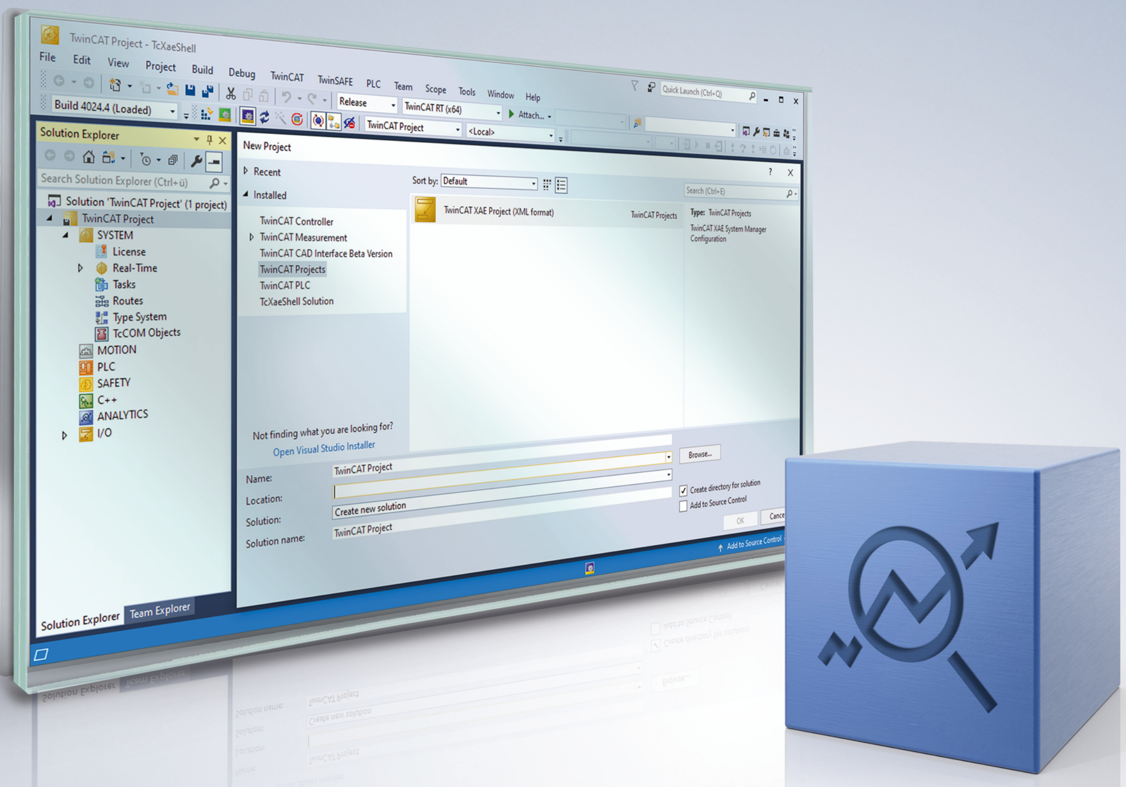
BECKHOFF New Automation Technology

Manual | EN

TE3500

TwinCAT 3 | Analytics Workbench



2025-06-24 | Version: 1.12.0

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 For your safety 5
1.3 Notes on information security 7
2 Overview 8
3 Installation 11
3.1 System requirements 11
3.2 Installation and licensing 11
3.3 Licensing 14
4 Analytics Workflow - First Steps 17
4.1 Recording data from the machine 17
4.2 Communication 20
4.3 Historicize data 21
4.4 Analyse data 28
4.5 24h Analytics application 33
5 Technical introduction 42
5.1 Basic concept 42
6 Configuration 44
6.1 Data Source 44
6.1.1 Wizard 46
6.2 Data Scout 52
6.2.1 Data Track Editor 52
6.3 Editor Basics 65
6.3.1 Networks 65
6.3.2 Smart Link 69
6.3.3 Enable Disable 73
6.3.4 Static Values 75
6.3.5 Global parameters 76
6.4 Algorithms 77
6.4.1 Analytics - Base 79
6.4.2 Analytics - Classification 133
6.4.3 Analytics - Clustering 158
6.4.4 Analytics - Compare 167
6.4.5 Analytics - Energy 191
6.4.6 Analytics - Math 196
6.4.7 Analytics - Preprocessing 210
6.4.8 Analytics - Reporting 212
6.4.9 Analytics - Specific 216
6.4.10 Analytics - Statistics 227
6.4.11 Analytics - Training Base 251
6.4.12 Analytics - Visualization Only 253
6.4.13 Analytics - XY Path Analysis 260
6.4.14 Analytics - extension of the algorithms 270

