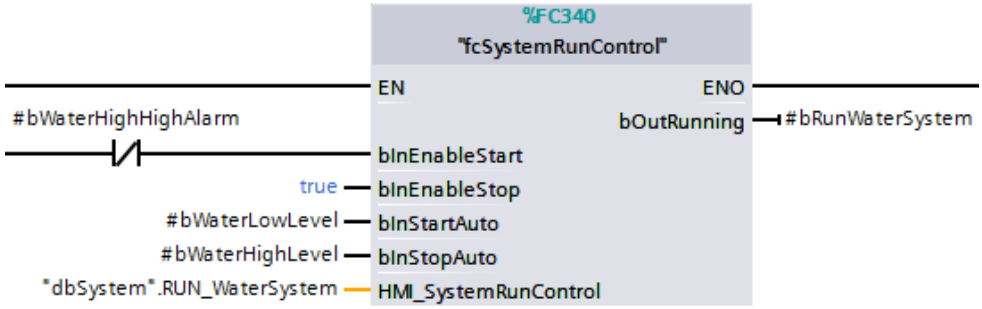
The device interface is shown below. Four properties are linked in this interface: HMI_SystemControl, sFont, sFontTitle, and sTitle. The HMI_ SystemControl needs to be mapped to the same 'udtHMI_System_Control' used by the Function Block. The sFont property controls the font name and size used by text elements. The sFontTitle property controls the font name used by the title text. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



9.2. Run Control

9.2.1. Description

This library object provides simple start/stop logic from both the PLC and HMI. It also allows for interlocking on certain states and gives feedback for control and display.



9.2.2. Function Block Interface

9.2.2.1. Input Parameters

Input Variables	Type	Description
bInEnableStart	Bool	Interlocks the system from being able to start running
bInEnableStop	Bool	Interlocks the system from being able to stop
bInStartAuto	Bool	Starts the system running in auto mode
bInStopAuto	Bool	Stop the system in auto mode

9.2.2.2. InOut Parameters

In/Out Variables	Type	Description

HMI_SystemRunControl	udtHMI_System RunControl	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

9.2.2.3. Out Parameters

Output Variables	Type	Description
bOutRunning	Bool	System is running

9.2.3. User Defined Types

9.2.3.1. udtHMI_SystemRunControl

Name	Data Type	Description
bRunning	Bool	System is running
bPB_Start	Bool	Start command pushbutton
bPB_Stop	Bool	Stop command pushbutton
bPBEN_Start	Bool	Start command pushbutton enable
bPBEN_Stop	Bool	Stop command pushbutton enable

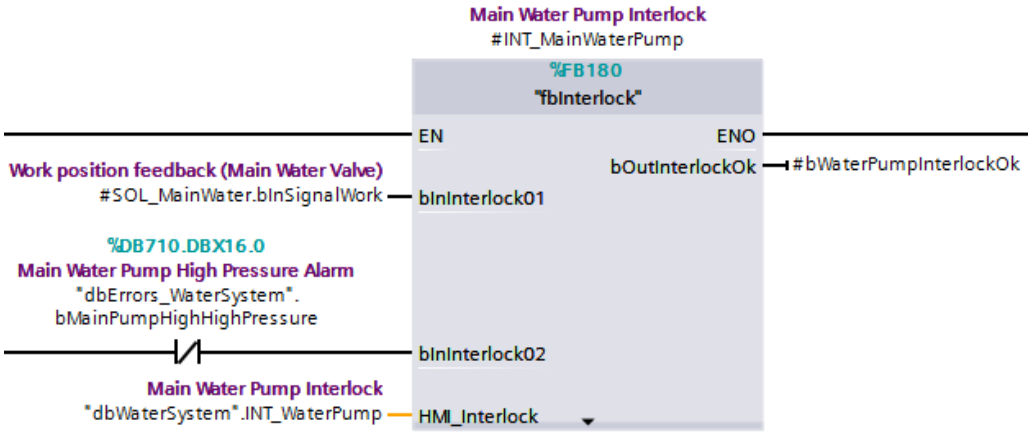
9.2.4. HMI Display

The system run control does not currently have a provided HMI implementation

9.3. Interlock – fbInterlock

9.3.1. Description

This library object is used to provide operator information for multiple values that are interlocking a process. It has inputs for items that may be interlocking a process, and provides operator information to inform the operator of what interlocks are preventing operation.



9.3.2. Function Block Interface

9.3.2.1. Input Parameters

Input Variables	Type	Description
bInInterlock_01	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_02	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_03	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_04	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_05	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_06	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_07	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_08	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_09	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_10	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_11	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_12	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_13	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_14	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_15	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInInterlock_16	Bool	Interlock for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True

9.3.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_Interlock	udtHMI_Interlock	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

9.3.2.3. Out Parameters

Output Variables	Type	Description
bOutInterlockOK	Bool	Value of 1 means system is okay to operate

9.3.3. User Defined Types

9.3.3.1. udtHMI_Interlock

Name	Data Type	Description
abInterlocks	Array[1..16] of Bool	Interlock statuses
asInterlockNames	Array[1..16] of String[20]	Interlock names
bInterlockOk	Bool	System Interlock is ok

9.3.4. HMI Icon Display

The Interlock block is configured so that the Faceplate can be tied to a faceplate of another object, and allow viewing from other library pop-ups.

9.3.4.1. HMI Icon Display Objects

The library contains the following display to be used on the HMI screen:

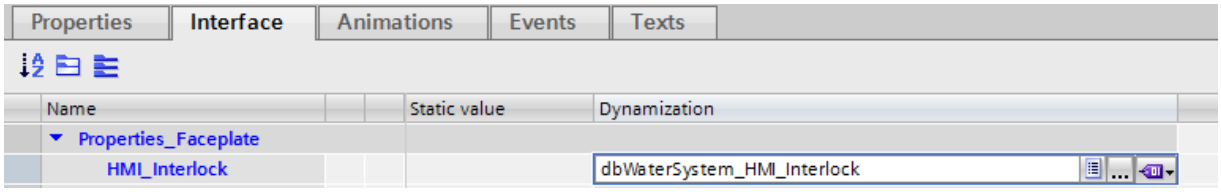
9.3.4.2. HMI Icon Appearance and Functionality

The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Visibility	Visible: Device interlocked Hidden: Device not interlocked

9.3.4.3. HMI Icon Interface

The device interface is shown below. Only one property is required to be linked: the HMI_Interlock property. It needs to be mapped to the same 'udtHMI_Interlock' used by the Function Block.



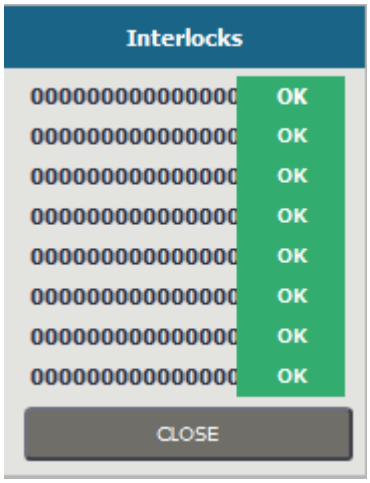
9.3.5. HMI Pop-up Faceplate

The HMI pop up provides a display of the status of each interlock in the system

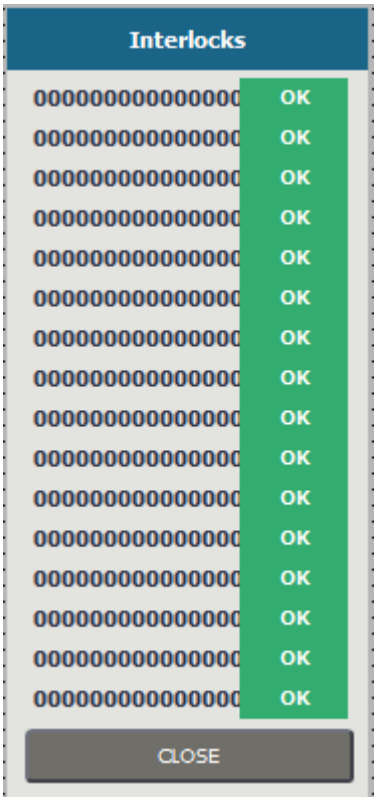
9.3.5.1. HMI Pop-up Display

The library contains two pop-up faceplates shown below:

Interlock8



Interlock16



Both faceplates use the same UDT- 'udtHMI_Interlock'. fpInterlock8 is to be used when you have 8 or fewer interlocks for a system.

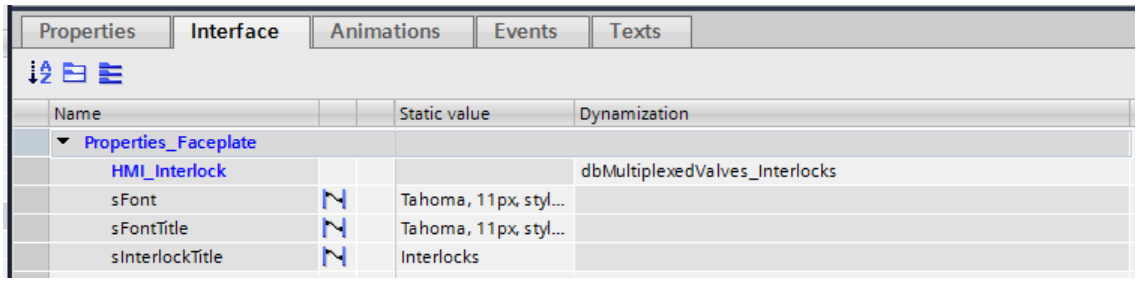
9.3.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop-up faceplate:

Object	Type	Description
Device Status Field	Output	Device OK: Displays 'OK' in a green box Device Interlocked, unable to run: Displays 'Interlock' displayed in Red Box
CLOSE Button	Button	Exits the pop-up screen

9.3.5.3. HMI Pop-up Interface

The device interface is shown below. Four properties are linked in this interface: HMI_Interlock, sFont, sFontTitle, and sTitle. The HMI_Interlock needs to be mapped to the same 'udtHMI_Interlock' used by the Function Block. The sFont property controls the font name and size used by text elements. The sFontTitle property controls the font name used by the title text. The sInterlockTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



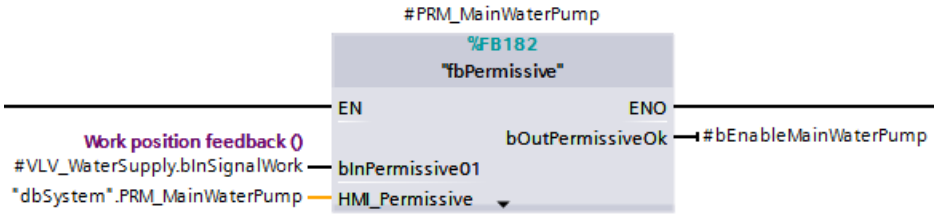
9.3.6. SiVarc Implementation

Three screen rules are defined for the object. One creates an icon for each instance of the function block, while the other two create pop-ups for each instance, depending on conditions. Which pop-up faceplate is used is dependent on whether bInInterlock09 is assigned. It is assumed that if it isn't assigned, there are 8 or fewer interlocks, and if it is assigned, there are 9 or more interlocks. Thus, if it isn't assigned, fpInterlock8_Popup is used, if it is assigned, fpInterlock16_Popup is used. NetworkTitle[0] is used for the title of the popup, no other parts of the network title are used.

9.4. In WinCC Professional, there is no rule to create a pop-up, as there is not a separate pop-up screen for each object in the WinCC Pro implementation. Otherwise, the WinCC Professional rules function exactly as the rules for WinCC Comfort/Advanced.Permissive – fbPermissive

9.4.1. Description

This library object is used to provide operator information for multiple values that are not allowing process to start. It has inputs for items that are a permissive for the process, and provides operator information to inform the operator of what permissives are not enabling the process to start.



9.4.2. Function Block Interface

9.4.2.1. Input Parameters

Input Variables	Type	Description
bInPermissive_01	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_02	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_03	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_04	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_05	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_06	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_07	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_08	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_09	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_10	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_11	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_12	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_13	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_14	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True

bInPermissive_15	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True
bInPermissive_16	Bool	Permissive for the system. When value is 1, system is okay to operate. Disable input by not mapping or putting in a value of True

9.4.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_Permissive	udtHMI_Permissive	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

9.4.2.3. Out Parameters

Output Variables	Type	Description
bOutPermissiveOK	Bool	Value of 1 means system is okay to operate

9.4.3. User Defined Types

9.4.3.1. udtHMI_Permissive

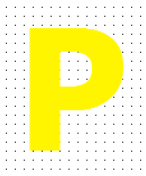
Name	Data Type	Description
abPermissives	Array[1..16] of Bool	Permissive statuses
asPermissiveNames	Array[1..16] of String[20]	Permissive names
bPermissiveOk	Bool	System Permissive is ok

9.4.4. HMI Icon Display

The Permissive block is configured so that the Faceplate can be tied to a faceplate of another object, and allow viewing from other library pop-ups.

9.4.4.1. HMI Icon Display Objects

The library contains the following display to be used on the HMI screen:



9.4.4.2. HMI Icon Appearance and Functionality

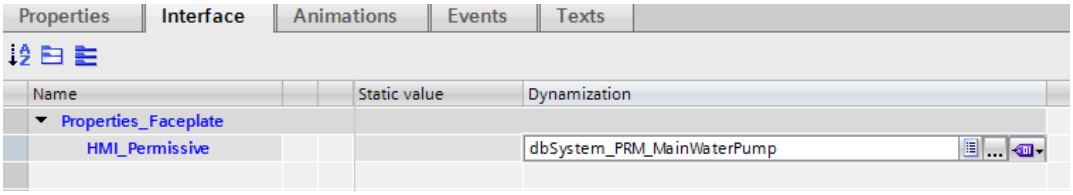
The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning

Visible	Visible: Permissives blocked Hidden: Permissives OK

9.4.4.3. HMI Icon Interface

The device interface is shown below. Only one property is required to be linked: the HMI_Permissive property. It needs to be mapped to the same 'udtHMI_Permissive' used by the Function Block.



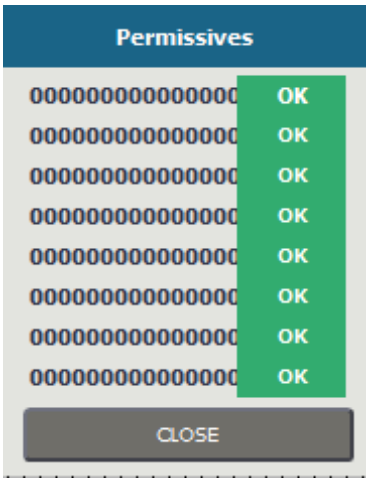
9.4.5. HMI Pop-up Faceplate

The HMI pop up provides a display of the status of each permissive in the system

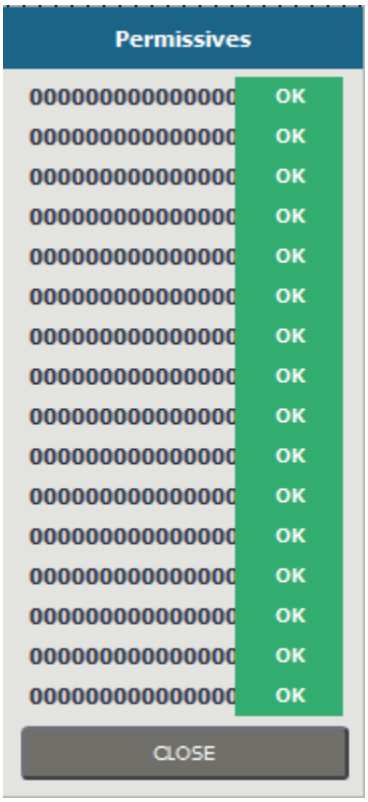
9.4.5.1. HMI Pop-up Display

The library contains two pop-up faceplates shown below:

Permissive8



Permissive16



Both faceplates use the same UDT- 'udtHMI_Permissive'. fpPermissive8 is to be used when you have 8 or fewer permissives for a single system.

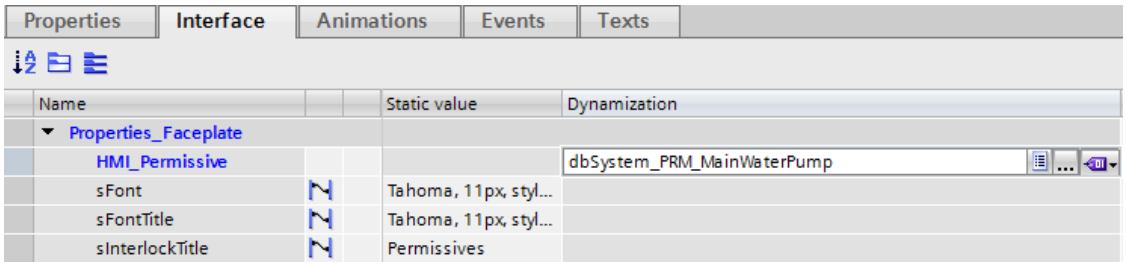
9.4.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop-up faceplate:

Object	Type	Description
Device Status Field	Output	Device OK: Displays 'OK' in a green box Device Interlocked, unable to run: Displays 'Interlock' displayed in Red Box
CLOSE Button	Button	Exits the pop-up screen

9.4.5.3. HMI Pop-up Interface

The device interface is shown below. Six properties are linked in this interface: HMI_Permissive, iFontSize, iFontSizeTitle, sFont, sFontTitle, and sTitle. The HMI_Permissive needs to be mapped to the same 'udtHMI_Interlock' used by the Function Block. The iFontSize property controls the font size of text elements. The iFontSizeTitle property controls the font size of the title text. The sFont property controls the font name used by text elements. The sFontTitle property controls the font name used by the title text. The sInterlockTitle property requires a string passed into it, and this value will display at the top of the Pop-up.



9.4.6. SiVarc Implementation

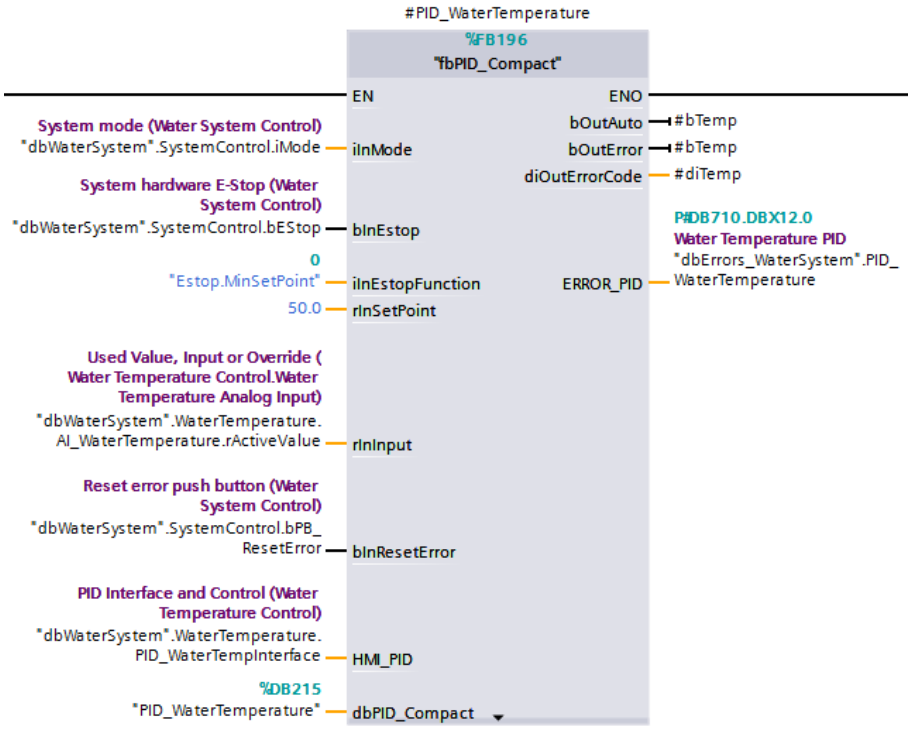
Three screen rules are defined for the object. One creates an icon for each instance of the function block, while the other two create pop-ups for each instance, depending on conditions. Which pop-up faceplate is used is dependent on whether bInPermissive_09 is assigned. It is assumed that if it isn't assigned, there are 8 or fewer permissives, and if it is assigned, there are 9 or more permissives. Thus, if it isn't assigned, fpPermissive8_Popup is used, if it is assigned, fpPermissive16_Popup is used. NetworkTitle[0] is used for the title of the popup, no other parts of the network title are used.

