Smart People. Expert Solutions.

SIEMENS OPEN LIBRARY

6 – PID Configuration

JUNE 11, 2019

Contents

1. Purpose 3
2. Intended Use 3
3. Revision History 3
4. Open Library License 3
5. Open Library PID Compact Setup 5
5.1. Initial Setup 5
5.2. PLC 5
5.2.1. Creating PID Technology Object 5
5.3. HMI 15

1. Purpose

The purpose of this document is to assist with configuration of the PID Open Library Object. In order to use the features of the built in Technology Object for the PID Compact Block, the library object requires special setup. The PID Comact Block only exists on the S7-1200 and S7-1500. The Open Library Object was written to be compatible with PID_Compact V2.2, and capabilities cannot be guaranteed for other versions of the PID Compact block.

2. Intended Use

This document is to be used by anyone utilizing the Siemens Open Library fbPID_CompactInterface. This document is used to configure the PLC and HMI objects for the library, as the configuration of these objects is not standard with the rest of the library.

3. Revision History

Version	Date	Author	Comments
1.0	2016-05-23	DMC	Initial Release
1.1	2016-06-20	DMC	Updated screenshots for PID_Compact and the interface function block
1.2	2016-08-23	DMC	No Changes
1.3	2016-10-11	DMC	No Changes
1.4	2017-06-27	DMC	Updated with new fbPID_Compact block and HMI_PID data type.
2.0	2017-11-3	DMC	No Changes
3.0	2018-12-05	DMC	Updates to data types used by block
4.0	201906-11	DMC	No Changes

4. Open Library License

Copyright (c) 2019 DMC, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN

AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

5. Open Library PID Compact Setup

The following steps walk through the configuration of the PID Open Library Object. In order to use the features of the built-in Technology Object settings for the PID_Compact Block, the library object requires special setup.

The benefit of using the Library Interface block are as follows:

1. It provides an easy interface for operation and changes to the PID from an HMI faceplate.
2. It utilizes the technology object, so all of the PID functions built in to portal can still be used.

5.1. Initial Setup

Before starting this document please make sure you have set up your project following the steps in the Basic Setup document. Each block is dependent on global constants and clock memory bits, and will not compile without correctly completing the initial setup. The following steps need to be performed:

- Enable system and clock memory bytes on the CPU.
- Retrieve the Open Library.
- Pull the Open Library PLC tags into the project.
- Setup Mode Control, or understand of how Open Library Modes function.

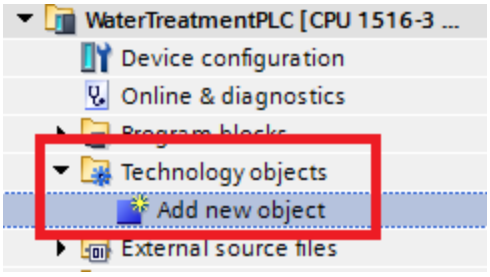
5.2. PLC

This section covers the setup required for the PLC programming. The following steps are required:

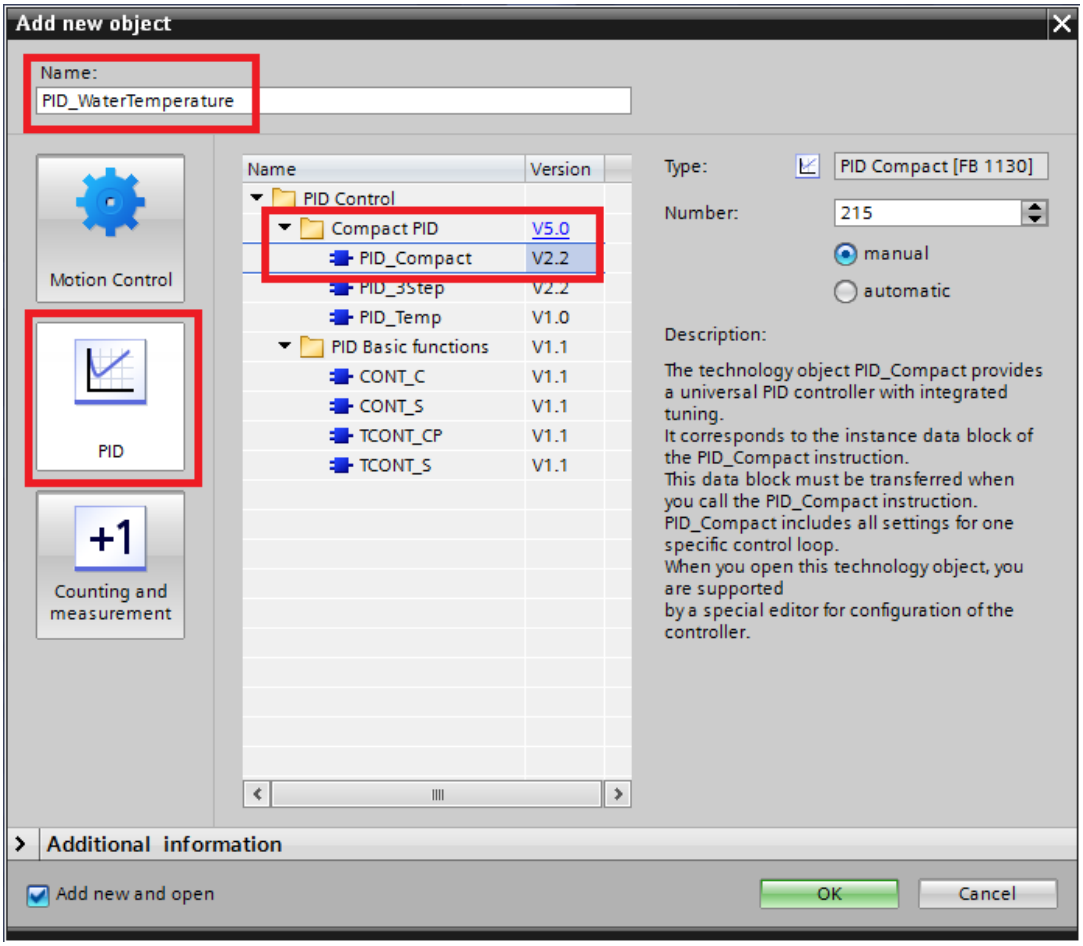
1. Create the PID Technology object
2. Create an FB and OB to call the fbPID_Compact block.
3. Map the fbPID_Compact block to the PID_Compact Technology Object.