6.5 Interaction with Scope ... 344
6.5.1 Scope configuration in the network template ... 348
6.6 Working with Historical Data ... 351
6.7 Runtime deployment ... 353
6.7.1 Algorithm properties ... 360
6.7.2 PLC Code ... 362
6.8 HMI One-Click Dashboard ... 419
6.8.1 Manage dashboard structure and content in Analytics project ... 422
6.8.2 Use customized and own controls ... 432
6.8.3 Configure user management and access rights ... 451
6.8.4 Dashboard Configuration ... 454
6.8.5 Modifying a generated dashboard in HMI engineering ... 461
6.8.6 Configuration of the dashboard at runtime in the web browser ... 461
6.8.7 Creating charts in the web browser ... 468
6.8.8 Switching multiple machines in the HMI Dashboard ... 493
6.8.9 Integration of a language switch ... 499
6.8.10 Using the interactive functions in standard TwinCAT HMI projects ... 502
6.9 Reporting Integration ... 504
6.9.1 Basic concept ... 504
6.9.2 Analytics Reporting ... 505
6.9.3 TwinCAT Reporting Server ... 513
6.10 Comparison of analysis results ... 524
6.10.1 Loading and saving project configurations ... 526
7 API ... 528
7.1 Data Exchange API ... 528
7.1.1 .NET ... 528
7.1.2 Python ... 534
8 Samples ... 537
8.1 Pick-and-place application ... 537
8.2 Pick-and-place application: DenStream ... 545
8.3 Pick-and-place application: correlation ... 547
9 Appendix ... 551
9.1 FAQ - frequently asked questions and answers ... 551

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
The documentation and the following notes and explanations must be complied with when installing and commissioning the components.
The trained specialists must always use the current valid documentation.

The trained specialists must ensure that the application and use of the products described is in line with all safety requirements, including all relevant laws, regulations, guidelines, and standards.

Disclaimer

The documentation has been compiled with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without notice.
Claims to modify products that have already been supplied may not be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

Beckhoff®, ATRO® , EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar®, and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of the designations or trademarks contained in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Copyright

Third-party trademarks

Trademarks of third parties may be used in this documentation. You can find the trademark notices here: https://www.beckhoff.com/trademarks.

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER

Hazard with high risk of death or serious injury.

⚠ WARNING

Hazard with medium risk of death or serious injury.

⚠ CAUTION

There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE

The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Overview

Machine statistics

Current part number	2,154
Total numbers of parts	12,144
Parts per hour	5.34
Parts per day	128.16
Parts per week	897.12
Total parts of the last week	865.42
Number of damaged parts	17
Total operation time	1482,5 h

The TwinCAT 3 Analytics Workbench is a TwinCAT 3 engineering product for creating continuous data analyses from various decentralized machine controllers. The configuration of the workbench is integrated into Microsoft Visual Studio® and is designed as a graphical user interface. Many algorithms, such as cycle time monitoring, life count, lifetime and minimum/maximum/mean, are available in a toolbox for configuring the analysis.

For simple visualization of the signal curves, the TwinCAT 3 Analytics Workbench contains the TwinCAT 3 Scope View Professional TE1300: the user can drag and drop the analysis results from the Analytics Configurator into the charting tool to highlight significant points in the data stream. Such markings can be simple minima and maxima, counter values or also the results of a logical operator that logically combines results from the machine controller so that they can be found in the data stream. This allows correlation with other signals in the Scope View to the exact cycle.