In WinCC Professional, there is no rule to create a pop-up, as there is not a separate pop-up screen for each object in the WinCC Pro implementation. Otherwise, the WinCC Professional rules function exactly as the rules for WinCC Comfort/Advanced.

9.5. PID Control - fbPID_Compact

9.5.1. Description

The S7-1200 and S7-1500 provide PID Technology objects that are best practices for use, therefore this would not replace those blocks, but would provide an interface to the technology object on the HMI as well as tuning, mode, and manual control functionality.



This block is used to handle HMI mode selection and control as well as scale inputs, handle e-stop conditions, and allow manual and automatic tuning from the HMI.

Since it contains the PID_Compact block inside of it, the fbPID_Compact block must be called from a cyclic interrupt OB.

9.5.2. Function Block Interface

9.5.2.1. Input Parameters

Input Variables	Type	Description
iInMode	Int	Sets the mode of the block. When the block is in auto mode, it utilizes rInSetpoint. When the block is in manual mode, it reads commands from the HMI through

		HMI_PID. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
iInEstopFunction	Int	Enumerated value of E-Stop response constant. 'Estop.MinSetPoint' sets Output to the minimum set point (PID output maximum), 'Estop.MaxSetPoint' sets to the maximum set point (PID output minimum), and 'Estop.HMISetPoint' sets to an HMI-specified value that can be in between the min and max.
rInSetpoint*	Real	PID set point
rInInput	Real	Scaled Process value (0-100%) Use this value or wInInputPer
wInInputPER	Word	Raw Process value. Use this value or rInInput
rInManualValue*	Real	Used by automation logic to override PID and provide a manual value through PLC logic
bInManualEnable*	Bool	rInManualValue is used to override the PID output value when this is true
iInPIDModeRequest	Int	PID_Compact Mode as described by the PID compact. Used by automation logic to enable the override of the PID_Compact mode
bInPIDModeActivate	Bool	PID_Compact Mode will change on a rising edge of this bit based on iInPIDModeRequest
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error

*Only valid in Automatic Mode

9.5.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_PID	udtHMI_PID	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the PID in manual mode
dbPID_Compact	PID_Compact v2.2	The data block of the technology object.

9.5.2.3. Out Parameters

Output Variables	Type	Description
bOutAuto	Bool	Returns if the PID interface is in auto mode.
bOutError	Bool	Returns if the PID interface has an error condition.
diOutErrorCode	Dint	Returns the error bits from the PID_Compact block. Can be used with ERROR_PID for debugging purposes.
ERROR_PID	udtError_PID	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

9.5.3. User Defined Types

9.5.3.1. udtHMI_PIDInterface

Name	Type	Description

iMode	Int	Current mode
iPIDMode	Int	PID compact mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iEstopFunction	Int	Enumerated value of E-Stop response constant.
rEstopOverride	Real	PID override value for an e-stop if custom value is selected
rManualOverride	Real	PID override value when in manual mode and override selected
rAutoOverride	Real	PID override value from the PLC in auto mode
rManualSP	Real	PID set point for manual mode
rAutoSP	Real	PID set point for automatic mode
rActualInput	Real	Actual PID input value
wActualInputPeripheral	Word	Actual PID input peripheral value
rOutput	Real	PID output value
wOutputPeripheral	Word	PID output peripheral value
rProportionalGain	Real	Proportional gain coefficient
rIntegralTime	Real	Integral control time
rDerivativeTime	Real	Derivative control time
rProportionalWeight	Real	Proportional control weighting
rDerivativeWeight	Real	Derivative control weighting
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Pretune	Bool	Start pretuning pushbutton
bPB_FineTune	Bool	Start finetuning pushbutton
bPB_AutomaticMode	Bool	Switch to automatic mode pushbutton
bPB_ManualMode	Bool	Switch to manual mode pushbutton
bPB_ManualTuning	Bool	Switch to manual PID tuning pushbutton
bTOG_ManualTuning	Bool	TOG Manual tuning enabled
bPBEN_ResetError	Bool	Reset errors pushbutton enabled
bPBEN_Pretune	Bool	Pretuning pushbutton enabled
bPBEN_FineTune	Bool	Fine tuning pushbutton enabled
bPBEN_AutomaticMode	Bool	Automatic mode pushbutton enabled
bPBEN_ManualMode	Bool	Manual mode pushbutton enabled
bError	Bool	Overall error

9.5.3.2. udtError_PID

Name	Description
InputOutOfRange	Input value is out of the configured range

InputPERInvalid	InputPER value is invalid
ValueOscillationFailed	Fine tuning - process value oscillation could not be maintained
ProcessValueCloseToSetPoint	Pre-tuning - process value is too close to set point
SetPointChangedDuringTuning	PID set point was changed during tuning
PretuningDuringFineTuning	Pre-tuning not allowed while fine tuning is active
InvalidOutputValueLimits	Pre-tuning - invalid configuration of output value limits
InvalidFineTuningParameter	Fine tuning - error occurred causing invalid parameters
InputInvalidFormat	Input value has an invalid number format
OutputCalculationError	Output value calculation error occurred
SamplingTimeError	PID_Compact not called within sampling time of cyclic interrupt OB
SetPointInvalidFormat	Set point value has an invalid number format
ManualInvalidFormat	Manual value has an invalid number format
SubstituteOutputInvalidFormat	Substitute output value has an invalid number format
DisturbanceInvalidFormat	Disturbance value has an invalid number format

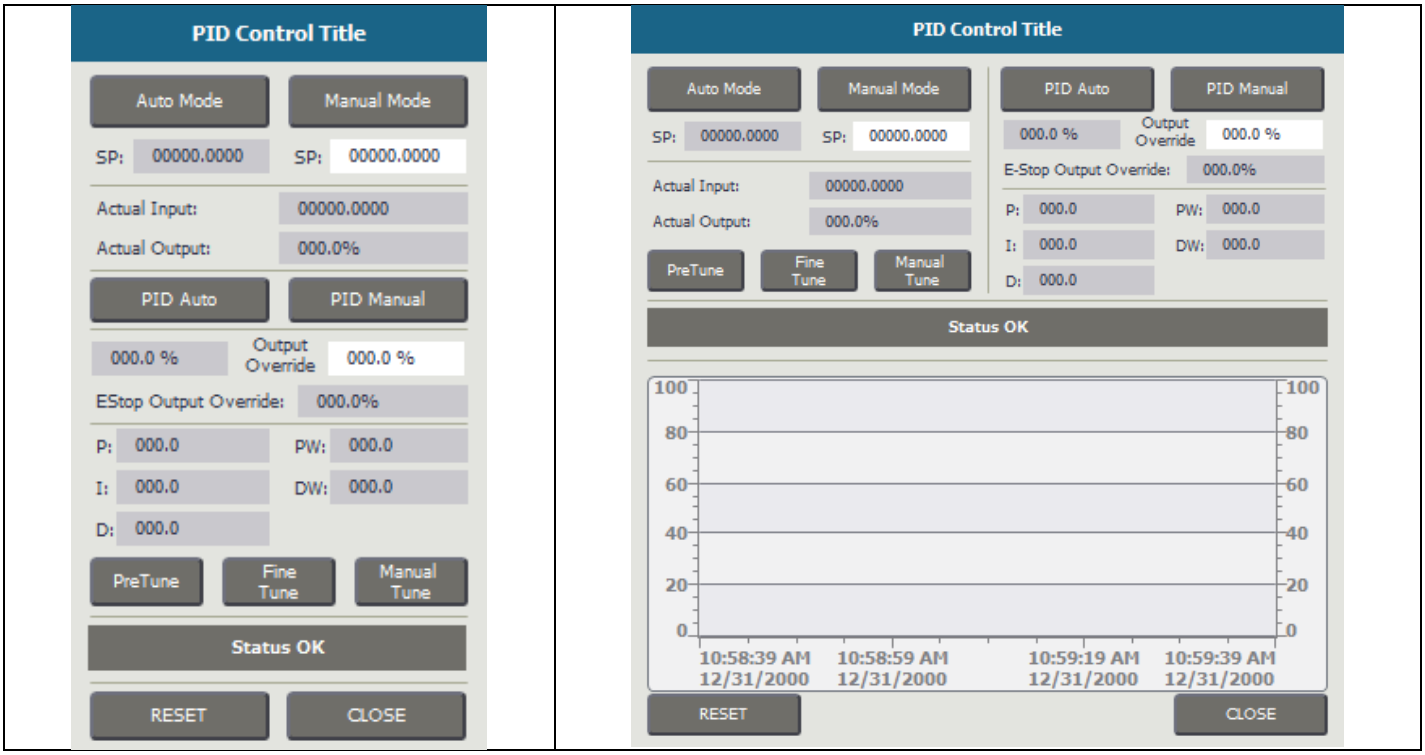
9.5.4. HMI Pop-up Faceplate

The HMI pop up provides a detailed panel for PID control. There are two pop-up faceplates available: fpPID_Compact_Popup and fpPID_Compact_Popup_Graph.

9.5.4.1. HMI Pop-up display

The library contains two pop-up faceplates, shown below:

PID Control Pop-up	PID Control Pop-up w/Trend Graph



9.5.4.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplates:

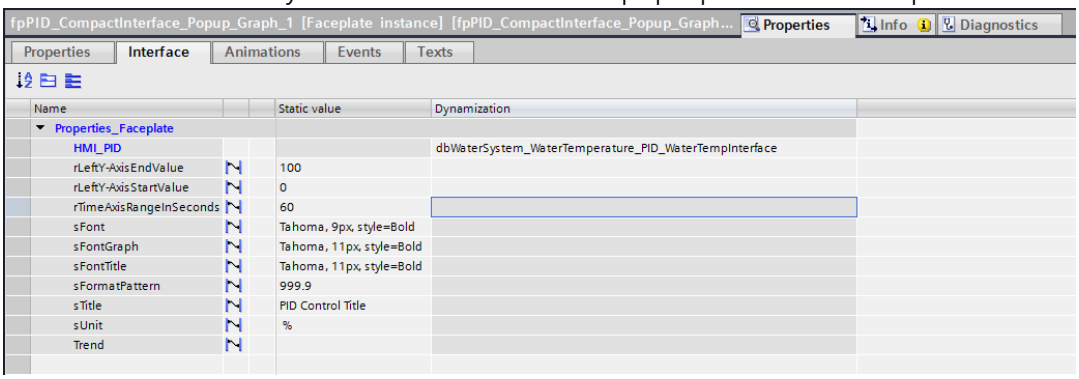
Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Auto Mode SP	Output	Displays PLC set point to be used in Auto Mode.
Manual Mode SP	Input	Adjusts the PLC set point when in Manual Mode
Actual Input	I/O Field	Displays the input to the PID_Compact block.
Actual Output	I/O Field	Displays the output of the PID_Compact block.
PID Auto Button	Button	Sets PID control to Auto
PID Manual Button	Button	Sets PID control to Manual
Auto Output Override	Output	Displays PLC value for auto mode PID override value
Manual Output Override	Input	Adjusts the override output value when in manual mode
EStop Output Override	I/O Field	Adjusts the PID override output when an E-Stop occurs. iEStopFunction must be set to 'HMISetPoint' to enable this input
P Field	I/O Field	Proportional gain value
I Field	I/O Field	Integral gain value
D Field	I/O Field	Derivative gain value

PW Field	I/O Field	Proportional Weight
DW Field	I/O Field	Derivative Weight
PreTune Button	Button	Initializes a pre-tune in the PID_Compact Block
Fine Tune Button	Button	Initializes a fine tune in the PID_Compact Block
Manual Tune Button	Button	Initializes a manual tune in the PID_Compact Block
RESET Button	Button	Resets the device
CLOSE Button	Button	Exits the pop up screen

9.5.4.3. HMI Pop-up Interface

The device interface is shown below. The following properties are linked to the interface:

- HMI_PID: Needs to be mapped to the same 'udtHMI_PID' used by the Function Block
- rLeftY-AxisEndValue: Maximum value for the left Y-Axis of the Trend Graph. This will typically be the maximum value of the variable that is being controlled, water temperature for this example. Adjusted directly in table. Note: This is only used for the PID Control pop-up with Trend Graph
- rLeftY-AxisStartValue: Minimum value for the left Y-Axis of the Trend Graph. This will typically be the minimum value of the variable that is being controlled, water temperature for this example. Adjusted directly in table. Note: This is only used for the PID Control pop-up with Trend Graph
- rTimeAxisRangeInSeconds: Defines the number of seconds being shown on the Trend graph. Adjusted directly in table. Note: This is only used for the PID Control pop-up with Trend Graph
- sFont: Sets the font style used in the popup.
- sFontGraph: Sets the font style used for the graph.
- sFontTitle: Sets the font style used in the title.
- sFormatPattern: Changes data format of values, setting amount of leading and trailing zeros of outputs. Adjusted directly in table
- sTitle: Sets title of the pop-up display. Adjusted directly in table
- sUnit: Sets the units used in the display. Adjusted directly in table
- Trend: This is configured by right clicking on the trend on the pop up and adding the relevant tags to the trend. Note: This is only used for the PID Control pop-up with Trend Graph



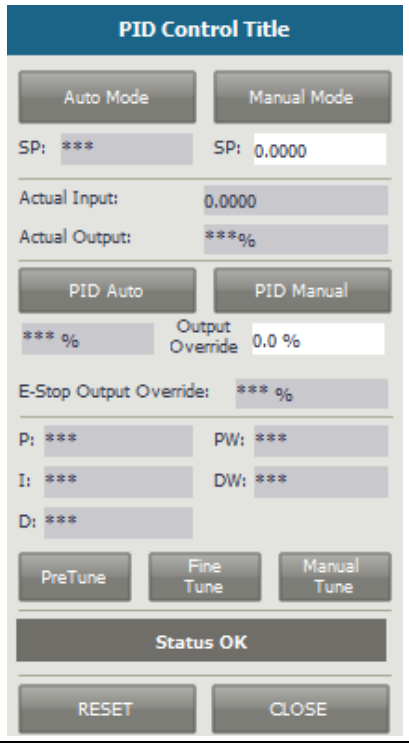
*Note: If custom values are required for both the right and left Y-Axis, use the PID control without the Trend Graph and add a new one in manually. The right Y-Axis for the pop-up with the Trend Graph is set to default to values between 0 and 100, representing the percentage of the PID control.

9.5.5. WinCC Professional Pop-up Screen

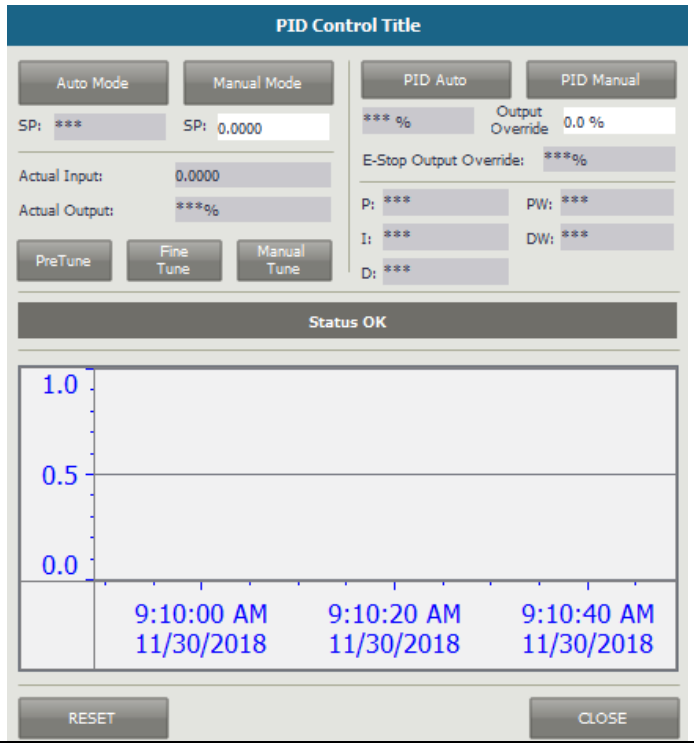
The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

9.5.5.1. Screen

PID Control Pop-up



PID Control Pop-up w/Trend Graph



9.5.5.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object.

2 = PID_Compact_Popup_Pro or PID_Compact_Popup_Graph_Pro

3 = The desired title for the pop-up. E.g. "Water Temperature PID"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

9.5.6. SiVArc Implementation.

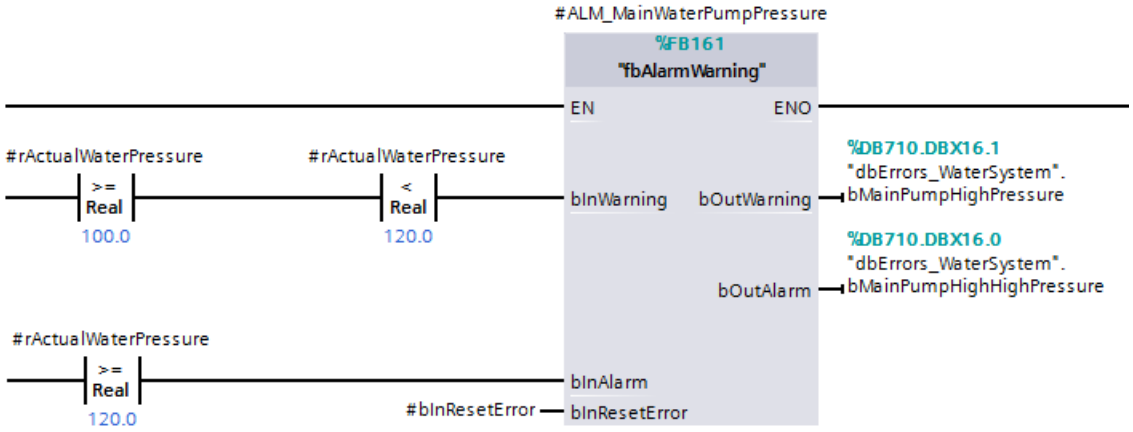
Two screen rules are defined for the object. Both create pop-ups, depending on conditions. If either NetworkTitle[1] isn't defined, or is defined to anything besides "TREND", fpPID_Compact_Popup (no graph) will be used. If it is defined to "TREND" fp_PID_Compact_Popup_Graph will be used.

There is no WinCC Professional SiVArc implementation for the PID objects.

9.6. Standard Alarm Interface – fbAlarmWarning

9.6.1. Description

The Standard Alarm Interface provides an easy way to setup additional alarms. Alarms will latch until reset.