5.2.1. Creating PID Technology Object

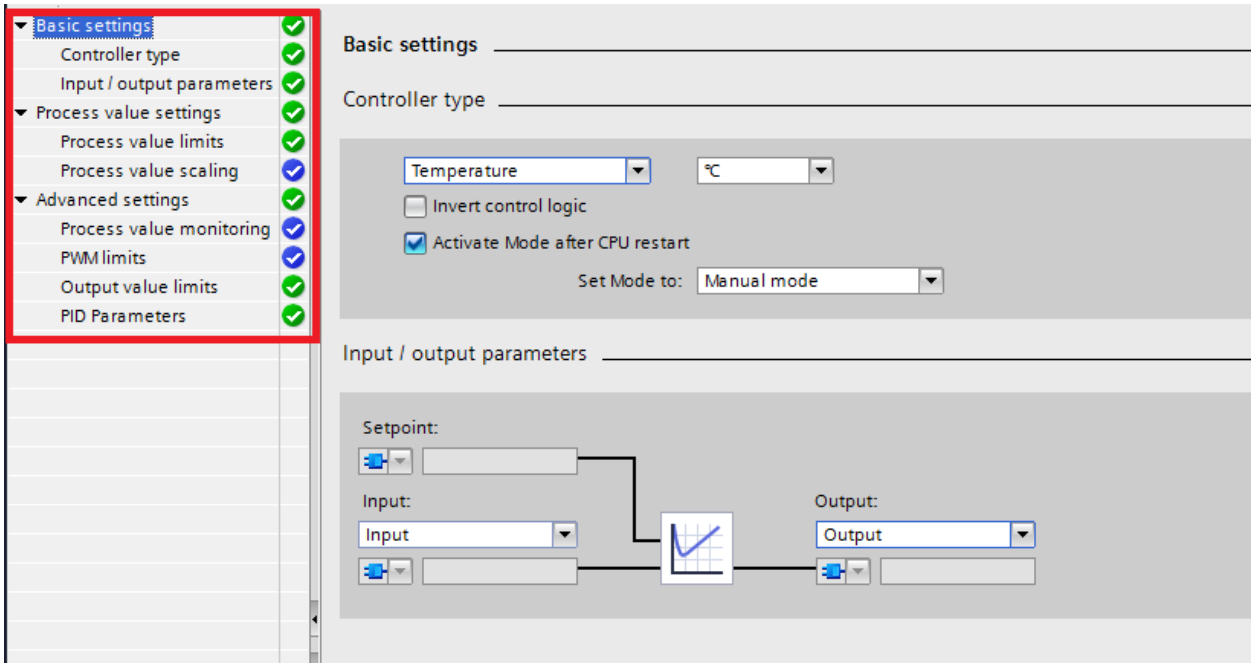
1. Start by creating a PID_Compact technology object. Expand the Technology Objects folder and click Add new object.



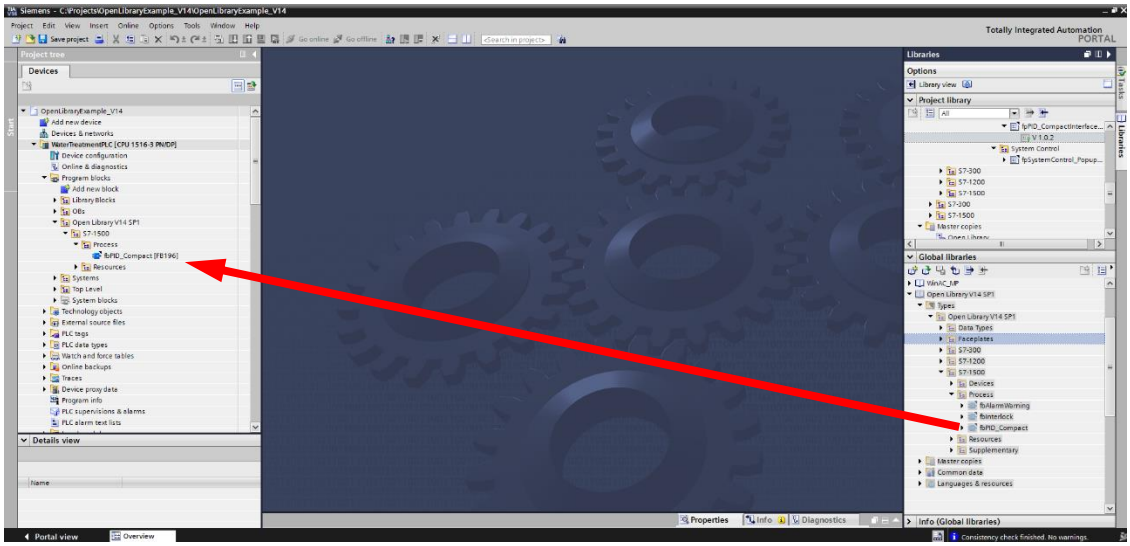
2. In the new object dialog box, give your object a name and select the PID_Compact v2.2 block.



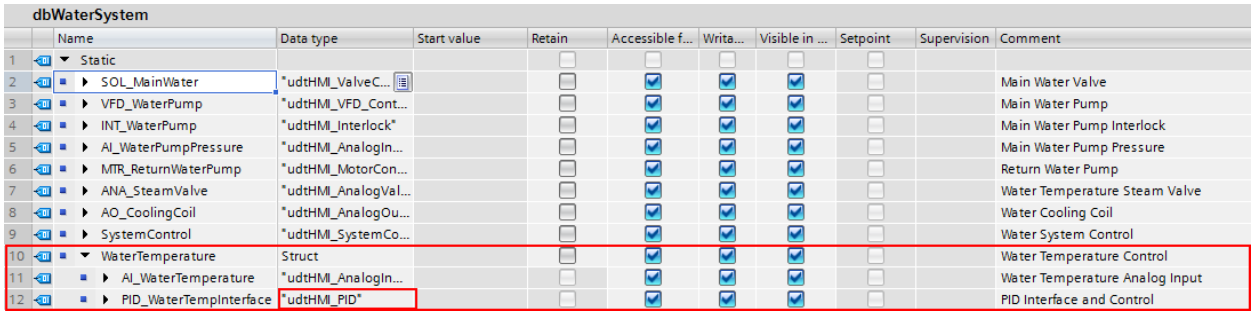
3. Configure your PID_Compact technology block to fit your physical system.