The MQTT input data is selected via the TwinCAT Target Browser, where live data and, via the TF3520 TwinCAT 3 Analytics Storage Provider, historical data are also available. Once the created analysis is complete and tested in the graphical editor, this configuration can be converted into readable PLC code with an associated HTML5 dashboard in just a few clicks. The automatically generated PLC code and dashboard can be downloaded directly to a device with TF3550 TwinCAT 3 Analytics Runtime and run there 24/7 in parallel with the actual production machines and provide analysis results. The display via the TwinCAT 3 HMI shows analysis results for machine operators, production managers and/or machine builders, for example.

Functionality

Compared to the Analytics Service Tool, the Analytics Workbench offers complementary features including code and dashboard generation. All functions of the Service Tool are also included in the Workbench.

Components

- Analytics configurator
- Base Analytics algorithms
- Analytics PLC library
- Analytics Storage Provider Recorder incl. PLC library
- TwinCAT Scope (TE1300 and TF3300)
- IoT connectivity
- HMI engineering (TE2000)

Features	TE3500 Analytics Workbench		TE3520 Analytics Service Tool	
	7-Days-Trial	Full License	7-Days-Trial	Full License
General:				
Analysis configurator	✓	✓	✓	✓
Analysis channels	max 5	unlimited	max 5	unlimited
Analysis moduls/ algorithm	max 3	unlimited	max 3	unlimited
Long time records >1h	✗	✓	✗	✓
Interaction with Scope View	✓	✓	✓	✓
Storage Provider Recorder	✓	✓	✓	✓
MQTT	✓ (max 5)	✓	✓ (max 5)	✓
Analytics File	✓ (max 5)	✓	✓ (max 5)	✓
ADS	✓ (no auto deployment)	✓ (no auto deployment)	✓	✓
Data export tool	✓	✓	✓	✓
Basic data export formats	✓ (max 5)	✓	✓ (max 5)	✓
Extended data export formats	✗	✓	✗	✓
Analytics Data Scout:				
Load data	✓	✓	✓	✓
Toolbar snipping buttons	✗ *	✓	✗ *	✓
Export data to Analytics File	✓	✓	✓	✓
Deploy Runtime:				
Deploy Wizard	✓	✓	/	/
Add PLC Code to existing Sln	✗	✓	/	/
Merge PLC Code	✗	✓	/	/
Auto Create Bootproject	✗	✓	/	/
Auto Activate Runtime	✗	✓	/	/
Stream Results	✗	✓	/	/
HMI Dashboard	✓ (defaults)	✓	/	/
Reset Algorithm in PLC Code	✗	✓	/	/
Reporting Integration:				
24/7-Reporting with Reporting Algorithm	✓	✓	/	/

Features	TE3500 Analytics Workbench		TE3520 Analytics Service Tool	
On-Demand-Reporting				
Adding Additional Information	✓ (max 1)	✓	✓ (max 1)	✓
Select specific Analytics Information	✗	✓	✗	✓
✓	Full support			
✗	No support			
/	No feature of this product			
*	By existing TE1300 Scope Professional full support			

3 Installation

3.1 System requirements

The following system requirements must be fulfilled for proper function of TwinCAT Analytics.

Supported operating systems

Windows 10

TwinCAT

Minimum is TwinCAT 3.1 Build 4022.29 for engineering with TwinCAT Analytics Service Tool and Workbench.

.NET Framework

Engineering requires a .NET Framework 4.7.2.