9.6.2. Function Block Interface

9.6.2.1. Input Parameters

Input Variables	Type	Description
bInWarning	Bool	Warning Signal. Provides warning when signal is true
bInAlarm	Bool	Alarm Signal. Provides alarm when signal is true
bInResetError	Bool	If an error condition exists, the error must be resolved first, then set this bit high to reset the internal error

9.6.2.2. Out Parameters

Out Variables	Type	Description
bOutWarning	Bool	Signal used to show warning is active
bOutAlarm	Bool	Signal used to show alarm is active and has not been reset

9.6.3. HMI Icon Display

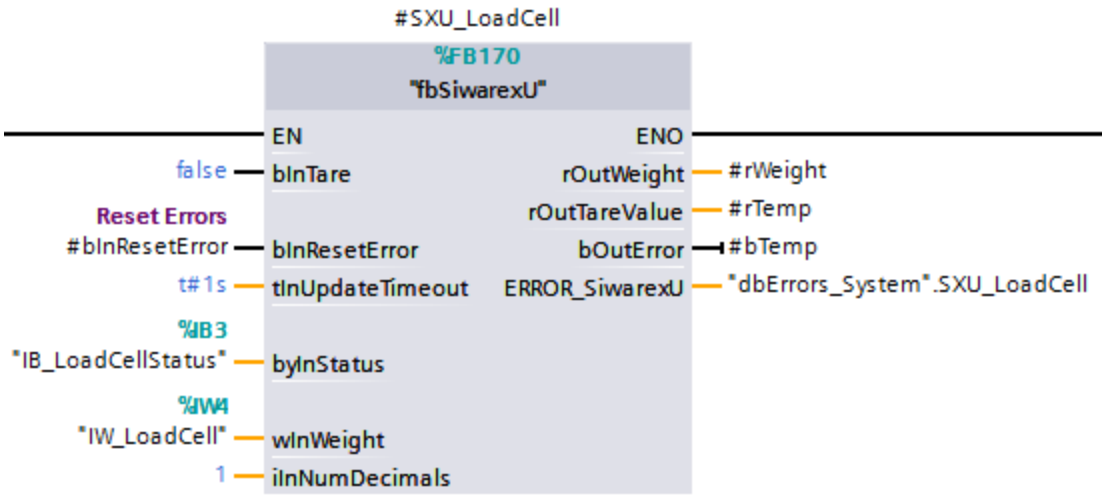
The HMI interface is done through the Alarm banner. There is no faceplate for this object. For information on how to utilize alarms, see 'Siemens Open Library – Siemens Alarm Generation' document.

10. Load Cell Modules

10.1. SiwarexU – fbSiwarexU

10.1.1. Description

This block provides a basic weighing interface when using a Siemens SiwarexU load cell module.



10.1.2. Function Block Interface

10.1.2.1. Input Parameters

Input Variables	Type	Description
bInTare	Bool	Tares the current load cell reading
tInUpdateTimeout	Time	Defined update time of the load cell reading
byInStatus	Byte	Status byte from the Siwarex module
wInWeight	Word	Weight input from the Siwarex module
iInNumDecimals	Int	Number of decimals configured on the module
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error

10.1.2.2. Out Parameters

Output Variables	Type	Description
rOutWeight	Real	Scaled weight reading
bOutError	Bool	Error exists
rOutTareValue	Real	Current amount between module zero and current reading
ERROR_SiwarexU	udtError_SiwarexU	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

10.1.3. User Defined Types

10.1.3.1. udtError_SiwarexU

Name	Data Type	Description
GroupError	Bool	Group error, check module
WriteAccessFailed	Bool	Module write access denied, check configuration
CalibrationNeeded	Bool	Module has not been calibrated or has invalid calibration parameters
UpdateTimeout	Bool	Module failed to update within specified time

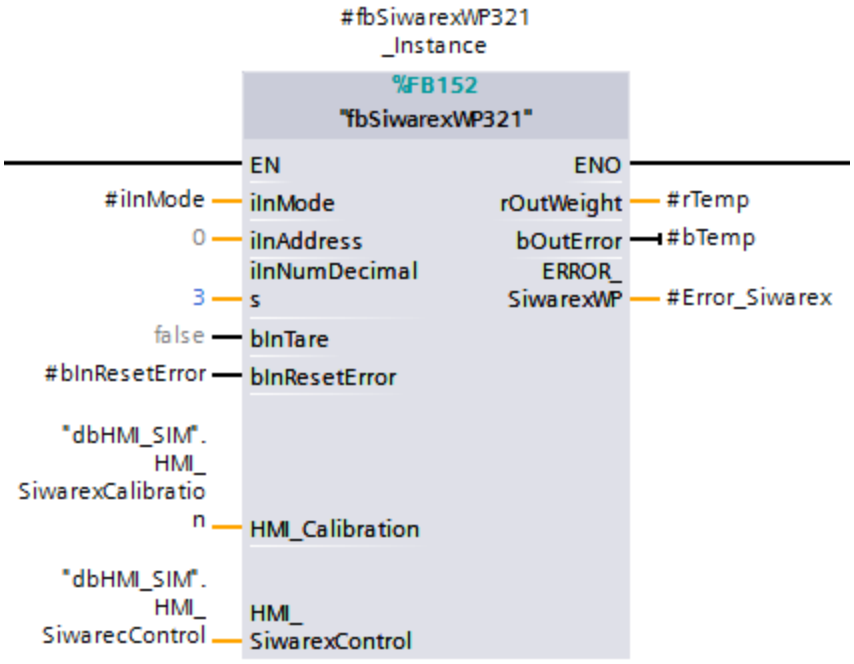
10.1.4. HMI Display

The SiwarexU block does not currently have a provided HMI implementation.

10.2. SiwarexWP321 Weighing System – fbSiwarexWP321

10.2.1. Description

The SiwarexWP321 Function Block is used to interface with Siemens' Siwarex WP321 electronic weighing systems. It provides a more easily read and interfaced between the fbSiwarexWP321Limited_SIMATIC function block and HMIs.



10.2.2. Function Block Interface

10.2.2.1. Input Parameters

Input Variables	Type	Description
iInMode	Int	Mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
iInAddress	Int	Starting logical address for the SiwarexWP321.
iInNumDecimal	Int	Number of decimal places to return.
bInTare	Bool	Tare push button.
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error
HMI_Calibration	udtHMI_SiwarexCalibration	This UDT is used to communicate to the HMI. Inside the UDT are variables needed to zero, calibrate and configure and standing still range and time. Also inside this UDT are all feedback signals including errors and current weight.

*Only valid in Automatic Mode

10.2.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_SiwarexControl	udtHMI_SiwarexControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to set device mode, manual override tare and address the device.

10.2.2.3. Out Parameters

Output Variables	Type	Description

rOutWeight	Real	Current system weight either the measured weight, in auto mode, or the input weight from the HMI in manual mode.
bOutError	Bool	Error Exists
ERROR_SiwarexWP	udtError_SiwarexWP	This output UDT has one bit for every type of error. It should be uses for debugging errors and displaying error messages.

10.2.3. User Defined Types

10.2.3.1. udtHMI_SiwarexCalibration

Name	Type	Description
bPB_ZeroLoadCell	Bool	Push button to zero the load cell
bPBEN_ZeroLoadCell	Bool	Enable scale zeroing
bPB_CalibrationWeight0	Bool	Push button to set current weight as calibration weight 0
bPBEN_CalibrationWeight0	Bool	Enables set calibration weight 0
bPB_CalibrationWeight1	Bool	Push button to set current weight as calibration weight 1
bPBEN_CalibrationWeight1	Bool	Enables set calibration weight 1
bPB_CalibrationWeight2	Bool	Push button to set current weight as calibration weight 2
bPBEN_CalibrationWeight2	Bool	Enables set calibration weight 2
bPB_SetValues	Bool	Push button to set calibration values
bPBEN_SetValues	Bool	Enables setting of calibration values
bPB_EnableCalibration	Bool	Toggles the enabling of calibration
bTOG_EnableCalibration	Bool	Status of calibration enable
bPB_ResetCalibration	Bool	PB to reset calibration routine
bPB_ReserError	Bool	Push button to reset error
b3WeightCalibrationMode	Bool	Calibrate using 3 weights instead of 2
bError	Bool	Error exists
iModuleAddress	Int	IO address of module to calibrate
iState	Int	State of the module calibration
iErrorCode	Int	HMI error code
iNumberOfDecimals	Int	Number of decimals to display weight
rCalibrationWeight0	Real	Value of calibration weight 0
rCalibrationWeight1	Real	Value of calibration weight 1
rCalibrationWeight2	Real	Value of calibration weight 2
rMaximumWeight	Real	Maximum measurable weight
rActualWeight	Real	Current Weight measured by load cell

rStandstillRange	Real	Weight range of standstill
rStandstillTime	Real	Time needed to achieve standstill (s)

10.2.3.2. udtHMI_SiwarexControl

Name	Type	Description
iMode	Int	Current device mode
iErrorCode	Int	Error code
iStatus	Int	Status for the HMI
iModuleAddress	Int	IO Start address of the module
bPB_TarLoadCell	Bool	Push button to tear the load cell
bPBEN_TarLoadCell	Bool	Enable taring of the load cell
bPB_DeleteTareValue	Bool	Push button to delete current tare
bPB_ReserError	Bool	Push button to reset error
bError	Bool	Error exists
rActualWeight	Real	Current weight measured by the load cell
rManualWeight	Real	Manual override weight for HMI

10.2.3.3. udtError_SiwarexWP

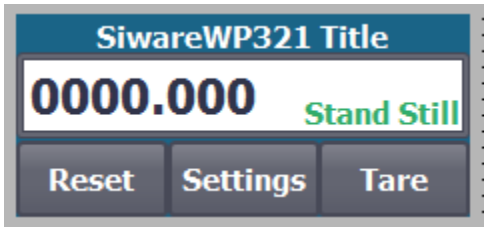
Name	Type	Description
CommandError	Bool	A command finished with an error
AddressError	Bool	Incorrect hardware address
RecordTransmissionFailed	Bool	Runtime error during transmission of data record
Read Error	Bool	Reading of I/O data failed
CommunicationLost	Bool	SiwarexWP module is not responding
CommandSentAtStartup	Bool	Command was sent during startup
DataRecordMismatch	Bool	Data record version dose not match firmware
InvalidCommand	Bool	An invalid command was sent to the module
DataOperationError	Bool	Synchronous data operations error has occurred
InternalFault	Bool	Module failed – replace the module
LoadingVoltageMissing	Bool	Low voltage – check supply voltage on module
HardwareInterruptLost	Bool	Incorrect interrupt processing
ModuleNotAvailable	Bool	Module is not ready for normal operations
Undervoltage	Bool	Voltage drop to load cells is to large
Overload	Bool	Scale overloaded

Underload	Bool	Scale underloaded – check mechanical system
ChecksumErrorInParameter	Bool	Incorrect parameters – reset to factory defaults
ChecksumErrorInProgram	Bool	Incorrect module program – reset firmware or replace module

10.2.4. HMI Icon Display

10.2.4.1. HMI Icon Display Object

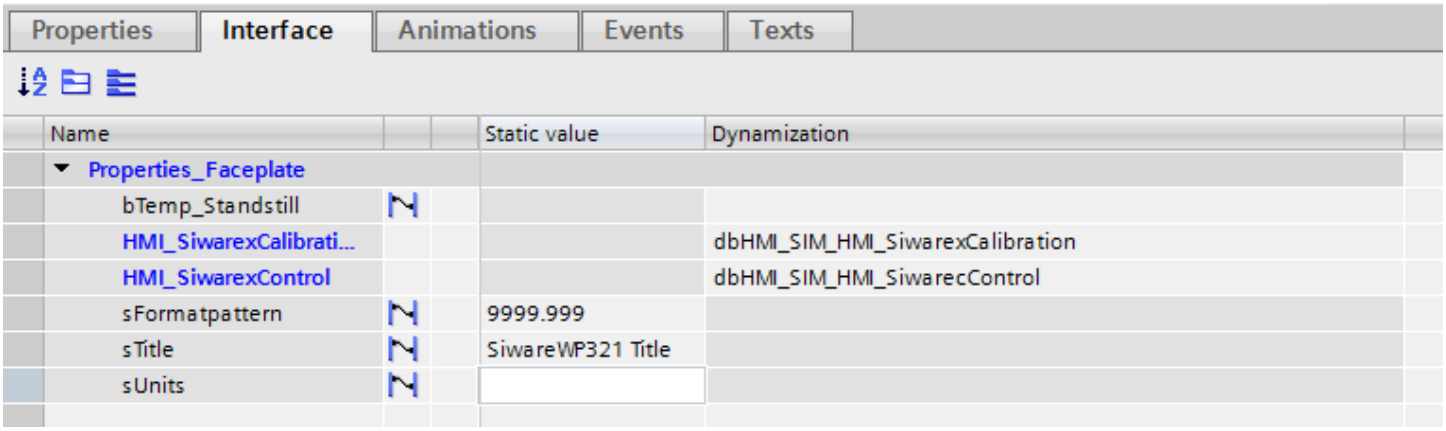
The library contains the following display to be used on the HMI screen:



10.2.4.2. HMI Icon Interface

The device interface is shown below. The following six properties need to be assigned to the object:

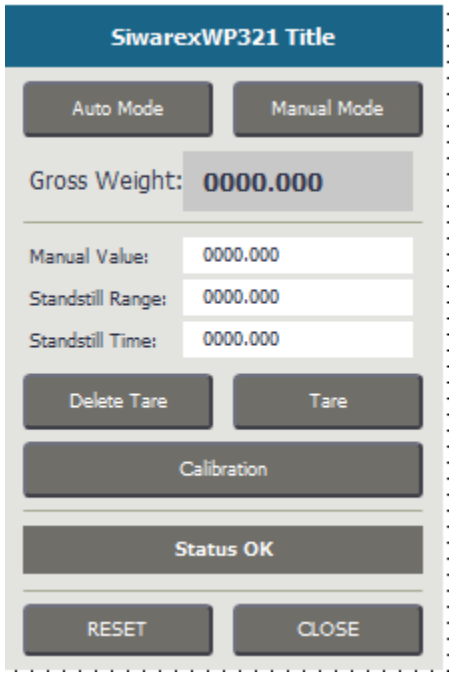
- bTemp_Standstill: needs to be mapped to the static variable bAtStandstill uses by the function block.
- HMI_SiwarexCalibration: Calibration needs to be mapped to the same 'udtHMI_SiwarexCalibration
- used by the Function Blocks
- HMI_SiwarexControl: Control needs to be mapped to the same 'udtHMI_SiwarexCcontrol
- used by the Function Blocks
- sFormatpattern: Sets the amount of leading/trailing zeros of the display value
- sTitle: Sets the icons title.
- sUnits: Sets the unit the value is displayed in (i.e. amps, centimeters etc.)



10.2.5. HMI Control Pop-up Faceplate

The HMI pop up provides a detailed display of the status of the SiwarexWP321. Additionally, it allows for control when the System Mode is Independent or Manual modes.

10.2.5.1. HMI Pop-up display



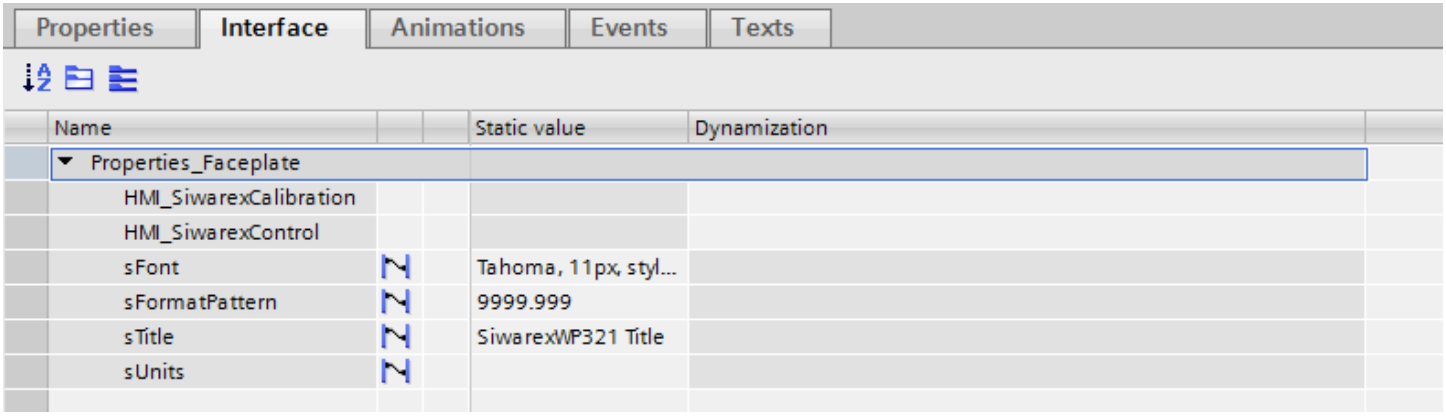
10.2.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Error Status Field	I/O Field	Displays current error status. Field box turns red if any errors are present, and displays active error text. It will scroll through multiple alarms.
Manual Value	I/O Field	Weight to output when in manual mode
Standstill Range	I/O Field	Maximum amount the weight can drift up or down in the standstill time and still return that the scale has reached a standstill.
Standstill Time	I/O Field	Time for which the weight must remain in the range set by standstill range to return that the scale has reached standstill
Delete tare	Button	Deletes the existing tare value used by the scale
Tare	Button	Sets the scales current weight as the tare value, effectively shifting the reading to zero.
Calibration	Button	Opens the calibration popup screen, used to calibrate the SiwarexWP321 module.
RESET Button	Button	Resets errors
CLOSE Button	Button	Exits the pop up screen

10.2.5.3. HMI Pop-up Interface

The device interface is shown below. Six properties are linked in this interface: HMI_SiwarexCalibration, HMI_SiwarexControl, sFont, sFormatPattern, sTitle, and sUnits. The HMI_ SiwarexCalibration needs to be mapped to the same 'udtHMI_ SiwarexCalibration' used by the Function Block. The HMI_ SiwarexControl needs to be mapped to the same 'udtHMI_ SiwarexControl used by the Function Block. The sFont property controls the font name used by text elements. The sFormatPattern should be set to display the desired number of decimal places and preceding zeros. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. sUnits should be set to display the correct units that the scale is reading out in.



10.2.6. HMI Calibration Pop-up Faceplate

The HMI calibration pop up provides a detailed display for calibrating the SiwarexWP321. Additionally, it allows for calibration modes and configuration changes.