4. Pull fbPID_Compact from the Open Library Types group into the Library Blocks group to use it in our project.



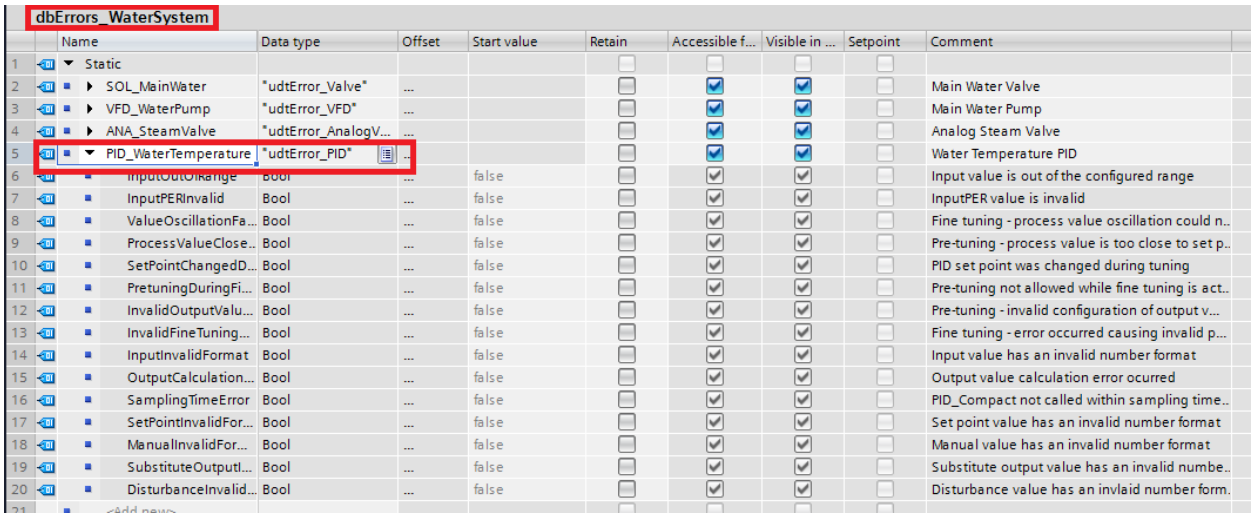
5. In a global data block (dbWaterSystem in this example), add a structure for all of your PID related variables. The required variables are seen in the screenshot below. This structure should include your HMI control for the PID system (udtHMI_PID).



dbWaterSystem

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervision	Comment
1	Static									
2	SOL_MainWater	"udtHMI_ValveC...			☑	☑	☑			Main Water Valve
3	VFD_WaterPump	"udtHMI_VFD_Cont...			☑	☑	☑			Main Water Pump
4	INT_WaterPump	"udtHMI_Interlock"			☑	☑	☑			Main Water Pump Interlock
5	AI_WaterPumpPressure	"udtHMI_AnalogIn...			☑	☑	☑			Main Water Pump Pressure
6	MTR_ReturnWaterPump	"udtHMI_MotorCon...			☑	☑	☑			Return Water Pump
7	ANA_SteamValve	"udtHMI_AnalogVal...			☑	☑	☑			Water Temperature Steam Valve
8	AO_CoolingCoil	"udtHMI_AnalogOu...			☑	☑	☑			Water Cooling Coil
9	SystemControl	"udtHMI_SystemCo...			☑	☑	☑			Water System Control
10	WaterTemperature	Struct			☑	☑	☑			Water Temperature Control
11	AI_WaterTemperature	"udtHMI_AnalogIn...			☑	☑	☑			Water Temperature Analog Input
12	PID_WaterTempInterface	"udtHMI_PID"			☑	☑	☑			PID Interface and Control

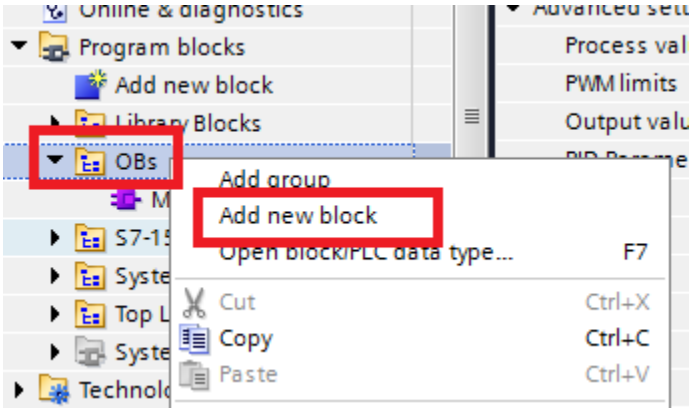
6. Similarly, add the error structure to the errors Data Block, dbErrors_WaterSystem in this example.