Visual Studio development environment

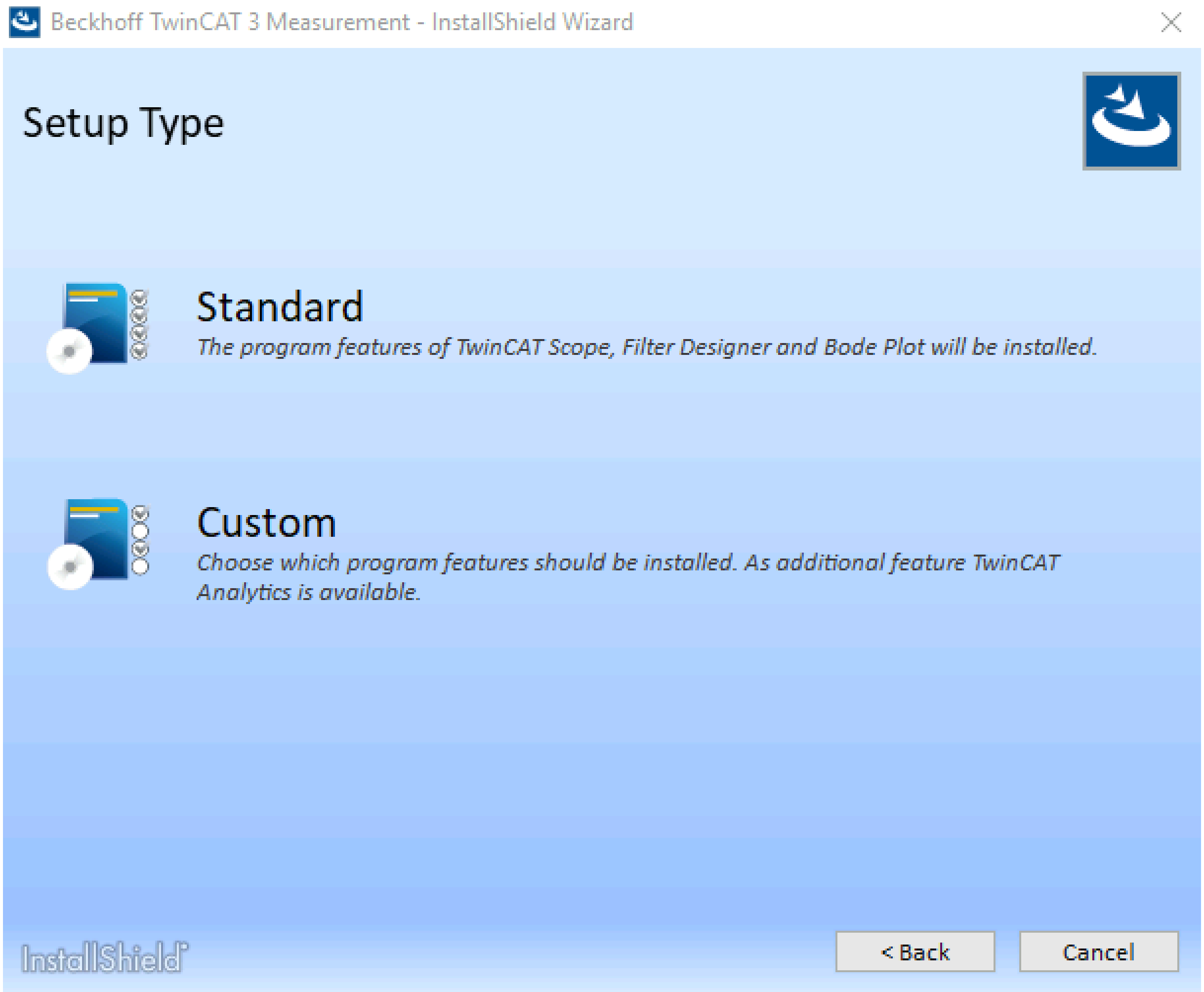
- Visual Studio® 2015
- Visual Studio® 2017
- Visual Studio® 2019
- TwinCAT XAE Shell

In general, using the Visual Studio® Shell is sufficient. If you select the "Full" setup, the TwinCAT XAE Shell is installed automatically. The "Update" setup only provides an update of the Analytics sources and not a Visual Studio® Shell.

3.2 Installation and licensing

The TwinCAT Analytics setup is part of the TwinCAT Measurement Suite setup. You have the option of choosing between two setup variants:
The TC3-Measurement-Full Setup includes the TwinCAT Measurement components and a Visual Studio® environment.
The TC3 Measurement Update Setup, on the other hand, only includes the measurement components.

After confirming the terms in the license agreement, you can choose between Standard and Custom. By default, all measurement components including TwinCAT Analytics Engineering are automatically installed.