10.2.6.1. HMI Pop-up display

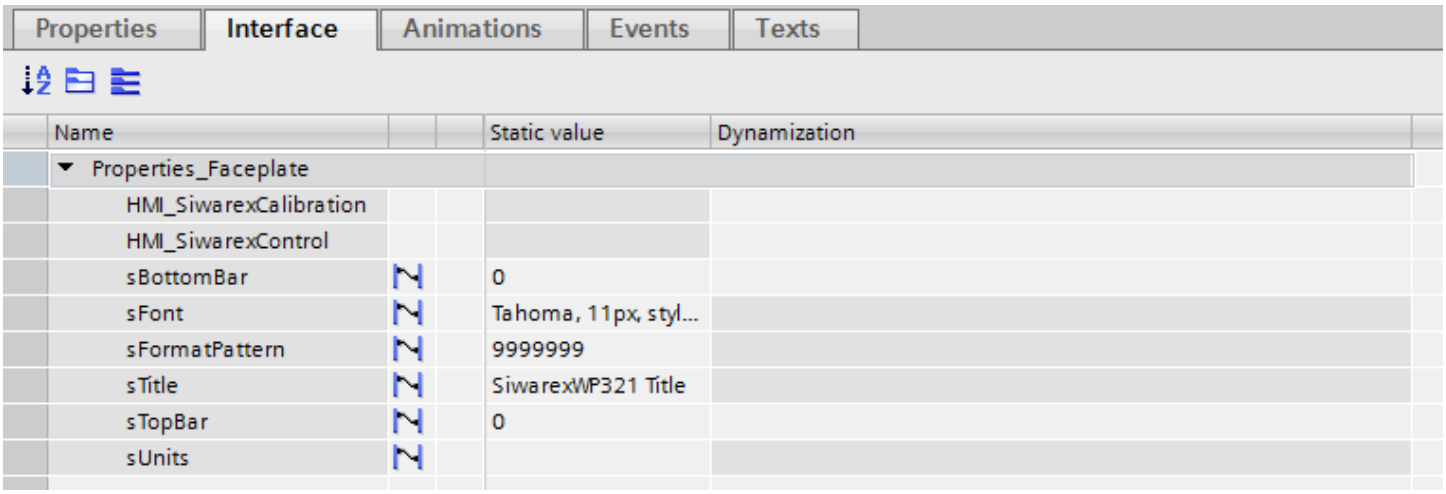
10.2.6.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Top bar	I/O Field	Input IO field at the top of the popup screen
Enable Calibration	Button	Enables calibration of the SiwarexWP321
Module Address	I/O Field	Siwarexwp321 module address
Start Calibration	Button	Starts a calibration cycle
Disable 3 Weight Calibration	Button	Disables and reenables three weight calibration mode
Gross Weight	I/O Field	Gross weight read by the unit
Zero Load	Button	Zeros the load cell (Not the same as tare)
Bottom bar	I/O Field	Output IO field for the calibration/system status
Calib. Weight 0	I/O Field	Actual mass of calibration weight zero
Calib. Weight 1	I/O Field	Actual mass of calibration weight one
Calib. Weight 2	I/O Field	Actual mass of calibration weight two
Max Weight	I/O Field	Maximum allowable weight for the unit
Standstill Range	I/O Field	How much can a reading drift and still be considered at a standstill
Standstill Time	I/O Field	How long must the reading stay in the stand still range to be considered at a standstill
Accept Values	Button	Accept calibration weights and parameters
Calibrate Weight 0	Button	Sets the SiwarexWP321 to measure calibration weight zero
Calibrate Weight 1	Button	Sets the SiwarexWP321 to measure calibration weight one
Calibrate Weight 2	Button	Sets the SiwarexWP321 to measure calibration weight two
Reset	Button	Resets the unit, clearing any errors
Exit	Button	

10.2.6.3. HMI Pop-up Interface

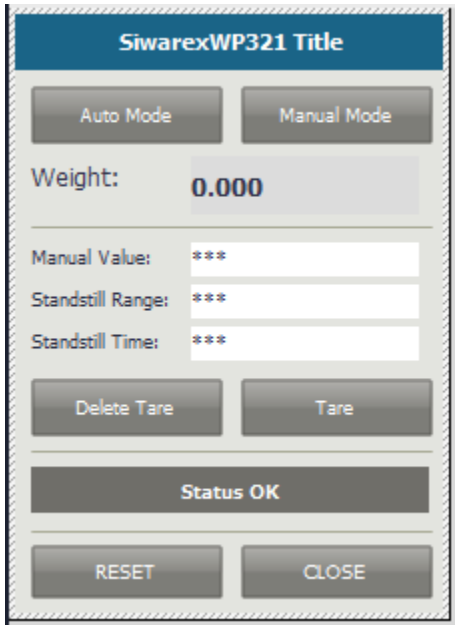
The device interface is shown below. Eight properties are linked in this interface: HMI_SiwarexCalibration, HMI_SiwarexControl, sFont, sFormatPattern, sTitle, and sUnits. The HMI_ SiwarexCalibration needs to be mapped to the same 'udtHMI_ SiwarexCalibration' used by the Function Block. The HMI_ SiwarexControl needs to be mapped to the same 'udtHMI_ SiwarexControl used by the Function Block. The sFont property controls the font name used by text elements. The sFormatPattern should be set to display the desired number of decimal places and preceding zeros. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. sUnits should be set to display the correct units that the scale is reading out in. sBottomBar should be tied to the calibration status. sTopbat can be tied to any input needed by the SiwarexWP321 for calibration.



10.2.7. WinCC Professional Pop-up

The pop-up screen looks and functions like the faceplate described above. However, it must be loaded into a screen window, and a visual basic script must be used to launch it from the corresponding icon. See "3-Example Object Configuration" for step-by-step instructions.

10.2.7.1. Screen



10.2.7.2. Script

To launch the screen from the corresponding icon, use the below script in a click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_MTR_ReturnWaterPump"

2 = Motor_Reversing_Popup_Pro

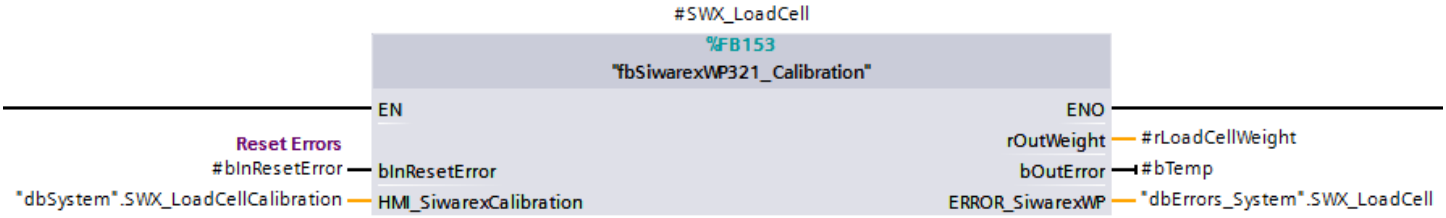
3 = The desired title for the pop-up. E.g. "Return Water Pump"

```
Sub OnClick(ByVal item)

SetPropertyByConstant  AccessPath, "Screen window_1", "TagPrefix", "**1**"

SetPropertyByConstant  AccessPath, "Screen window_1", "Visible", "True"

ActivateScreenInScreenWindow  AccessPath, "Screen window_1", "**2**"

HMIRuntime.Screens(AccessPath  & ".Screen window_1:**2**").ScreenItems("Title").Text  = "**3**"

End Sub
```

10.3. SiwarexWP 321 Calibration – fbSiwarexWP321_Calibration

10.3.1. Description

This block works in coordination with the fbSiwarexWP321 block and adds calibration routine functionality on top of standard weighing functions.



10.3.2. Function Block Interface

10.3.2.1. Input Parameters

Input Variables	Type	Description
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error

10.3.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_Calibration	udtHMI_SiwarexCalibration	This UDT is used to communicate to the HMI. Inside the UDT are variables needed to zero, calibrate and configure and standing still range and time. Also inside this UDT are all feedback signals including errors and current weight.

10.3.2.3. Out Parameters

Output Variables	Type	Description

rOutWeight	Real	Current system weight either the measured weight, in auto mode, or the input weight from the HMI in manual mode.
bOutError	Bool	Error Exists
ERROR_SiwarexWP	udtError_SiwarexWP	This output UDT has one bit for every type of error. It should be uses for debugging errors and displaying error messages.

10.3.3. User Defined Types

10.3.3.1. udtHMI_SiwarexCalibration

Name	Type	Description
bPB_ZeroLoadCell	Bool	Push button to zero the load cell
bPBEN_ZeroLoadCell	Bool	Enable scale zeroing
bPB_CalibrationWeight0	Bool	Push button to set current weight as calibration weight 0
bPBEN_CalibrationWeight0	Bool	Enables set calibration weight 0
bPB_CalibrationWeight1	Bool	Push button to set current weight as calibration weight 1
bPBEN_CalibrationWeight1	Bool	Enables set calibration weight 1
bPB_CalibrationWeight2	Bool	Push button to set current weight as calibration weight 2
bPBEN_CalibrationWeight2	Bool	Enables set calibration weight 2
bPB_SetValues	Bool	Push button to set calibration values
bPBEN_SetValues	Bool	Enables setting of calibration values
bPB_EnableCalibration	Bool	Toggles the enabling of calibration
bTOG_EnableCalibration	Bool	Status of calibration enable
bPB_ResetCalibration	Bool	PB to reset calibration routine
bPB_ReserError	Bool	Push button to reset error
b3WeightCalibrationMode	Bool	Calibrate using 3 weights instead of 2
bError	Bool	Error exists
iModuleAddress	Int	IO address of module to calibrate
iState	Int	State of the module calibration
iErrorCode	Int	HMI error code
iNumberOfDecimals	Int	Number of decimals to display weight
rCalibrationWeight0	Real	Value of calibration weight 0
rCalibrationWeight1	Real	Value of calibration weight 1
rCalibrationWeight2	Real	Value of calibration weight 2
rMaximumWeight	Real	Maximum measurable weight
rActualWeight	Real	Current Weight measured by load cell

rStandstillRange	Real	Weight range of standstill
rStandstillTime	Real	Time needed to achieve standstill (s)

10.3.3.2. udtError_SiwarexWP

Name	Type	Description
CommandError	Bool	A command finished with an error
AddressError	Bool	Incorrect hardware address
RecordTransmissionFailed	Bool	Runtime error during transmission of data record
Read Error	Bool	Reading of I/O data failed
CommunicationLost	Bool	SiwarexWP module is not responding
CommandSentAtStartup	Bool	Command was sent during startup
DataRecordMismatch	Bool	Data record version dose not match firmware
InvalidCommand	Bool	An invalid command was sent to the module
DataOperationError	Bool	Synchronous data operations error has occurred
InternalFault	Bool	Module failed – replace the module
LoadingVoltageMissing	Bool	Low voltage – check supply voltage on module
HardwareInterruptLost	Bool	Incorrect interrupt processing
ModuleNotAvailable	Bool	Module is not ready for normal operations
Undervoltage	Bool	Voltage drop to load cells is to large
Overload	Bool	Scale overloaded
Underload	Bool	Scale underloaded – check mechanical system
ChecksumErrorInParameter	Bool	Incorrect parameters – reset to factory defaults
ChecksumErrorInProgram	Bool	Incorrect module program – reset firmware or replace module

10.3.4. HMI Display

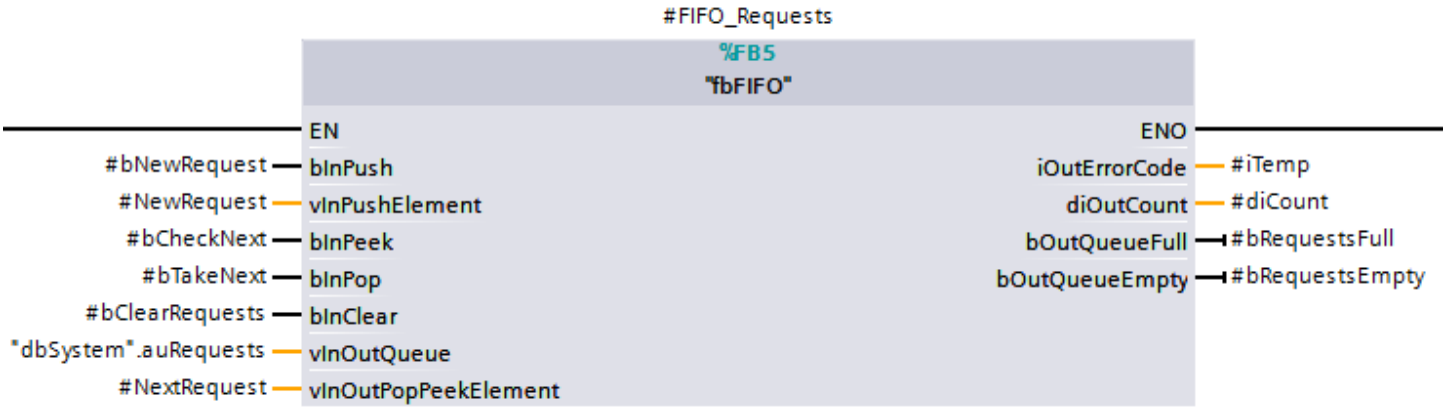
The SiwarexWP 321 Calibration block does not currently have a provided HMI implementation.

11. Process Control

11.1. First-In-First-Out (FIFO) Queue – fbFIFO

11.1.1. Description

This block implements a standard FIFO queue utilizing variant data types to work dynamically with any array structure. It includes standard queuing commands such as push, pop, and peek.



11.1.2. Function Block Interface

NOTE: The types of 'vInPushElement' and 'vInOutPopPeekElement' must be identical. The type of 'vInOutQueue' must be a single-dimensional array of the type of 'vInPushElement'. Any type, including UDTs, may be used.

11.1.2.1. Input Parameters

Input Variables	Type	Description
bInPush	Bool	Pushes a new item (vInPushElement) to the queue on rising edge
vInPushElement	Variant	The new element to push to the queue
bInPeek	Bool	Places the next item in the queue into vInOutPopPeekElement without removing it from the queue.
bInPop	Bool	Removes the next item in the queue and places it in vInOutPopPeekElement
bInClear	Bool	Resets the queue, removing all items

11.1.2.2. InOut Parameters

In/Out Variables	Type	Description
vInOutQueue	Variant	The single-dimensional array used as the queue
vInOutPopPeekElement	Variant	When and item is popped or peeked, it is placed here

11.1.2.3. Out Parameters

Output Variables	Type	Description

iOutErrorCode	Int	Displays the current error of the block, if any
diOutCount	Dint	The number of items in the queue
bOutQueueFull	Bool	Indicates that the queue has filled the entire array
bOutQueueEmpty	Bool	Indicates there are no items in the queue

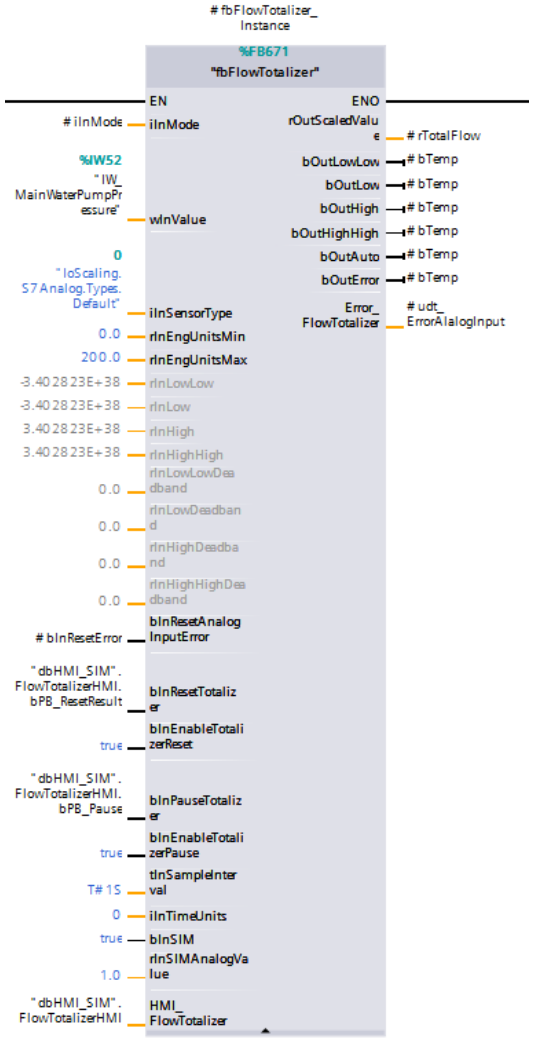
11.1.3. HMI Display

The FIFO block does not currently have a provided HMI implementation.

11.2. Flow Totalizer – fbFlowTotalizer

11.2.1. Description

The Flow Totalizer Function Block integrates the signal from an analog input. The block uses standardized user defined data types that makes it easy to display total flow, warnings, and alarms on an HMI. The Flow Totalizer Function Block can be used to integrate any analog signal, but was specifically designed for uses with fluid, mass and volume based, flow meters.



11.2.2. Function Block Interface

11.2.2.1. Input Paramiters

Input Variables	Type	Description
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
wInValue	Word	Raw input signal from hardware input
iInSensorType	Int	Sensor Type determines how the input should be scaled based on the type of sensor.
rInEngUnitsMin	Real	Minimum value for engineering units
rInEngUnitsMax	Real	Maximum value for engineering units
rInLowLow	Real	Value for Low Alarm to trigger. Use IoScaling.S7Analog.iRawMin to disable
rInLow	Real	Value for Low Warning to trigger. Use IoScaling.S7Analog.iRawMin to disable
rInHigh	Real	Value for High Warning to trigger. Use IoScaling.S7Analog.iRawMax to disable
rInHighHigh	Real	Value for High Alarm to trigger. Use IoScaling.S7Analog.iRawMax to disable
rInLowLowDeadband	Real	Deadband for Low Low Alarm. Use 0 to disable
rInLowDeadband	Real	Deadband for Low Warning. Use 0 to disable
rInHighDeadband	Real	Deadband for High Warning. Use 0 to disable
rInHighHighDeadband	Real	Deadband for High High Alarm. Use 0 to disable
bInResetAnalogInputError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInResetTotalizer	Bool	Reset Totalized Value. Setting this high will reset the totalized value to zero.
bInEnableTotalizerReset	Bool	Enable Totalizer Reset will enable the totalizer to be reset if the condition is met. Any specific conditions that need to be met before resetting the totalizer should be attached here.
bInPauseTotalizer	Bool	Pause Totalizer will pause the totalizer preventing the current flow, or analog signal from being added to the totalized value.
bInEnableTotalizerPause	Bool	Enable Totalizer Pause will enable the totalizer to be paused.
tInSampleInterval	Time	Sample Interval is the discreet time interval over which the analog input will be integrated. For optimal accuracy this should be set to the maximum refresh rate of the analog sensor.
iInTimeUnits	Int	Time Units for Sample Interval. Default, 0 = milliseconds, 1 = seconds, 2 = minutes, 3 = hours, 4 = Days.
bInSIM	Bool	Sets simulation mode on/off
rInSIMAnalogValue	Real	Simulation override value

*Only valid in Automatic Mode

11.2.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_FlowTotalizer	udtHMI_FlowTotalizer	Stores information about the flow totalizer, such as totalized flow, mode, current flow and control, for uses with the HMI.