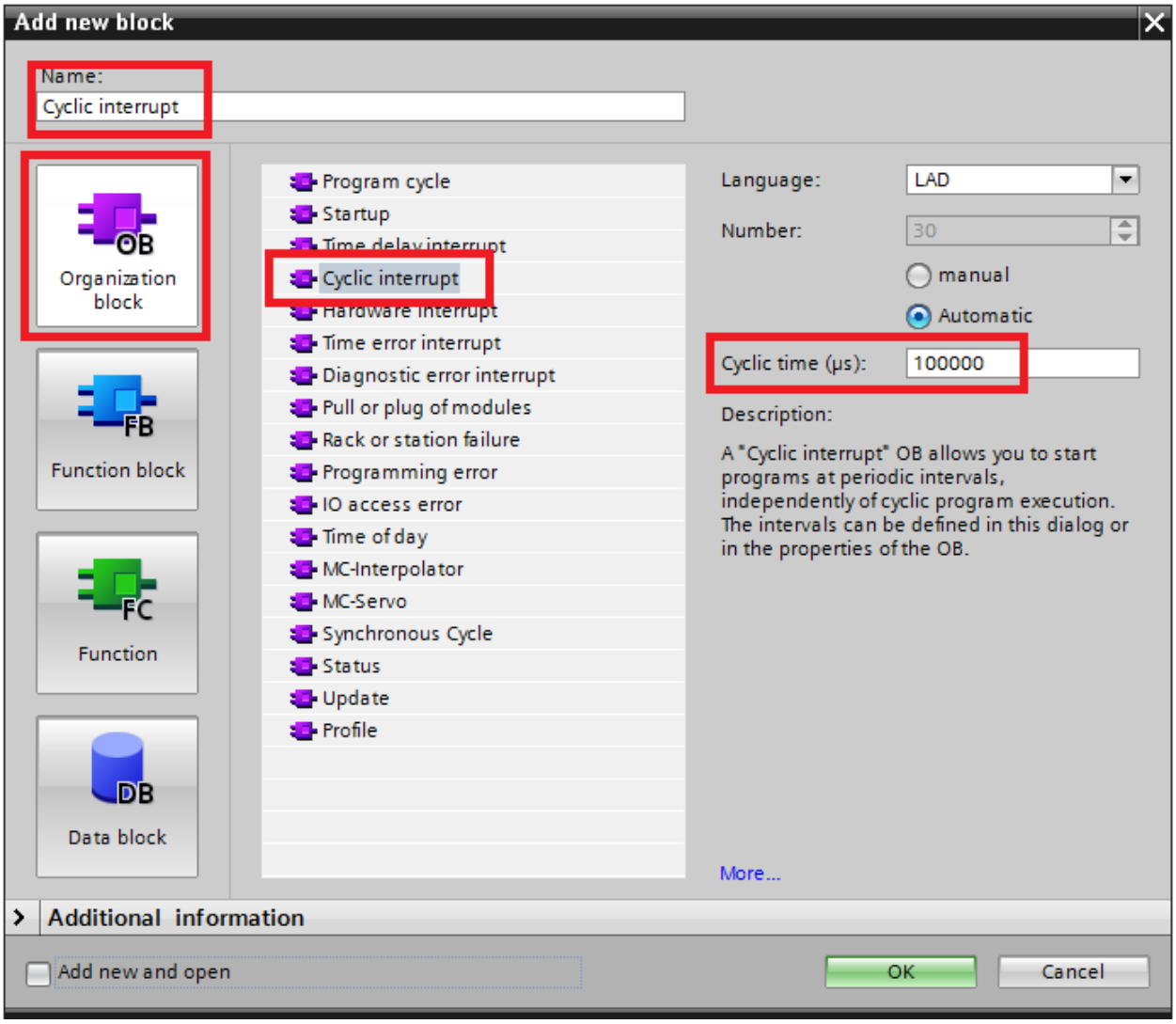
dbErrors_WaterSystem

	Name	Data type	Offset	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static								
2	SOL_MainWater	"udtError_Valve"	...			☑	☑		Main Water Valve
3	VFD_WaterPump	"udtError_VFD"	...			☑	☑		Main Water Pump
4	ANA_SteamValve	"udtError_AnalogV...	...			☑	☑		Analog Steam Valve
5	PID_WaterTemperature	"udtError_PID"	...			☑	☑		Water Temperature PID
6	InputOutOfRange	Bool	...	false		☑	☑		Input value is out of the configured range
7	InputPERInvalid	Bool	...	false		☑	☑		InputPER value is invalid
8	ValueOscillationFa...	Bool	...	false		☑	☑		Fine tuning - process value oscillation could n...
9	ProcessValueClose...	Bool	...	false		☑	☑		Pre-tuning - process value is too close to set p...
10	SetPointChangedD...	Bool	...	false		☑	☑		PID set point was changed during tuning
11	PretuningDuringFi...	Bool	...	false		☑	☑		Pre-tuning not allowed while fine tuning is act...
12	InvalidOutputValu...	Bool	...	false		☑	☑		Pre-tuning - invalid configuration of output v...
13	InvalidFineTuning...	Bool	...	false		☑	☑		Fine tuning - error occurred causing invalid p...
14	InputInvalidFormat	Bool	...	false		☑	☑		Input value has an invalid number format
15	OutputCalculation...	Bool	...	false		☑	☑		Output value calculation error ocurred
16	SamplingTimeError	Bool	...	false		☑	☑		PID_Compact not called within sampling time...
17	SetPointInvalidFor...	Bool	...	false		☑	☑		Set point value has an invalid number format
18	ManualInvalidFor...	Bool	...	false		☑	☑		Manual value has an invalid number format
19	SubstituteOutputI...	Bool	...	false		☑	☑		Substitute output value has an invalid numbe...
20	DisturbanceInvalid...	Bool	...	false		☑	☑		Disturbance value has an invlaid number form...

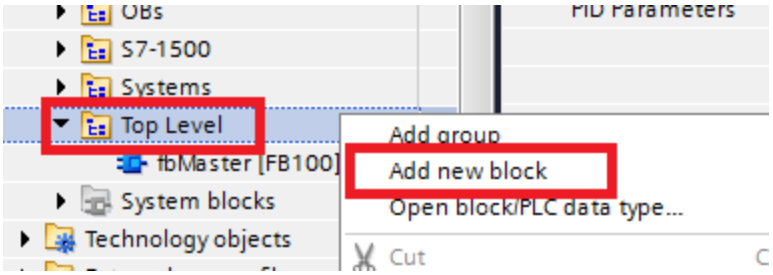
7. Add a new Cyclic Interrupt OB that will call our time sensitive cyclic operations.



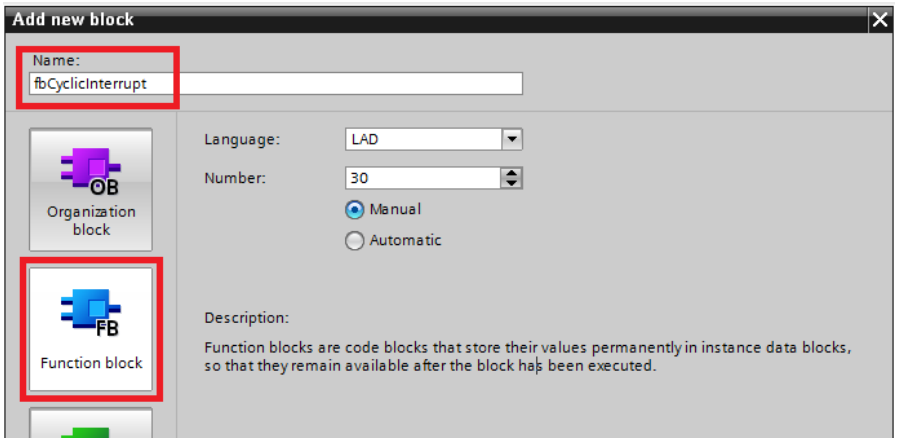
8. Give the cyclic interrupt OB a name and set the cycle time to something appropriate for the application. In this case, temperature does not need a fast cycle so it was set to 100ms (100,000µs).