11.2.2.3. Out Parameters

Output Variables	Type	Description
rOutScaledValue	Real	Current totalized flow

bOutLowLow	Bool	Low low alarm is active
bOutLow	Bool	Low warning is active
bOutHigh	Bool	High warning is active
bOutHighHigh	Bool	High high alarm is active
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the motor
ERROR_FlowTotalizer	Udt_Error_AnalogInput	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

11.2.3. User Defined Types

11.2.3.1. udtHMI_FlowTotalizer

Name	Data Type	Description
bPB_ResetResult	Bool	PB reset result
bPBEN_ResetResult	Bool	Enable PB reset result
bPB_Pause	Bool	PB pause result
bPBEN_Pause	Bool	Enable PB pause result
rOutScaledValue	Bool	Current Scaled result
HMI_AnalogInput	udtHMI_AnologInput	HMI and settings for the analog sensor input

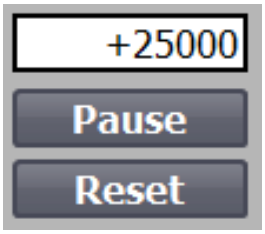
11.2.3.2. udtError_FlowTotalizer

Name	Data Type	Description
LowLowAlarm	Bool	Value below low low set point
HighHighAlarm	Bool	Value above high high set point
Invalid	Bool	Invalid input valid

11.2.4. HMI Icon Display

11.2.4.1. HMI Icon Display Object

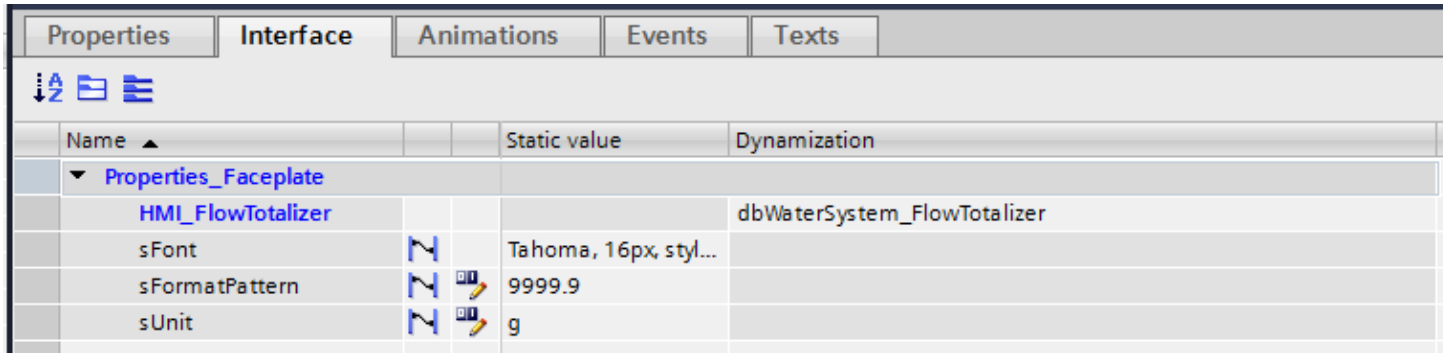
The library contains the following display to be used on the HMI screen:

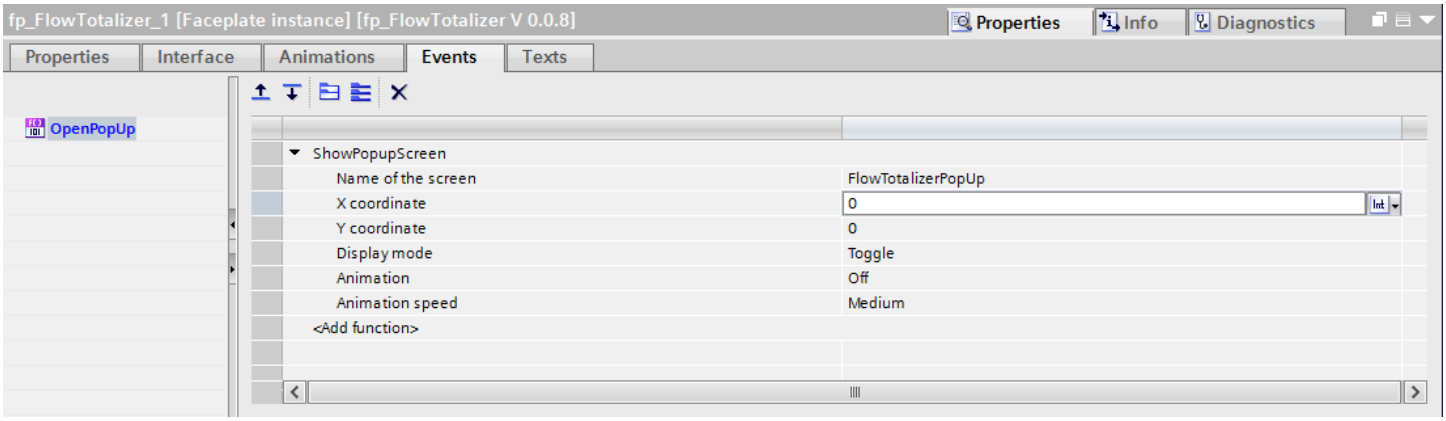


11.2.4.2. HMI Icon Interface

The device interface is shown below. The following four properties need to be assigned to the object:

- HMI_FlowTotalizer: Control needs to be mapped to the same 'udtHMI_FlowTotalizer used by the Function Blocks
- sFont: Determines the font style and size
- sFormatPattern: Sets the amount of leading/trailing zeros of the display value
- sUnit: Sets the unit the value is displayed in (i.e. amps, centimeters etc.)
- OpenPopUp: will open the HMI pop up to control the Flow Totalizer



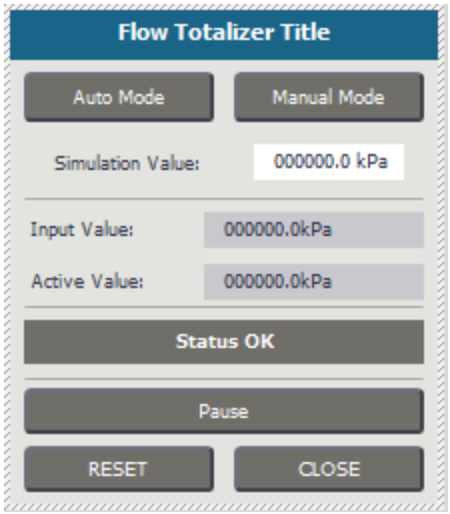


11.2.5. HMI Pop-up Display

The HMI pop up provides a detailed display of the status of the Flow Totalizer device. Additionally, it allows for control of the device when the System Mode is Independent or Manual.

11.2.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



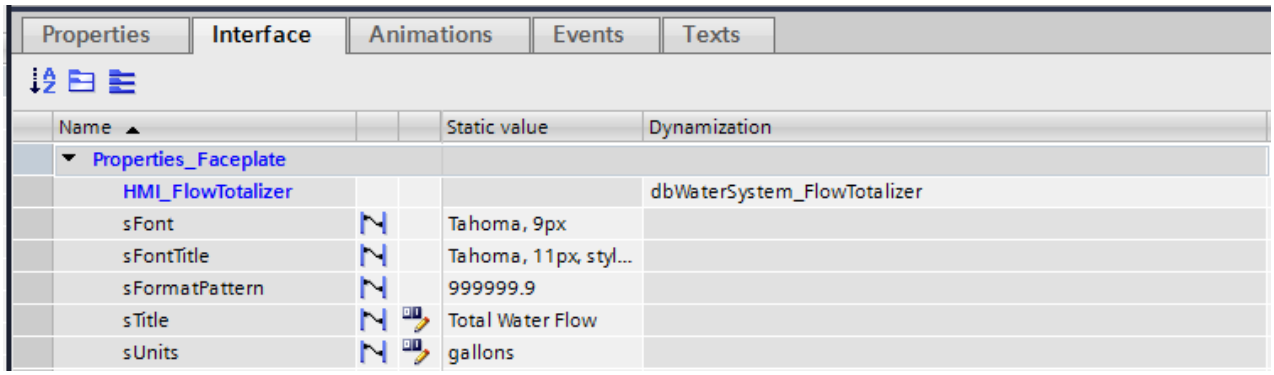
11.2.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Only available when global mode is Independent. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Only available when global mode is Independent. Button is green when device is in Manual Mode.
Simulation Value	I/O Field	Sets the analog input override value which is used if the totalizer is set to manual mode.
Input Value	I/O Field	Current input being read by the analog input.
Active Value	I/O Field	Current totalized flow value.
Status Field	Symbolic I/O Field	Displays current status of device. Field is red for alarm state, yellow for warning state, and grey for normal operation.
Pause	Button	Will Pause the flow totalizers value until pressed again
RESET Button	Button	Resets errors. Note: To reset the totalizer value press the reset button on the flow totalizer icon.
CLOSE Button	Button	Exits the pop up screen

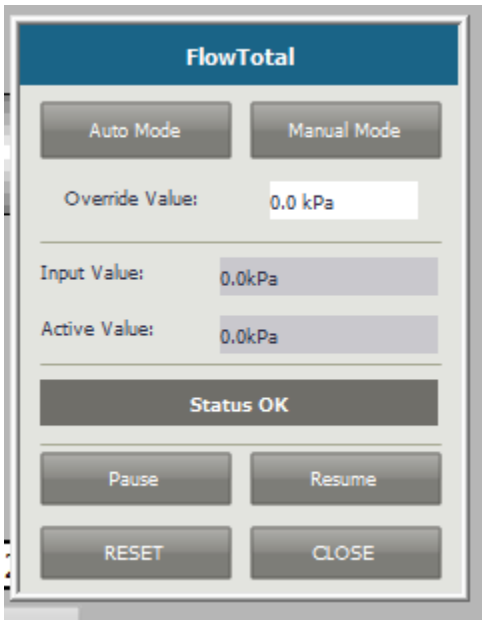
11.2.5.3. HMI Pop-up Interface

The device interface is shown below. Six properties are linked in this interface: HMI_FlowTotalizer, sFont, sFontTitle, sTitle, sFormatPattern, and sUnits. The HMI_FlowTotalizer needs to be mapped to the same 'udtHMI_FlowTotalizer_Control' used by the Function Block. The sFont property controls the font name and size used by text elements. The sFormatPattern property determines how the values are displayed. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sUnits property controls the displayed unit.



11.2.6. WinCC Professional Pop-up Screen

11.2.6.1. PopUp



11.2.6.2. Script

To launch the screen from the corresponding icon, use the below script in a right click event on the icon, with the following replacements:

1 = the HMI tag prefix for the desired instance of the object. For example, this might be "dbWaterSystem_AO_CoolingCoil"

2 = IO_AnalogOutput_Popup_Pro

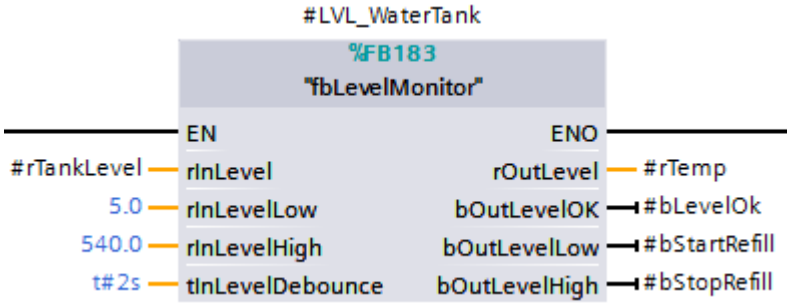
3 = The desired title for the pop-up. E.g. "Cooling Coil"

```
Sub OnPressRight(ByVal item)
SetPropertyByConstant AccessPath, "Screen window_1", "TagPrefix", "**1**"
SetPropertyByConstant AccessPath, "Screen window_1", "Visible", "True"
ActivateScreenInScreenWindow AccessPath, "Screen window_1", "**2**"
HMIRuntime.Screens(AccessPath & ".Screen window_1:**2**").ScreenItems("Title").Text = "**3**"
End Sub
```

11.3. Level Monitoring – fbLevelMonitor

11.3.1. Description

This block monitors the level of a tank or container with an analog level sensor and can alert on preset high and low level set points with a configurable debounce.



11.3.2. Function Block Interface

11.3.2.1. Input Parameters

Input Variables	Type	Description
rInLevel	Real	Input signal form the sensor to be monitored.
bInLevelLow	Real	Value to trigger low level active
bInLevelHigh	Real	Value to trigger high level active
tInLevelDebounce	Time	Debounce time for level value to hit high/low set point

11.3.2.2. Out Parameters

Output Variables	Type	Description
rOutLevel	Real	rInLevel value passthrough
bOutLevelOk	Bool	rInLevel is within the process values and there are no warnings or alarms.

bOutLevelLow	Bool	Indicates the level is at or below the configured low level
bOutLevelHigh	Bool	Indicates the level is at or below the configured high level

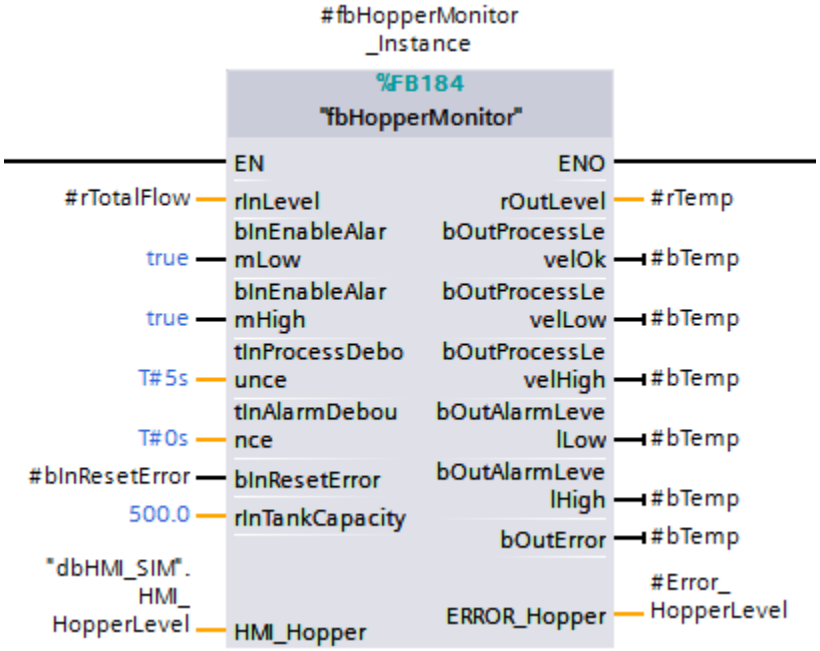
11.3.3. HMI Display

The Level Monitor block does not currently have a provided HMI implementation.

11.4. Advanced Level Monitoring - fbHopperLevel

11.4.1. Description

This block utilized multiple fbLevelMonitor instances to add more level monitoring features. This includes more alarms (low-low, low, high, and high-high) as well as positive feedback that the container level is within the specified allowable range.



11.4.2. Function Block Interface

11.4.2.1. Input Parameters

Input Variables	Type	Description
rInLevel	Real	Input signal form the sensor to be monitored.
bInEnableAlarmLow	Bool	Enables monitoring for the Low Level Alarm.
bInEnableAlarmHigh	Bool	Enables monitoring for the High Level Alarm.

tInProcessDebounce	Time	Debounce time, how long the signal needs to remain in violation of the process values to give a warning.
tInAlarmDebounce	Time	Debounce time, how long the signal needs to remain in violation of the process values to give a alarm.
bInReserError	Bool	If an error condition exist, the error must be fixed first, then set this bit high to reset the internal error.
rInTankCapacity	Real	What is the maximum value of rInLevel.

11.4.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_Hopper	udtHMI_HopperLevel	This In/Out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to configure additional settings and provide status to the HMI

11.4.2.3. Out Parameters

Output Variables	Type	Description
rOutLevel	Real	rInLevel unless the monitor is in manual mode, in which case this is the manual/override value set by the HMI pop-up.
bOutProcessLevelOk	Bool	rInLevel is within the process values and there are no warnings or alarms.
bOutProcessLevelLow	Bool	Warning that rInLevel is below the process low level.
bOutProcessLevelHigh	Bool	Warning that rInLevel is above the process high level.
bOutAlarmLevelLow	Bool	Alarm that rInLevel is below the alarm low level.
bOutAlarmLevelHigh	Bool	Alarm that rInLevel is above the alarm high level.
bOutError	Bool	This output indicates that there's an error condition in the monitor.
ERROR_Hopper	udtError_Hopper	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

11.4.3. User Defined Types

11.4.3.1. udtHMI_HopperLevel

Name	Data Type	Description
Control:		
iMode	Int	Current mode, 1 = Auto, 2 = Manual/Override
rLLAlarm	Real	Low level alarm setpoint.
rHLAlarm	Real	High level alarm setpoint.

rLLProcess	Real	Low level warning setpoint.
rHLProcess	Real	High level warning setpoint.
rTankCapacity	Real	Maximum value of input/tank.
bPB_ResetError	Bool	Reset errors pushbutton.
rManualLevel	Real	Manual override value.
Status:		
bError	Bool	Overall Error.
bLLProcess	Bool	Value below low level warning setpoint.
bHLProcess	Bool	Value above high level warning setpoint.
bLLAlarm	Bool	Value below low level alarm setpoint.
bHLAlarm	Bool	Value above high level alarm setpoint.
rLevel	Real	Current effective level.
iStatus	Int	Status value

11.4.3.2. udtError_Hopper

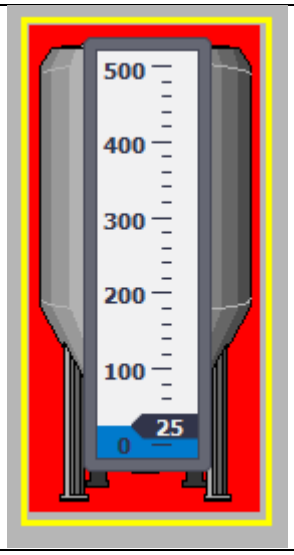
Name	Data Type	Description
LowLevelSignalFailure	Bool	Not used
LowPressureSignalFailure	Bool	Not used
HighPressure	Bool	Not used
LowPressure	Bool	Not used
HighLevel	Bool	Value is above the high level alarm setpoint.
LowLevel	Bool	Value is below the low level alarm setpoint.

11.4.4. HMI Icon Display

11.4.4.1. HMI Icon Display Objects

The library contains the following objects to be used on the HMI screen:

Hopper Level Bar	Hopper Level Numeric
	



11.4.4.2. HMI Icon Appearance and Functionality

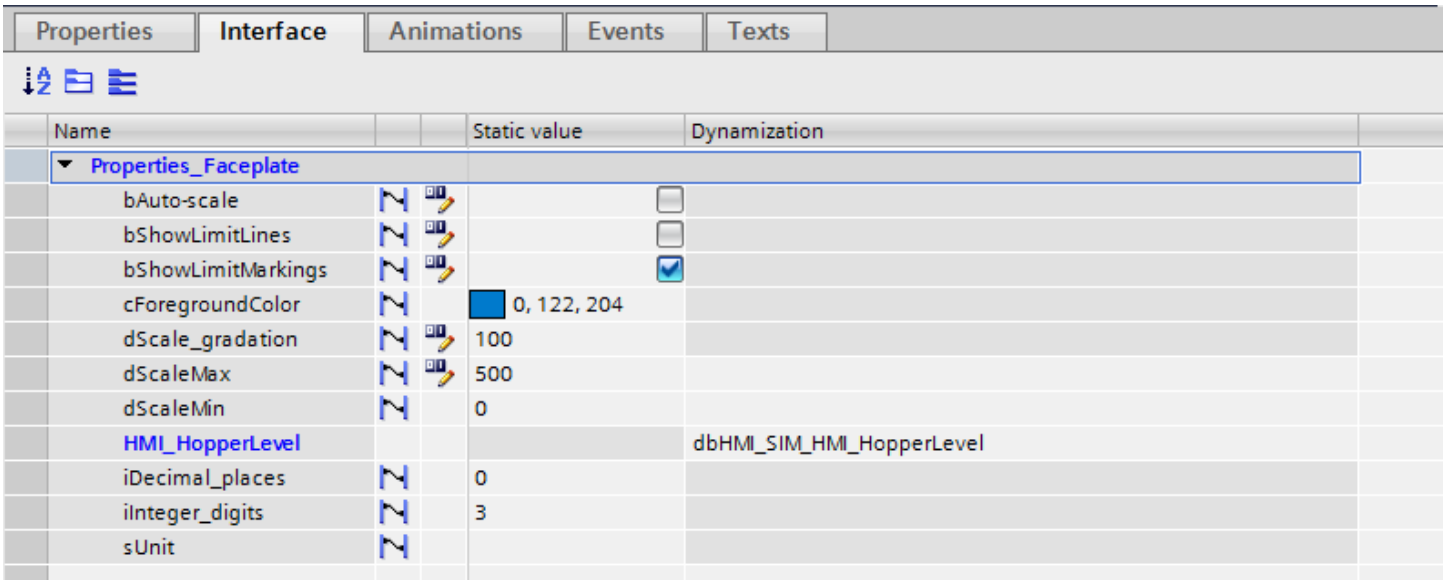
The device appearance's changes depending on the state it is in, see table below for description:

Indicator	Meaning
Background Color	Clear/Grey: Ok, state Yellow: Warning state Red: Alarm state
Yellow Border	Manual override mode

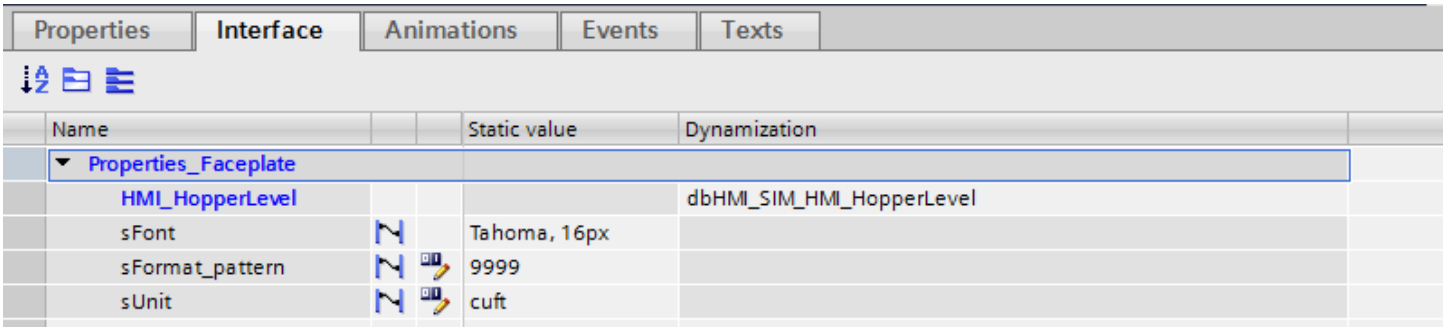
11.4.4.3. HMI Icon Interface

The device interface is shown below. The HMI_HopperLevel needs to be mapped to the same 'udtHMI_HopperLevel' used by the Function Block. Also, units and scale max and min should be set to match physical parameters. bAuto-scale can be set to allow the program to automatically set the scale gradation. bShowLimitLines toggles the visibility of markers that denote high and low limits. cForegroundColor is an rgb input that determines the color of the bar. dScale_gradation allows the user to manually determine the tick mark locations on the bar. dScaleMax and dScaleMin set the maximum and minimum values respectively. iDecimalPlaces determines the decimal accuracy of the scale markers. iIntegerDigits sets the max number of digits that an integer on the scale can have. sUnit sets the units which are displayed following the max and min values. For the numeric icon sFormatPattern accomplishes the work of iIntergerDigits and iDecimalPlaces. Additionally on the numeric icon the user can set sFont to set the font style and size of the displayed data.

Hopper Level Bar:



Hopper Level Numeric:

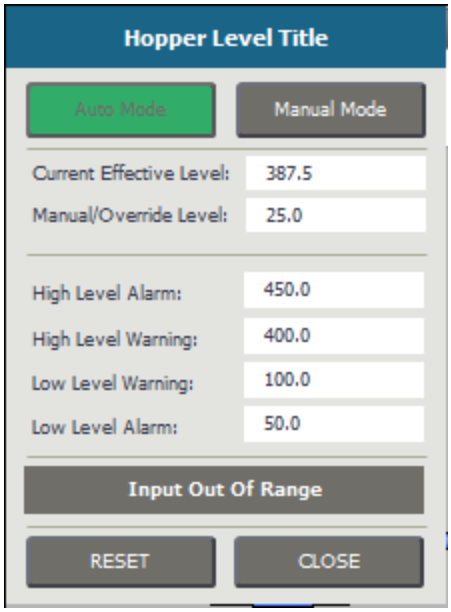


11.4.5. HMI Pop-up Faceplate

The HMI pop up provides a detailed display of the monitor. Additionally, it allows for control of setpoints and System Mode for manual override.

11.4.5.1. HMI Pop-up Display

The library contains a single pop-up faceplate shown below:



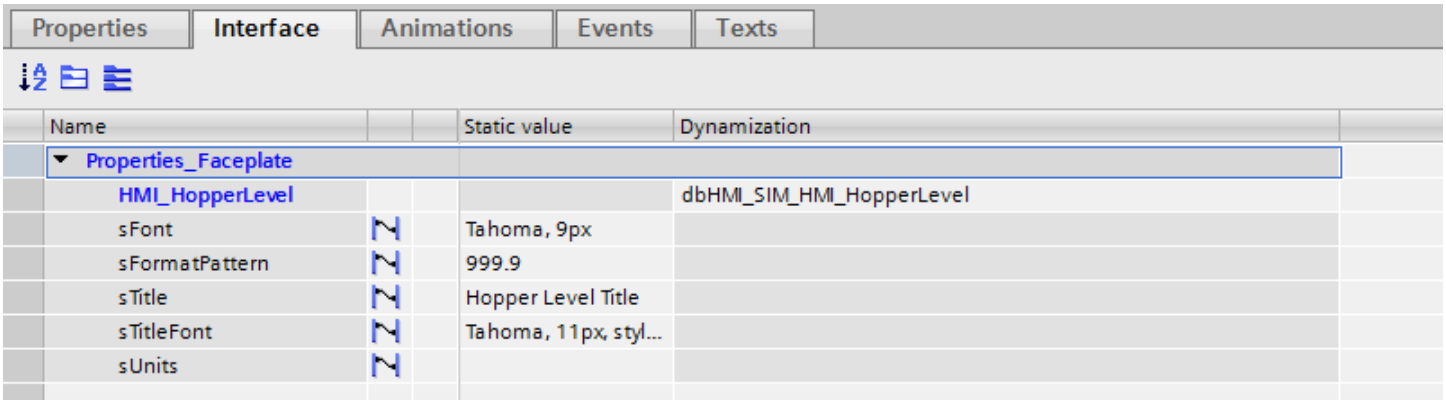
11.4.5.2. HMI Pop-up Appearance and Functionality

The table below gives a feature description for the pop up faceplate:

Object	Type	Description
Auto Mode Button	Button	Sets device to Auto Mode. Button is green when devices is in Auto Mode.
Manual Mode	Button	Sets device to Manual Mode. Button is green when device is in Manual Mode.
Current Effective Level	Output	Current level value being monitored
Manual/Override Level	Input	Manual override value uses when in manual mode
High Level Alarm	I/O Field	High level alarm setpoint
High Level Warning	I/O Field	High level warning setpoint
Low Level Warning	I/O Field	Low level warning setpoint
Low Level Alarm	I/O Field	Low level alarm setpoint
Status Field	Output	Monitor error status
Reset	Button	Reset button to clear errors
Close	Button	Exists the pop up screen

11.4.5.3. HMI Pop-up Interface

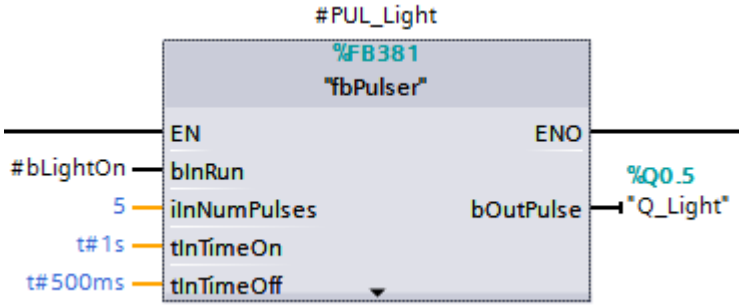
The device interface is shown below. Six properties are linked in this interface: HMI_HopperLevel, sFont, sFormatPattern, sTitle, sTitleFont, and sUnit. The HMI_ HopperLevel needs to be mapped to the same 'udtHMI_HopperLevel' used by the Function Block. The sFont property controls the font name used by text elements. The sFormatPattern property determines how the setpoint and actual values are displayed. The sTitle property requires a string passed into it, and this value will display at the top of the Pop-up. The sTitleFont property controls the font name used by the title text. The sUnit property controls the unit that is displayed on the setpoint and actual values.



11.5. Pulsing Output – fbPulser

11.5.1. Description

This block takes a input command and turns it into a pulsing output with configurable on/off times as well as a maximum count limit.



11.5.2. Function Block Interface

11.5.2.1. Input Parameters

Input Variables	Type	Description
bInRun	Bool	Enables the pulsing output
tInTimeOn	Time	Amount of time output is turned on each period
tInTimeOff	Time	Amount of time output is off each period
iInNumPulses	Int	Number of pulses to output (0 for infinite)

11.5.2.2. Out Parameters

Output Variables	Type	Description
bOutPulse	Bool	The pulsing output
bOutDone	Bool	Pusling complete, iNumPulses achieved
iOutCount	Int	Current number of pulses complete

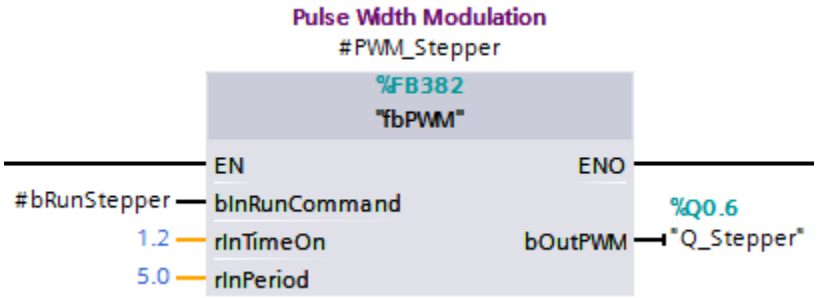
11.5.3. HMI Display

The Pulsing Output block does not currently have a provided HMI implementation.

11.6. PWM Control – fbPWM

11.6.1. Description

Similar to fbPulser, this block outputs a rough PWM signal with a configurable period and on time. NOTE: For small or specific time periods, this block may shift the pulse by a maximum of 1 scan length. To minimize error, place this block in a cyclic interrupt with a length that is a divisor of the period.



11.6.2. Function Block Interface

11.6.2.1. Input Parameters

Input Variables	Type	Description
bInRunCommand	Bool	Enables the PWM output
rInTimeOn	Real	Amount of time output is turned on each period, in seconds
rInPeriod	Real	The total time of 1 period of the PWM, in seconds

11.6.2.2. Out Parameters

Output Variables	Type	Description
bOutPWM	Bool	The PWM signal

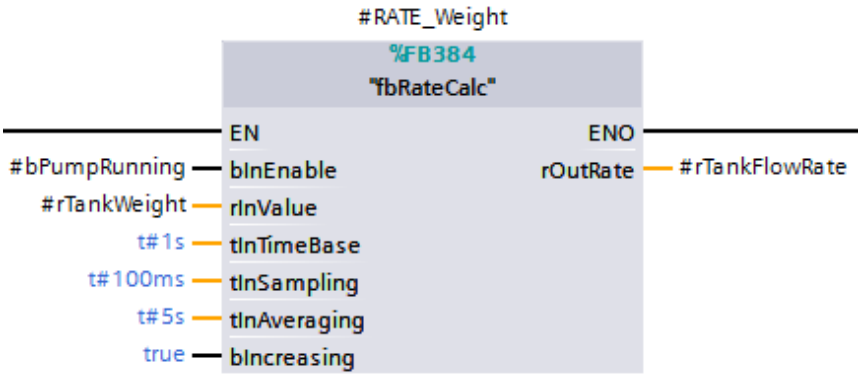
11.6.3. HMI Display

The PWM Output block does not currently have a provided HMI implementation.

11.7. Rate Calculation – fbRateCalc

11.7.1. Description

This block calculates the derivate (rate) of any process value over time with configurable sampling period and averaging. NOTE: This block should be called from a cyclic interrupt for maximum accuracy.



11.7.2. Function Block Interface

11.7.2.1. Input Parameters

Input Variables	Type	Description
bInEnable	Bool	Enables the rate calculation
rInValue	Real	The process value of which to calculate the rate
tInTimeBase	Time	The denominator of the rate you wish to calculate (e.g. 'rInValue' is in pounds and you desire to calculate pounds per minute, 'tInTimeBase' would be 'T#1m')
tInSampling	Time	Frequency of recording process values
tInAveraging	Time	Frequency of averaging of rate
bIncreasing	Bool	Indicates the rate is increasing (i.e. 'true') or descreasing (i.e. 'false')

11.7.2.2. Out Parameters

Output Variables	Type	Description
bOutRate	Real	The calculated derivative/rate

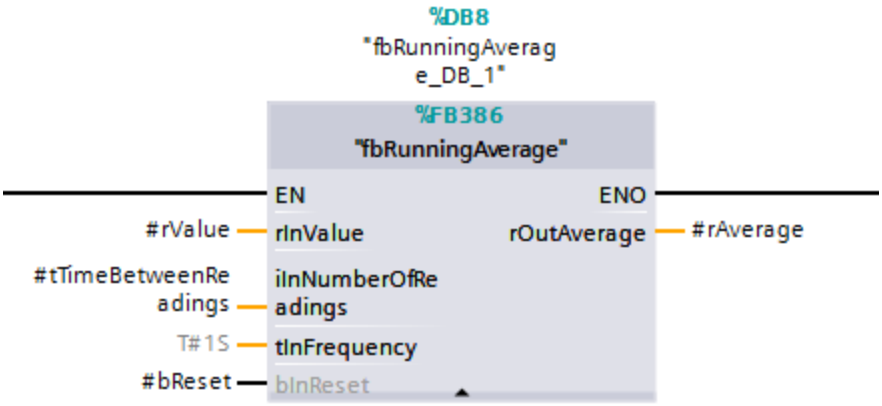
11.7.3. HMI Display

The Rate block does not currently have a provided HMI implementation.

11.8. Running Average – fbRunningAverage

11.8.1. Description

This block calculates the running average of a single input over a specified period. NOTE: For more accurate timing this block should be called from a cyclic interrupt, with tInSampleInterval set to zero.



11.8.2. Function Block Interface

11.8.2.1. Input Parameters

Input Variables	Type	Description
rInValue	Real	The current process value
iInNumberOfReadings	Int	Number of readings to average together.
tInFrequency	Time	Frequency at which to gather readings, if using a cyclic interrupt set this to zero, and call the cyclic interrupt at the desired sample frequency.
bInReset	Bool	Reset running average.

11.8.2.2. Out Parameters

Output Variables	Type	Description
bOutAverage	Real	The average of the process value over the specified time

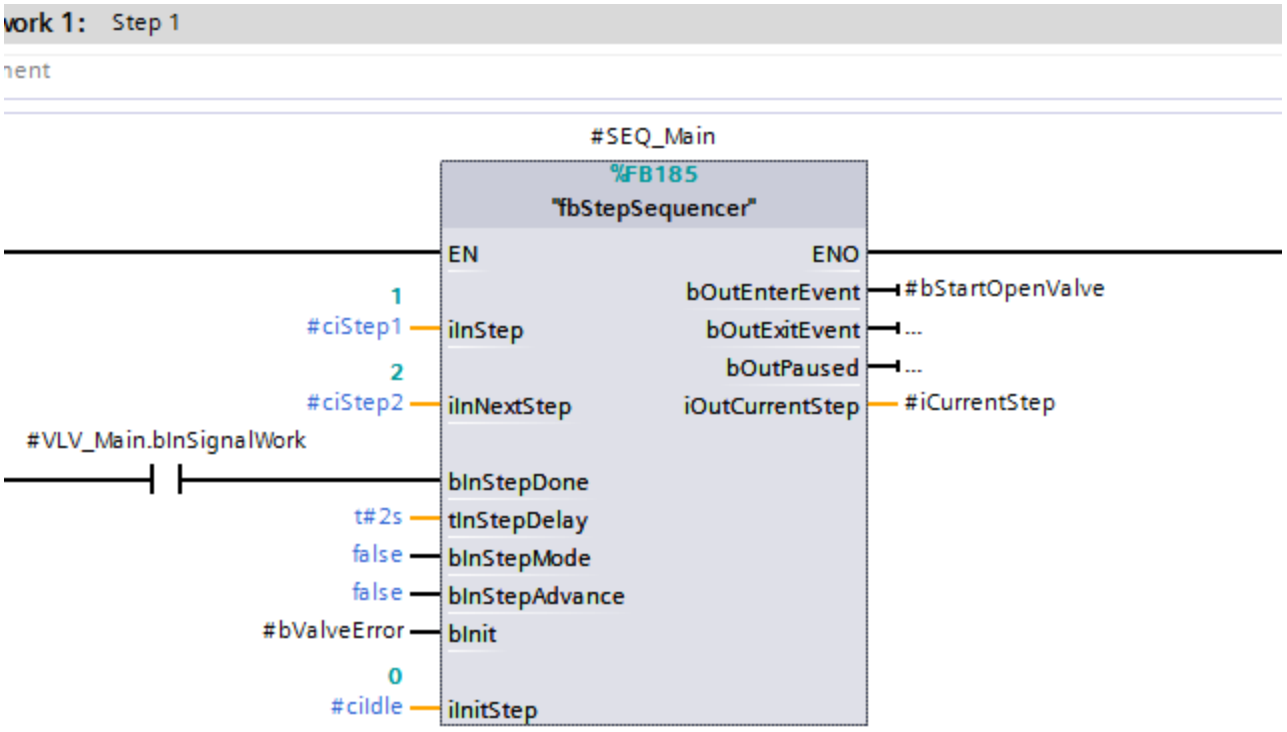
11.8.3. HMI Display

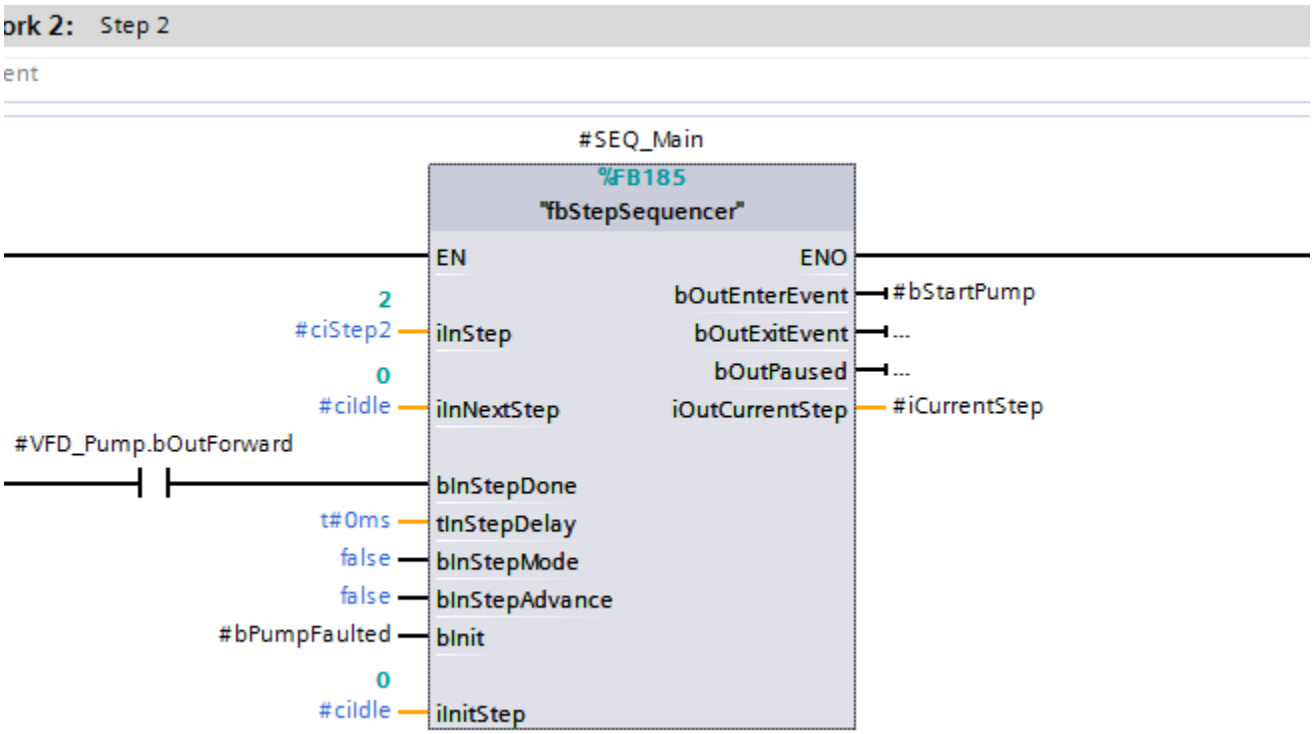
The Running Average block does not currently have a provided HMI implementation.