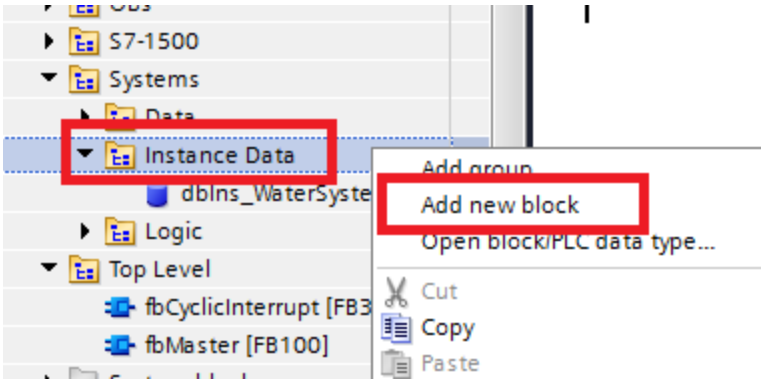
9. Add a new function block that will call our PID_Compact and will be called from the cyclic interrupt OB. It is recommended best practice to limit logic inside an OB, and the library object requires instance memory, so we will use the FB to encapsulate our cyclic operation logic.



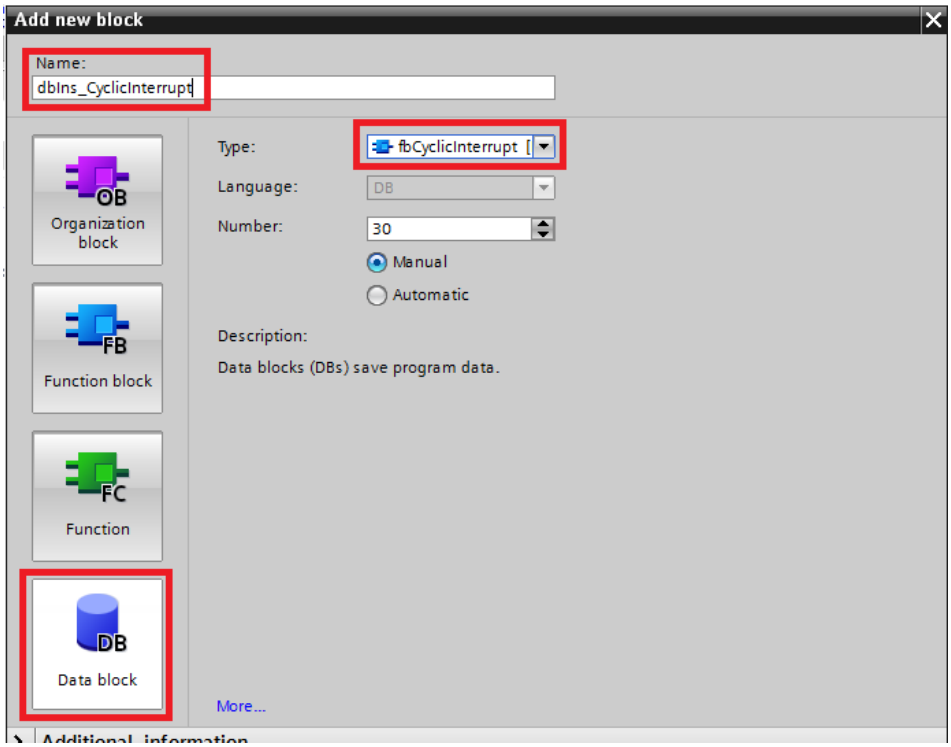
10. Name the cyclic function block.



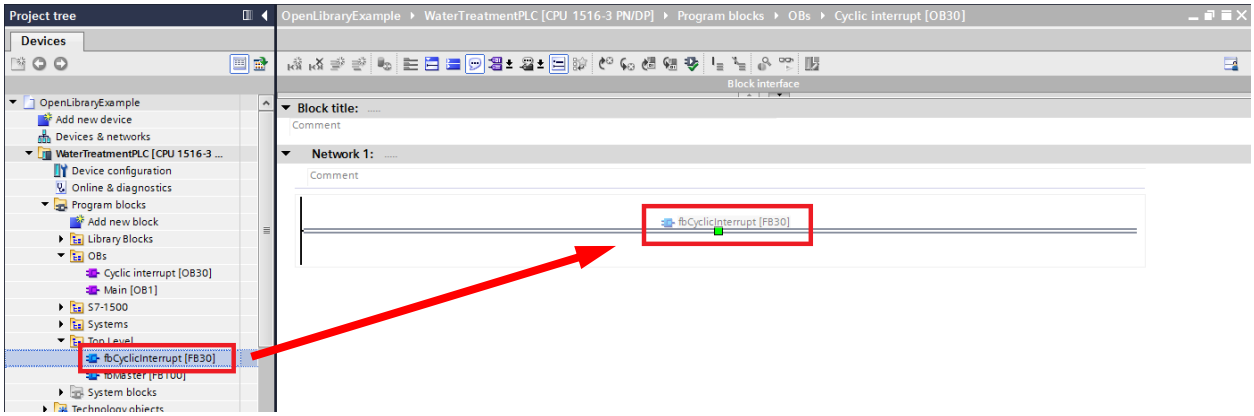
11. Create an instance data block to accompany the cyclic function block.



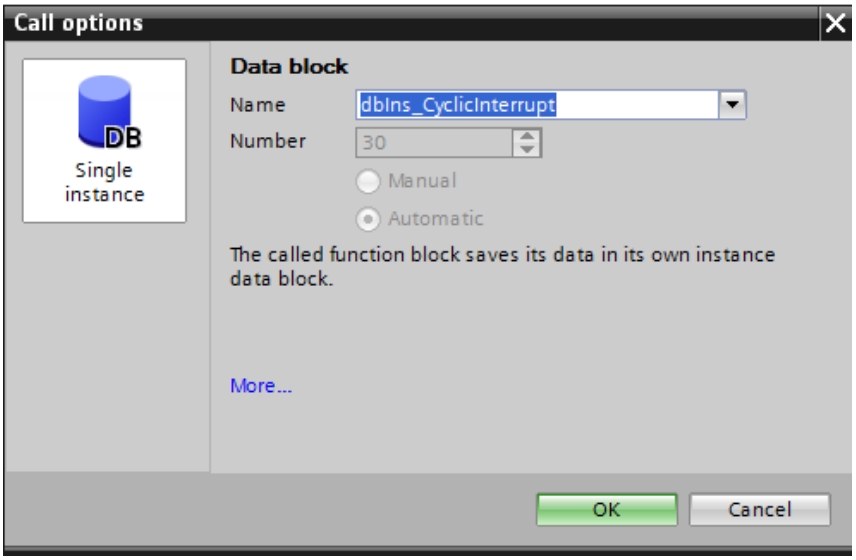
12. Name the instance data block and give it the type of your cyclic function block.