11.9. Sequencer – fbStepSequencer

11.9.1. Description

This library object provides an easy way to making sequencers and state machines in ladder logic. The block is intended to be used by multiple instances, one for each step or state, that all share the same instance memory. It is important that each state machine or sequence use the same instance. If you are creating more than one state machine, you will need a single instance for each. For example, we want 3 state machines, each containing 5 different states. We would need to create 3 instance memory allocations of fbStepSequencer and 15 usages of the block. Below is an example of a sequencer with 3 steps (step 1 – open valve, step 2 – start pump, and idle – stop pump, close valve). Note that both block usages have the same instance memory (SEQ_Main).


work 1: Step 1
#SEQ_Main
%FB185
"fbStepSequencer"
EN
ENO
1
#ciStep1
iInStep
bOutEnterEvent
#bStartOpenValve
bOutExitEvent
2
#ciStep2
iInNextStep
bOutPaused
iOutCurrentStep
#iCurrentStep
#VLV_Main.bInSignalWork
bInStepDone
t#2s
tInStepDelay
false
bInStepMode
false
bInStepAdvance
#bValveError
bInit
0
#ciIdle
iInitStep


ork 2: Step 2
#SEQ_Main
%FB185
"fbStepSequencer"
EN
ENO
2
#ciStep2
iInStep
bOutEnterEvent
#bStartPump
bOutExitEvent
0
#ciIdle
iInNextStep
bOutPaused
iOutCurrentStep
#iCurrentStep
#VFD_Pump.bOutForward
bInStepDone
t#0ms
tInStepDelay
false
bInStepMode
false
bInStepAdvance
#bPumpFaulted
bInit
0
#ciIdle
iInitStep

11.9.2. Functional Block Interface

11.9.2.1. Input Parameters

Input Variables	Type	Description
iInStep	Int	Constant, the step which this block instance represents/
iInNextStep	Int	Next step to go to when this step is done. Will be moved into iCurrentStep when bInStepDone is true.
bInStepDone	Bool	If true and iCurrentStep = iInStep, then we move to the next step.
tInStepDelay	Time	Time between when the bInStepDone rises to when we move states, set to 0 to disable.
bInStepMode	Bool	This input will pause the state machine at the completion of the current step until bInStepAdvance is set high.
bInStepAdvance	Bool	When the state machine is paused, this input will trigger the next step.
bInit	Bool	Initializes the state machine to step iInitStep.
iInitStep	Int	Step that the machine will be initialized to when bInit is true.

11.9.2.2. Out Parameters

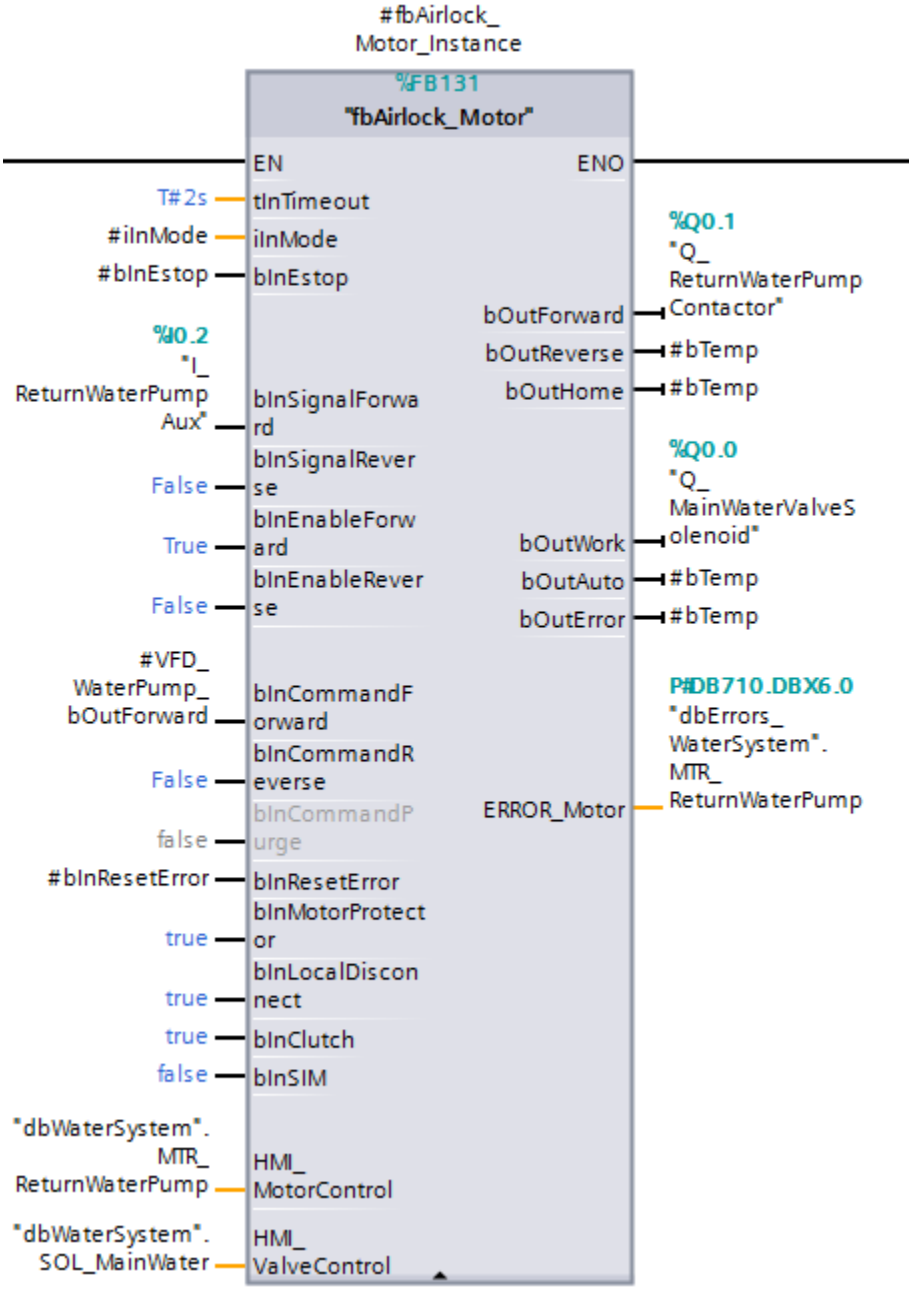
Output Variables	Type	Description
bOutEnterEvent	Bool	High for one scan when the state machine enters the state specified by iInStep for a single block usage.
bOutExitEvent	Bool	High for one scan when the state machine leaves iInStep.
bOutPaused	Bool	Indicates if the sequencer is paused and waiting on a bInStepAdvance signal.
iOutCurrentStep	Int	Current state/step.

12. Additional Control Devices

12.1. Airlock Motor - fbAirlock_Motor

12.1.1. Description

The Air Lock Motor Function Block is used to control airlocks, or interlocks, consisting of motor and solenoid valve controlled by digital outputs.



12.1.2. Function Block Interface

12.1.2.1. Input Paramiters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error

iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
bInSignalForward	Bool	Forward feedback bit. This tells the motor function block whether the motor is going in the forward direction. If no feedback input is available, wire the forward output to the forward feedback input
bInSignalReverse	Bool	Reverse feedback bit. This tells the motor function block whether the motor is going in the reverse direction. If no feedback input is available, wire the forward output to the forward feedback input
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will continue to run in this direction until the reverse interlock becomes false, or this input is set low
bInCommandPurge	Bool	Sets the valve to work, if mode is set to 1 and sets the valve's mode to 1.
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be wired directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled, and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled, and an error is triggered
bInClutch	Bool	This input should be wired directly to the motor's clutch input. When false, the motor is disabled, and an error is triggered
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

12.1.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_MotorControl	udtHMI_MotorControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands
HMI_ValveControl	udtHMI_ValveControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the cylinder in manual mode

12.1.2.3. Out Parameters

Output Variables	Type	Description
bOutForward	Bool	This output should be wired to the motor's forward contact
bOutReverse	Bool	This output should be wired to the motor's reverse contact
bOutHome	Bool	Output for home position
bOutWork	Bool	Output for work position
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the motor
ERROR_Motor	udtError_Motor	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

12.1.3. User Defined Types

12.1.3.1. udtHMI_MotorControl

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Forward	Bool	Move forward in manual mode pushbutton
bPB_Reverse	Bool	Move Reverse in manual mode pushbutton
bPB_Stop	Bool	Stop in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Errors pushbutton enabled
bPBEN_Forward	Bool	Forward pushbutton enabled
bPBEN_Reverse	Bool	Reverse pushbutton enabled
bPBEN_Stop	Bool	Stop pushbutton enabled
bForwardOn	Bool	Forward command is on
bReverseOn	Bool	Reverse command is on
bSignalForward	Bool	Forward signal
bSignalReverse	Bool	Reverse signal
bError	Bool	Overall error
bInterlock	Bool	Motor interlocked

12.1.3.2. udtHMI_ValveControl

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Home	Bool	Move to home in manual mode pushbutton
bPB_Work	Bool	Move to work in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Error pushbutton enabled
bPBEN_Home	Bool	Home pushbutton enabled
bPBEN_Work	Bool	Work pushbutton enabled
bHomeOn	Bool	Home command is on
bWorkOn	Bool	Work command is on
bSignalHome	Bool	Home feedback
bSignalWork	Bool	Work feedback
bError	Bool	Error status
bInterlock	Bool	Valve interlocked

12.1.3.3. udtError_Motor

Name	Description
MotorProtectorTripped	Motor protector tripped
LocalDisconnectOff	Local disconnect off
ClutchTripped	Clutch tripped
NoSignalForward	No signal from forward contactor
NoSignalReverse	No signal from Reverse contactor
MotorNotStopped	Motor doesn't stop

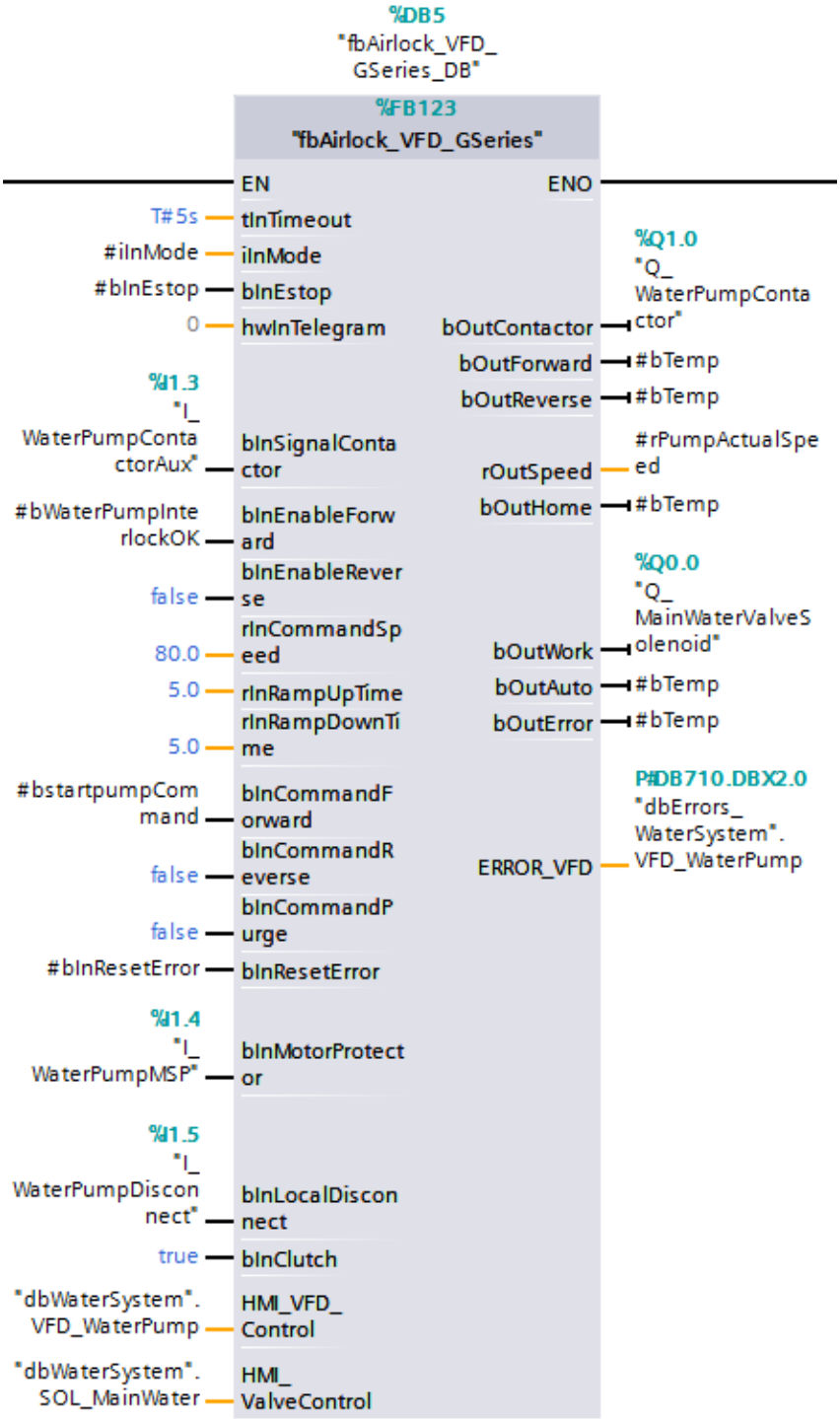
12.1.4. HMI Display

The Airlock Motor block utilizes a motor and a solenoid valve HMI objects. See their respective sections 6.4.4 and 7.1.4 for more details.

12.2. G Series VFD Airlock Motor – fbAirlock_VFD_GSeries

12.2.1. Description

The Air Lock Motor Function Block is used to control airlocks, or interlocks, consisting of motor controlled by a G series VFD and a solenoid valve controlled by digital outputs. The G Series VFD Airlock Motor controls Siemens G Series VFD or Micromaster VFDs using Standard Telegram 1. A technology object is not used by this block, which allows for a lower CPU requirement for each drive and avoids technology object count limitations.



12.2.2. Function Block Interface

12.2.2.1. Input Paramiters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
hwInTelegram	HW_Submodule	VFD standard telegram one HW address.
bInSignalContactor	Bool	Contactor feedback bit. This tells a VFD function block whether the contactor is in the correct state. If no feedback input is available, wire the contactor output, bOutContactor, to the feedback input
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
rInCommandSpeed*	Real	Speed set point for automatic mode in percent. (0.0-100.0%)
rInRampUpTime	Real	The ramp up time for the VFD in seconds (S7-1500 ONLY)
rInRampDownTime	Real	The ramp down time for the VFD in seconds (S7-1500 ONLY)
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will run in this direction until the reverse interlock becomes false, or this input is set low
bInCommandPurge	Bool	Sets the valve to work, if mode is set to 1 and sets the valve's mode to 1.
bInResetError	Bool	If an error condition exists, the error must be fixed first, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be wired directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled, and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled, and an error is triggered
bInClutch	Bool	This input should be wired directly to the motor's clutch input. When false, the motor is disabled, and an error is triggered
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

12.2.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_VFD_Control	udtHMI_VFD_Control	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward,

		reverse, and stop commands. Also inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals
HMI_ValveControl	udtHMI_ValveControl	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the cylinder in manual mode

12.2.2.3. Out Parameters

Output Variables	Type	Description
bOutContactor	Bool	Contactor output. Map to Output to turn on contactor if used
bOutForward	Bool	This output should be wired to the motor's forward contact
bOutReverse	Bool	This output should be wired to the motor's reverse contact
rOutSpeed	Real	Actual speed from VFD
bOutHome	Bool	Output for home position
bOutWork	Bool	Output for work position
bOutAuto	Bool	This output indicates whether the block is in auto mode
bOutError	Bool	This output indicates whether there's an error condition in the motor
ERROR_VFD	udtError_VFD	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

12.2.3. User Defined Types

12.2.3.1. udtHMI_VFD_Control

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
rManualSpeedSP	Real	Speed set point for manual mode
rAutoSpeedSP	Real	Speed set point for automatic mode
rActualSpeed	Real	Actual motor speed
rActualCurrent	Real	Actual motor current
rActualPower	Real	Actual motor power
bPB_ResetError	Bool	Pushbutton Reset errors
bPB_Forward	Bool	Pushbutton Move forward in manual mode

bPB_Reverse	Bool	Move reverse in manual mode pushbutton
bPB_Stop	Bool	Stop in manual mode pushbutton
bPBEN_ResetError	Bool	Reset errors pushbutton enabled
bPBEN_Forward	Bool	Forward pushbutton enabled
bPBEN_Reverse	Bool	Reverse pushbutton enabled
bPBEN_Stop	Bool	Stop pushbutton enabled
bForwardOn	Bool	Run forward command is on
bReverseOn	Bool	Run reverse command is on
bSignalForward	Bool	Forward signal
bSignalReverse	Bool	Reverse signal
bError	Bool	Overall error
bInterlock	Bool	VFD Interlocked

12.2.3.2. udtHMI_ValveControl

Name	Data Type	Description
iMode	Int	Current mode
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
iStatus	Int	Status for HMI display. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_Home	Bool	Move to home in manual mode pushbutton
bPB_Work	Bool	Move to work in manual mode pushbutton
bPBEN_ResetError	Bool	Reset Error pushbutton enabled
bPBEN_Home	Bool	Home pushbutton enabled
bPBEN_Work	Bool	Work pushbutton enabled
bHomeOn	Bool	Home command is on
bWorkOn	Bool	Work command is on
bSignalHome	Bool	Home feedback
bSignalWork	Bool	Work feedback
bError	Bool	Error status
bInterlock	Bool	Valve interlocked

12.2.3.3. udtError_VFD

Errors	Description

MotorProtectorTripped	Motor Protector tripped
LocalDisconnectOff	Motor Local switch OFF
ClutchTripped	Clutch error
NoContactorFeedback	No feedback from motor contactor
ContactorStillOn	Contactor still ON
NoSignalForward	No SIGNAL Forward from VFD
NoSignalReverse	No SIGNAL Reverse from VFD
MotorNotStopped	Motor doesn't stop
MaxFrequencyReached	Max frequency reached
Overcurrent	VFD overcurrent
MotorOverload	Motor overload
VFDOverload	VFD overload
GeneralFault	VFD fault
NoCoolingFanFeedback	Cooling fan did not come on within 30 seconds

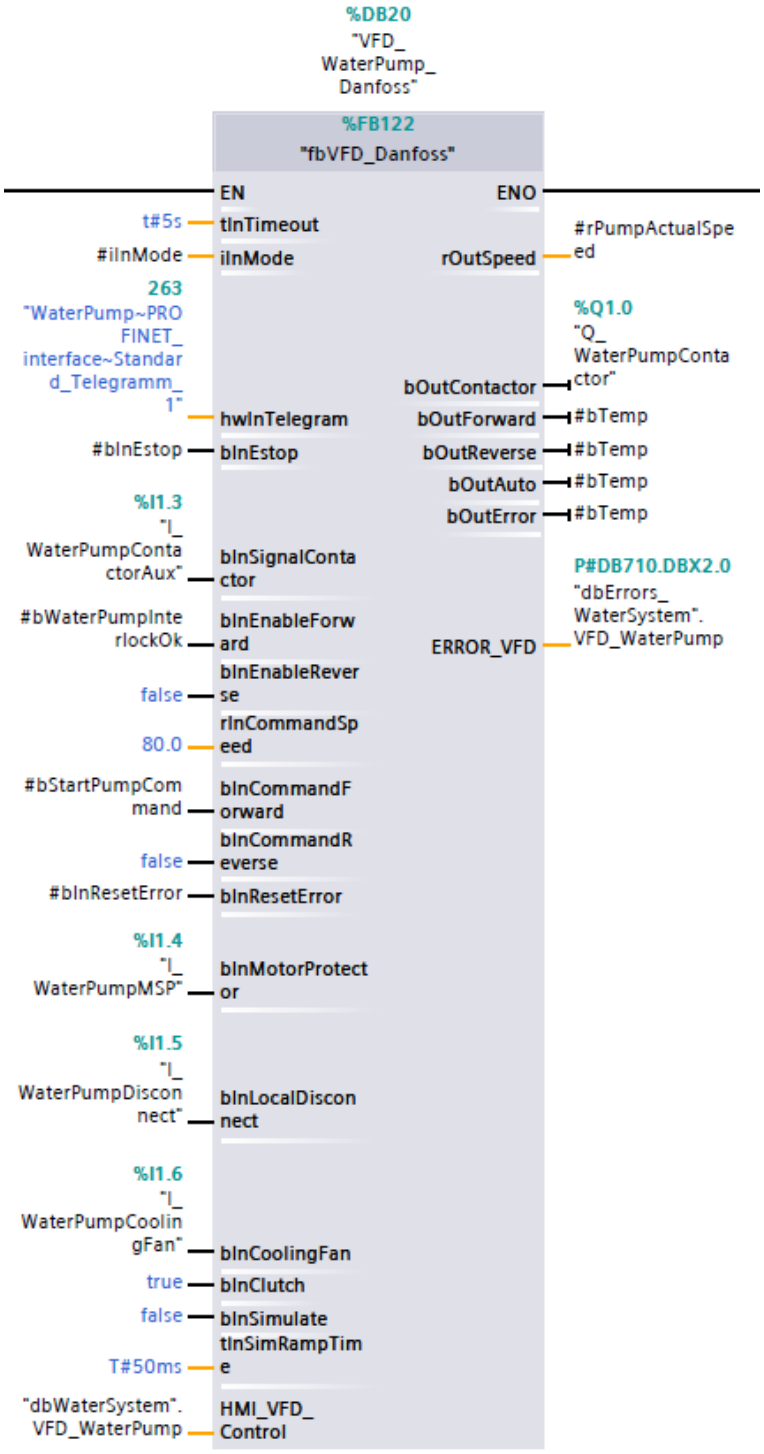
12.2.4. HMI Display

The Airlock VFD block utilizes a VFD and a solenoid valve HMI objects. See their respective section 6.1.4 and 7.1.4 for more details.