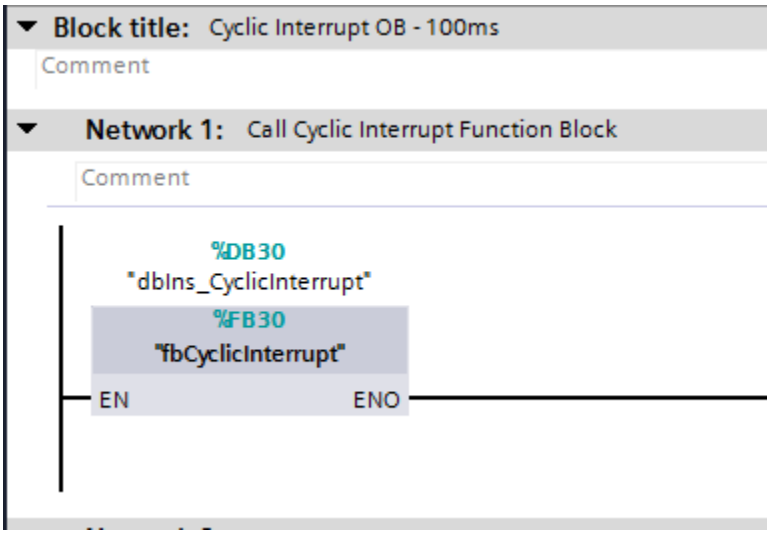


13. Inside of the cyclic interrupt OB, drag in an instance of the cyclic FB you created.

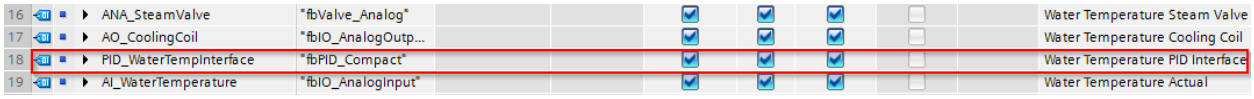


14. Assign the instance data block as a single instance.

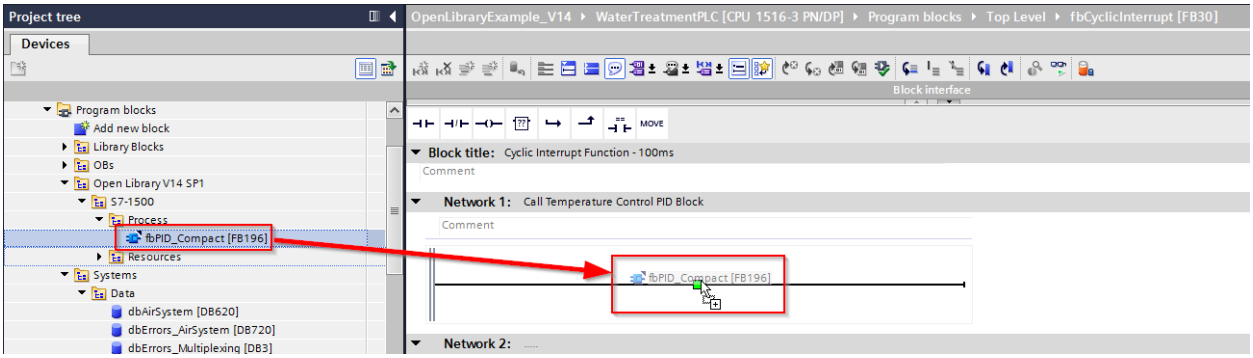




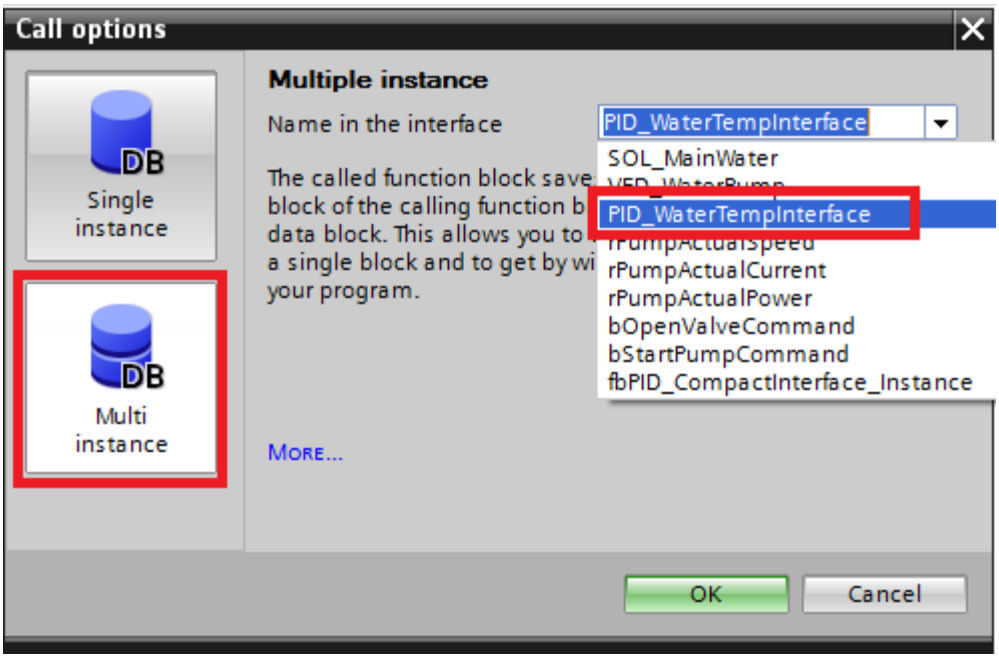
15. Add a multiple instance static memory variable for the fbPID_Compact block in the fbCyclicInterrupt block.



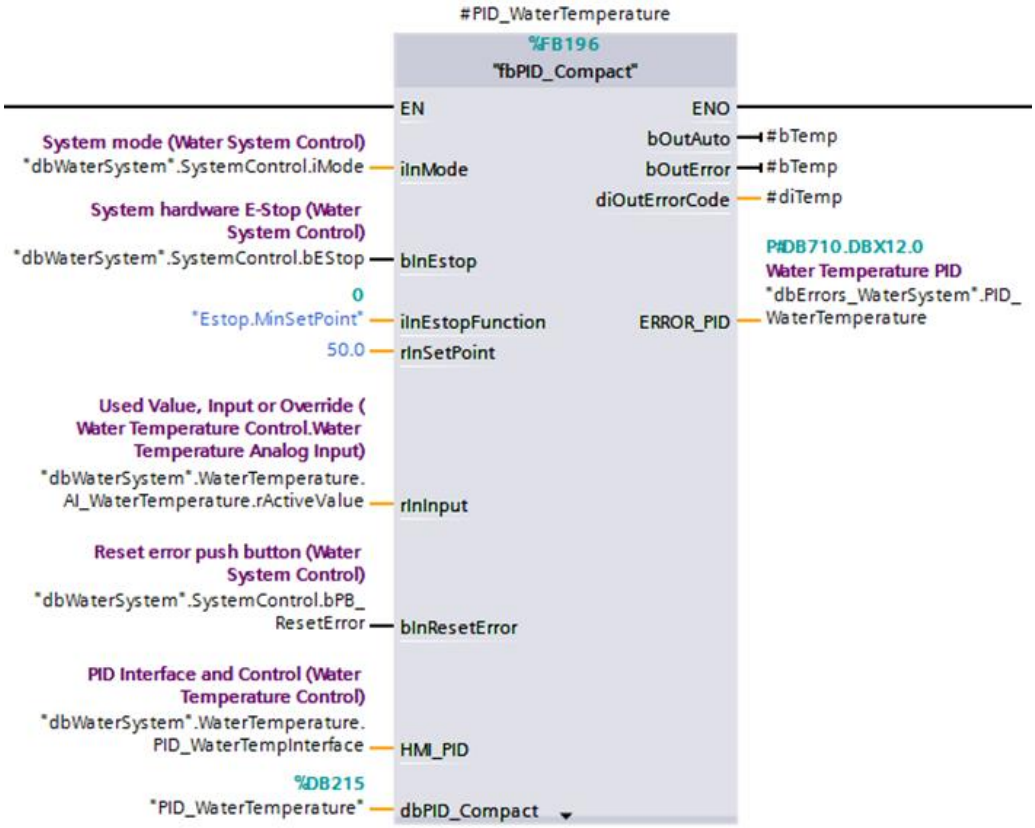
16. Drag an instance of fbPID_Compact into a network in fbCyclicInterrupt.



17. When the Call options dialog appears, choose the static memory multiple instance that you just created in the block interface.



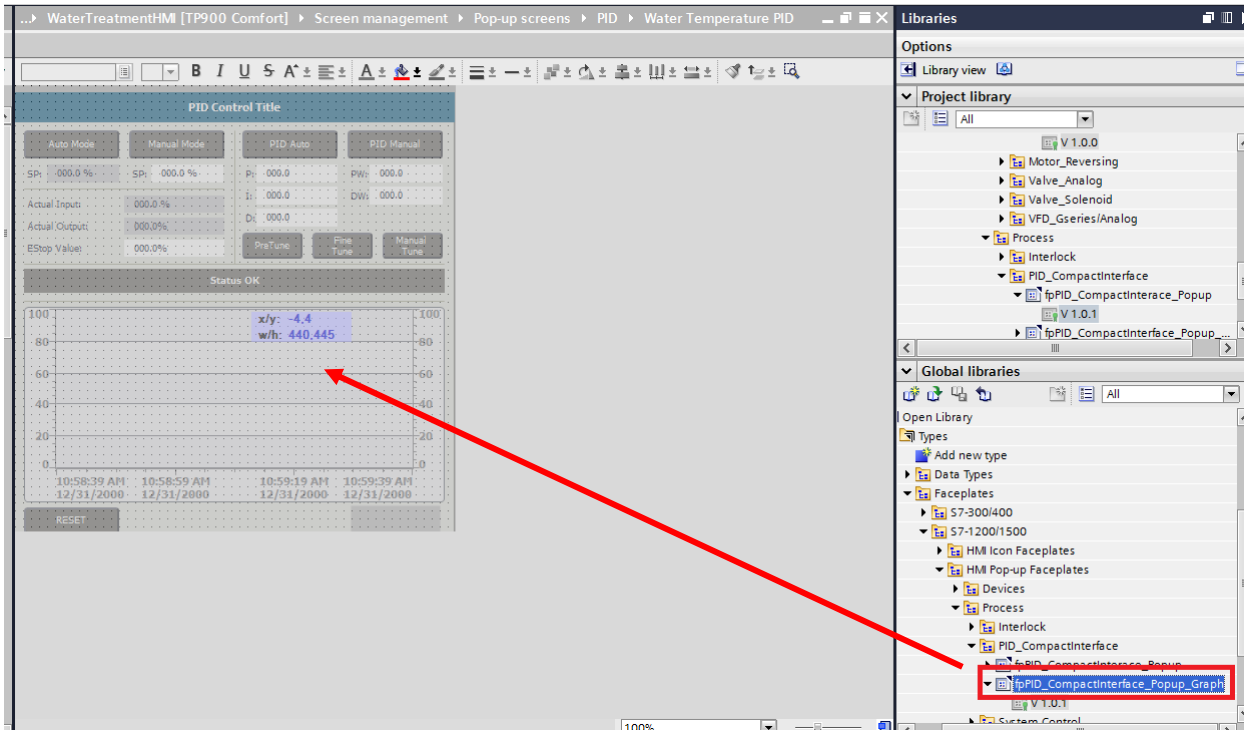
18. Fill in the interface for your fbPID_Compact instance. Notice the reference to the PID_Compact technology data block as well as the HMI and Errors data block structures.


#PID_WaterTemperature
%FB196
"fbPID_Compact"
EN
ENO
System mode (Water System Control)
"dbWaterSystem".SystemControl.iMode
iInMode
bOutAuto
#bTemp
bOutError
#bTemp
diOutErrorCode
#diTemp
System hardware E-Stop (Water System Control)
"dbWaterSystem".SystemControl.bEStop
bInEstop
P#DB710.DBX12.0
Water Temperature PID
"dbErrors_WaterSystem".PID_WaterTemperature
0
"Estop.MinSetPoint"
iInEstopFunction
ERROR_PID
50.0
rInSetPoint
Used Value, Input or Override (Water Temperature Control.Water Temperature Analog Input)
"dbWaterSystem".WaterTemperature.AI_WaterTemperature.rActiveValue
rInInput
Reset error push button (Water System Control)
"dbWaterSystem".SystemControl.bPB_ResetError
bInResetError
PID Interface and Control (Water Temperature Control)
"dbWaterSystem".WaterTemperature.PID_WaterTempInterface
HMI_PID
%DB215
"PID_WaterTemperature"
dbPID_Compact

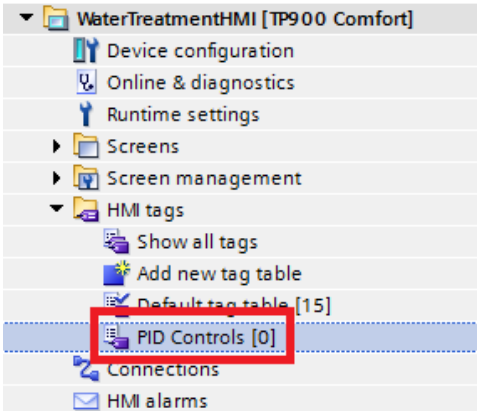
5.3. HMI

This section walks through the required steps to add the HMI pop-ups for the PID Interface Block.