12.3. Danfoss VLT Series Motor Control – fbVFD_Danfoss

12.3.1. Description

The Danfoss VFD Control Function Block controls Danfoss VLT® low-voltage Automation Drives using standard telegram 1. Danfoss VLT® drives has two different sets of Control and Status words which vary based on the communication protocol being used; either Profidrive or VLT Standard communication. This block assumes communication occurs via Profidrive. This block uses the same error and HMI udts as fbVFD_GSeries and fbVFD_Analog.



12.3.2. Function Block Interface

12.3.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error
iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
hwInTelegram	HW_SUBMODULE	The system constant that connects to the Telegram 1 configuration of the VFD
bInSignalContactor	Bool	Contactor feedback. Map the Output bOutContactor if no Contactor exists.
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
rInCommandSpeed*	Real	Speed set point for automatic mode in percent. (0.0-100.0%)
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will run in this direction until the reverse interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be mapped directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInClutch	Bool	This input should be wired directly into the motor's clutch overload input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered

bInCoolingFan	Bool	This input is used to tell the VFD that the cooling fan is running, if available. If not available, this input should be set to true. When false for 30 seconds after VFD starts, the motor is disabled and an error is triggered
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

12.3.2.2. 7.9.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_VFD_Control	udtHMI_VFD_Control	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

7.9.2.3 Out Parameters

Output Variables	Type	Description
bOutContactor	Bool	Contactor output. Map to Output to turn on contactor if used
bOutForward	Bool	Motor running forward feedback from VFD
bOutReverse	Bool	Motor running reverse feedback from VFD
rOutSpeed	Real	Actual speed from VFD
bOutError	Bool	This output indicates whether there's an error condition in the VFD
bOutAuto	Bool	Output indicating whether the block is in auto mode
ERROR_VFD	udtError_VFD	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

12.3.3. User Defined Types

See section 6.1.3 for more details.

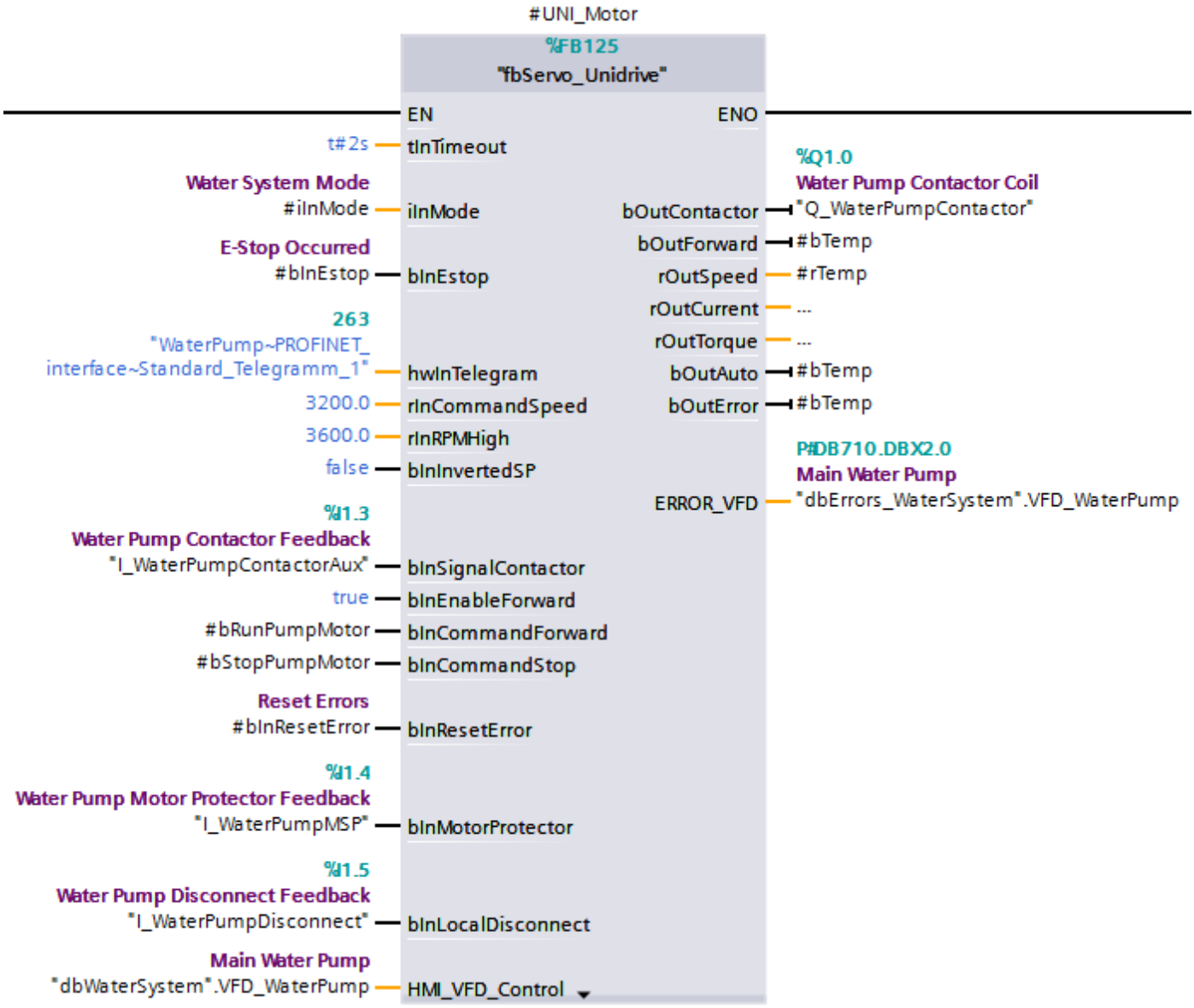
12.3.4. HMI Display

The Danfoss VFD block utilizes identical HMI objects as the GSeries VFD. See section 6.1.4 for more details.

12.4. Unidrive Servo – fbServo_Unidrive

12.4.1. Description

This block controls a Control Techniques (Emerson) servo drive with speed control and a standard telegram. Functionality is identical to fbVFD_GSeries.



12.4.2. Function Block Interface

12.4.2.1. Input Parameters

Input Variables	Type	Description
tInTimeout	Time	The amount of time before an error condition triggers an error

iInMode	Int	Device mode input. See Section 12.1.2 of the Library Overview and Architecture document of the Library Overview and Architecture document for further details
bInEstop	Bool	Emergency stop input
hwInTelegram	HW_SUBMODULE	The system constant that connects to the Telegram 1 configuration of the VFD
bInSignalContactor	Bool	Contactor feedback. Map the Output bOutContactor if no Contactor exists.
bInEnableForward	Bool	Forward interlock. The motor is allowed to run in the forward direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
bInEnableReverse	Bool	Reverse interlock. The motor is allowed to run in the reverse direction if this condition has been met. If the motor should always be able to move in this direction, this input should be set to true. To disable the motor from ever traveling in this direction, this input should be set to false
rInCommandSpeed*	Real	Speed set point for automatic mode in RPM. (0 - rInRPMHigh)
rInRPMHigh	Real	VFD configured maximum RPM limit
bInInvertedSP	Bool	Inverts the RPM set point sent to the drive
bInCommandForward*	Bool	This is the forward command for auto mode. If this input is set high in auto mode the motor will run in this direction until the forward interlock becomes false, or this input is set low
bInCommandReverse*	Bool	This is the reverse command for auto mode. If this input is set high in auto mode the motor will run in this direction until the reverse interlock becomes false, or this input is set low
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error
bInMotorProtector	Bool	This input should be mapped directly to the motor's circuit breaker input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInLocalDisconnect	Bool	This input should be wired directly to the motor's safety disconnect switch input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInClutch	Bool	This input should be wired directly into the motor’s clutch overload input, if available. If not available, this input should be set to true. When false, the motor is disabled and an error is triggered
bInSwapBytes	Bool	Signals the block to swap the bytes received from the telegram
bInSimulate	Bool	Enables software simulation of device.

*Only valid in Automatic Mode

12.4.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_VFD_Control	udtHMI_VFD_Control	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

7.9.2.3 Out Parameters

Output Variables	Type	Description
bOutContactor	Bool	Contactor output. Map to Output to turn on contactor if used
bOutForward	Bool	Motor running forward feedback from VFD
bOutReverse	Bool	Motor running reverse feedback from VFD
rOutSpeed	Real	Actual speed from VFD
bOutError	Bool	This output indicates whether there's an error condition in the VFD
bOutAuto	Bool	Output indicating whether the block is in auto mode
ERROR_VFD	udtError_VFD	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

12.4.3. User Defined Types

See section 6.1.3 for more details.

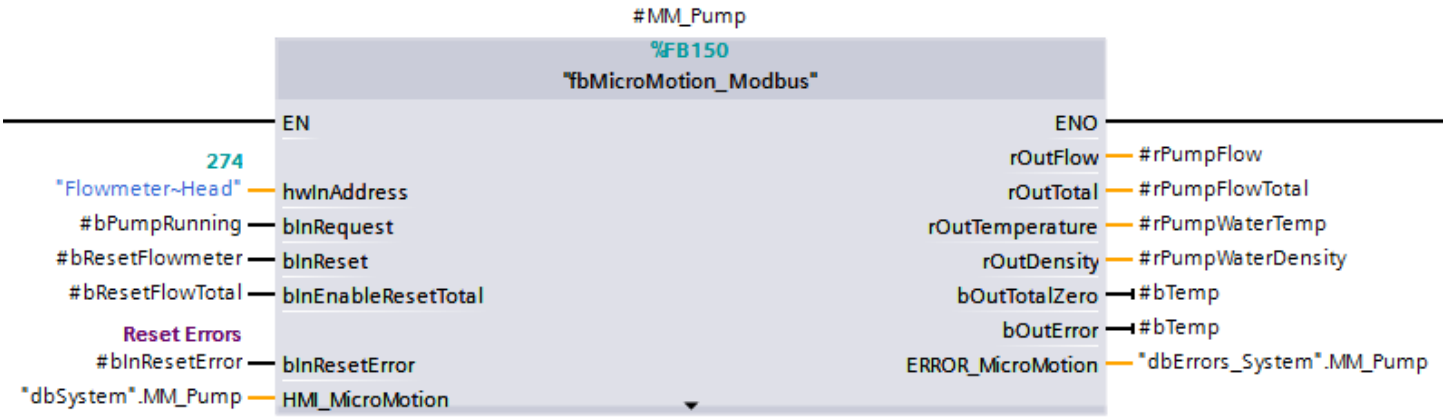
12.4.4. HMI Display

The Unidrive Servo block utilizes identical HMI objects as the GSeries VFD. See section 6.1.4 for more details.

12.5. MicroMotion (Modbus) – fbMicroMotion_Modbus

12.5.1. Description

This block controls a MicroMotion Coriolis flow meter using the Modbus protocol. This block provides information from the meter such as flow rate, flow total, and liquid density.



12.5.2. Function Block Interface

12.5.2.1. Input Parameters

Input Variables	Type	Description
hwInAddress	HW_ANY	The hardware address of the Modbus device configuration
bInRequest	Bool	Indicates that flow through the meter is expected
bInReset	Bool	Manually resets the totalizer on the flow meter
bInEnableResetTotal	Bool	Enables the totalizer reset functionality from the HMI
bInSwapBytes	Bool	Swaps the bytes (endianness) of all messages received, if necessary
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error

12.5.2.2. InOut Parameters

In/Out Variables	Type	Description
HMI_MicroMotion	udtHMI_MicroMotion	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

12.5.2.3. Out Parameters

Output Variables	Type	Description
rOutFlow	Real	The actual flow, units configured on device
rOutTotal	Real	The total flow measured since last reset, units configured on device

rOutTemperature	Real	The actual temperature of the fluid, units configured on device
rOutDensity	Real	The actual density of the fluid, units configured on device
bOutTotalZero	Bool	Flow totalizer successfully zeroed
bOutError	Bool	Error exists
rOutTareValue	Real	Current amount between module zero and current reading
ERROR_MicroMotion	udtError_MicroMotion	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

12.5.3. User Defined Types

12.5.3.1. udtHMI_MicroMotion

Name	Data Type	Description
rTotal	Real	The flow totalizer value
rFlow	Real	Device measured flow
rTemperature	Real	Temperature of the fluid in the flow meter
rDensity	Real	Density of the fluid in the flow meter
iErrorCode	Int	Error code. See Section 12.3.1 of the Library Overview and Architecture document for further details
bPB_ResetError	Bool	Reset errors pushbutton
bPB_ResetTotal	Bool	Manually resets the flow totalizer
bPBEN_ResetError	Bool	Reset errors pushbutton enabled
bPBEN_ResetTotal	Bool	Flow totalizer resetting enabled
bError	Bool	Device error exists

12.5.3.2. udtError_MicroMotion

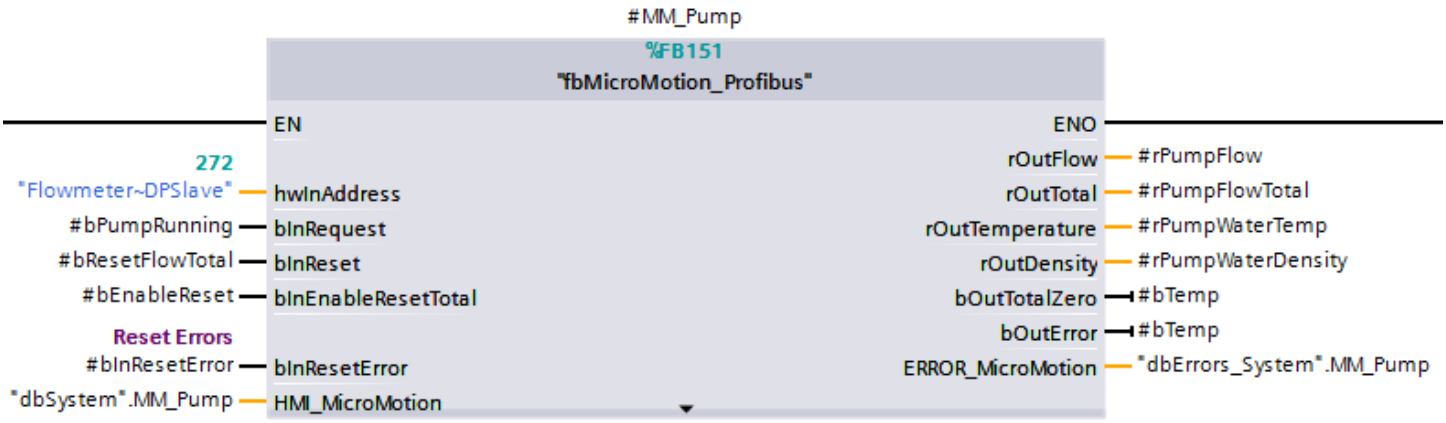
Name	Data Type	Description
ResetFailed	Bool	Failed to reset the flow totalizer
FlowWithoutRequest	Bool	Unexpected flow observed in the meter

12.5.4. HMI Display

The MicroMotion Modbus block does not currently have a provided HMI implementation.

12.6. MicroMotion (Profibus) – fbMicroMotion_Profibus

Identical to fbMicroMotion_Modbus with the exception that it uses the Profibus protocol for communication to the meter.



12.6.1. Function Block Interface

12.6.1.1. Input Parameters

Input Variables	Type	Description
hwInAddress	HW_ANY	The hardware address of the Modbus device configuration
bInRequest	Bool	Indicates that flow through the meter is expected
bInReset	Bool	Manually resets the totalizer on the flow meter
bInEnableResetTotal	Bool	Enables the totalizer reset functionality from the HMI
bInResetError	Bool	If an error condition exists, the error must first be fixed, then set this bit high to reset the internal error

12.6.1.2. InOut Parameters

In/Out Variables	Type	Description
HMI_MicroMotion	udtHMI_Micro Motion	This in/out UDT is used to communicate to the HMI. Inside the UDT are all the variables needed to control the motor in manual mode, including forward, reverse, and stop commands. Also inside this UDT are all feedback signals including actual speed, actual current, and the forward/reverse signals

12.6.1.3. Out Parameters

Output Variables	Type	Description
rOutFlow	Real	The actual flow, units configured on device
rOutTotal	Real	The total flow measured since last reset, units configured on device
rOutTemperature	Real	The actual temperature of the fluid, units configured on device
rOutDensity	Real	The actual density of the fluid, units configured on device

bOutTotalZero	Bool	Flow totalizer successfully zeroed
bOutError	Bool	Error exists
rOutTareValue	Real	Current amount between module zero and current reading
ERROR_MicroMotion	udtError_MicroMotion	This output UDT has one bit for every type of error. It should be mapped to the HMI's error list. It can also be used for advanced debugging of the error state

12.6.2. User Defined Types

See section 12.5.3.

12.6.3. HMI Display

The MicroMotion Profibus block does not currently have a provided HMI implementation.