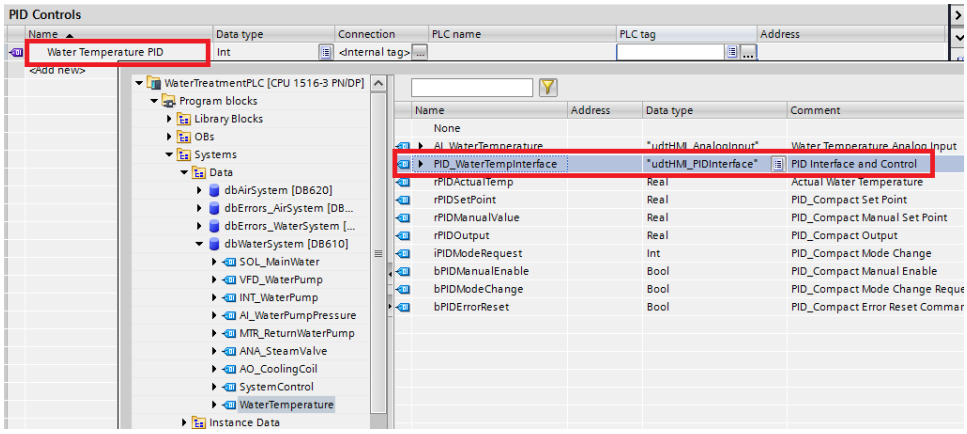
1. Create a new Pop-up screen for the PID Faceplate.
2. Pull the fpPID_CompactInterface_Popup_Graph into the new Pop-up screen.



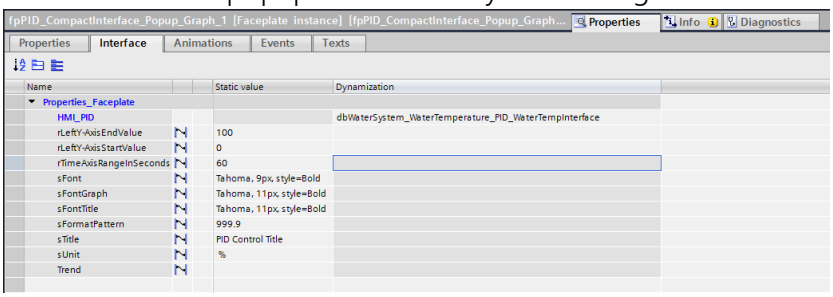
3. Add a new HMI tag table called PID Controls.



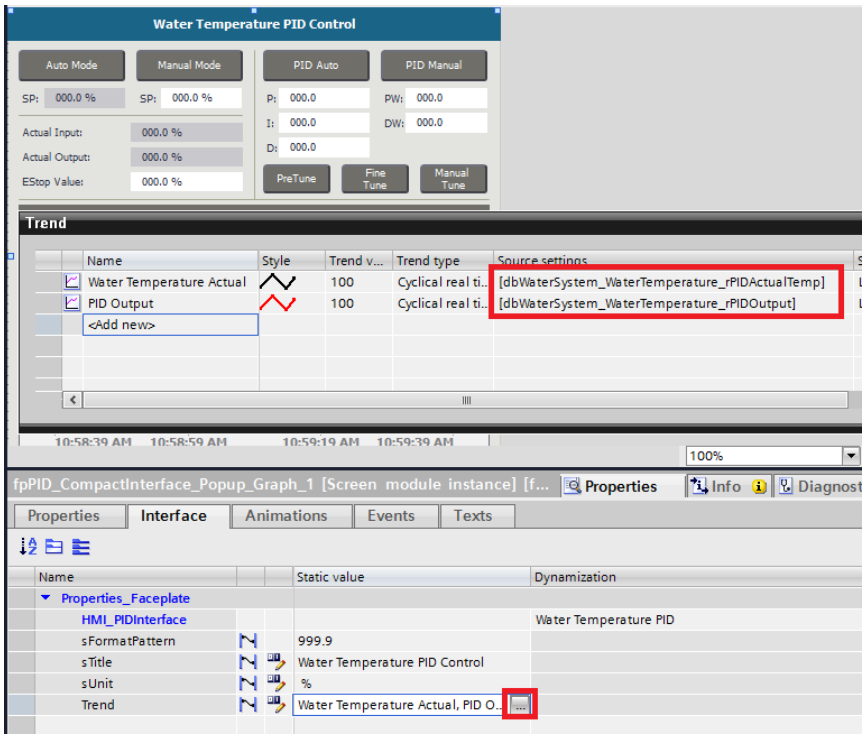
4. Create the Water Temperature PID tag in the new table and map it to "dbWaterSystem".WaterTemperature.PID_WaterTemperature.



5. Fill in the interface of the popup with the newly created tag as well as static values.



6. Lastly, configure the Trend view with the tags you would like to trend.


Water Temperature PID Control
Auto Mode
Manual Mode
PID Auto
PID Manual
SP: 000.0 %
SP: 000.0 %
P: 000.0
PW: 000.0
I: 000.0
DW: 000.0
D: 000.0
Actual Input: 000.0 %
Actual Output: 000.0 %
EStop Value: 000.0 %
PreTune
Fine Tune
Manual Tune
Trend
Name
Style
Trend v...
Trend type
Source settings
Water Temperature Actual
100
Cyclical real ti...
[dbWaterSystem_WaterTemperature_rPIDActualTemp]
PID Output
100
Cyclical real ti...
[dbWaterSystem_WaterTemperature_rPIDOutput]
<Add new>
10:58:39 AM
10:58:59 AM
10:59:19 AM
10:59:39 AM
100%
fpPID_CompactInterface_Popup_Graph_1 [Screen module instance] [f...
Properties
Info
Diagnost
Properties
Interface
Animations
Events
Texts
Name
Static value
Dynamization
Properties_Faceplate
HMI_PIDInterface
Water Temperature PID
sFormatPattern
999.9
sTitle
Water Temperature PID Control
sUnit
%
Trend
Water Temperature Actual, PID O...