Beckhoff TwinCAT 3 Measurement - InstallShield Wizard

Setup Type

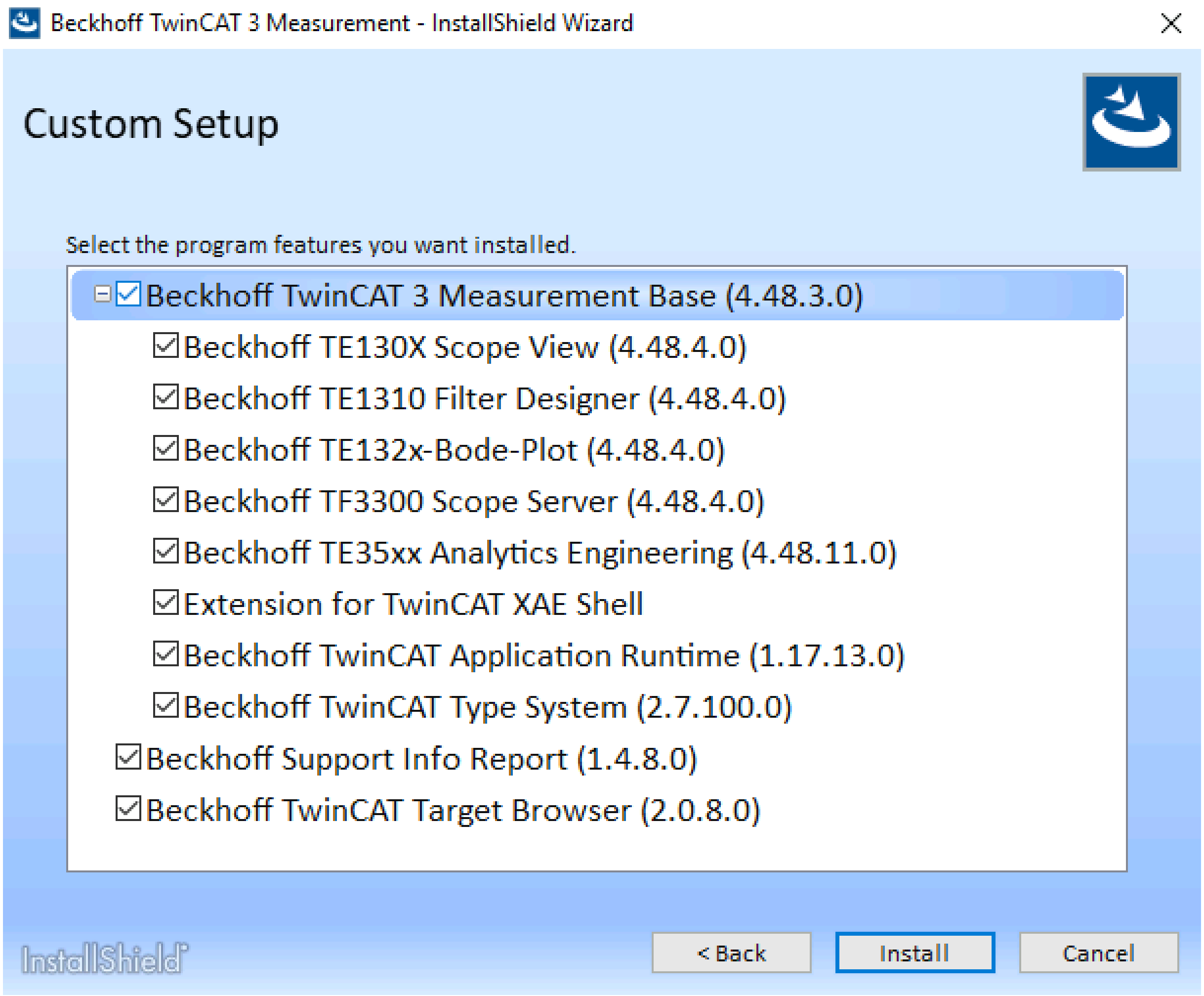
Standard

The program features of TwinCAT Scope, Filter Designer and Bode Plot will be installed.

Custom

Choose which program features should be installed. As additional feature TwinCAT Analytics is available.

With Custom, you can deselect individual components so that they are not installed on your system.



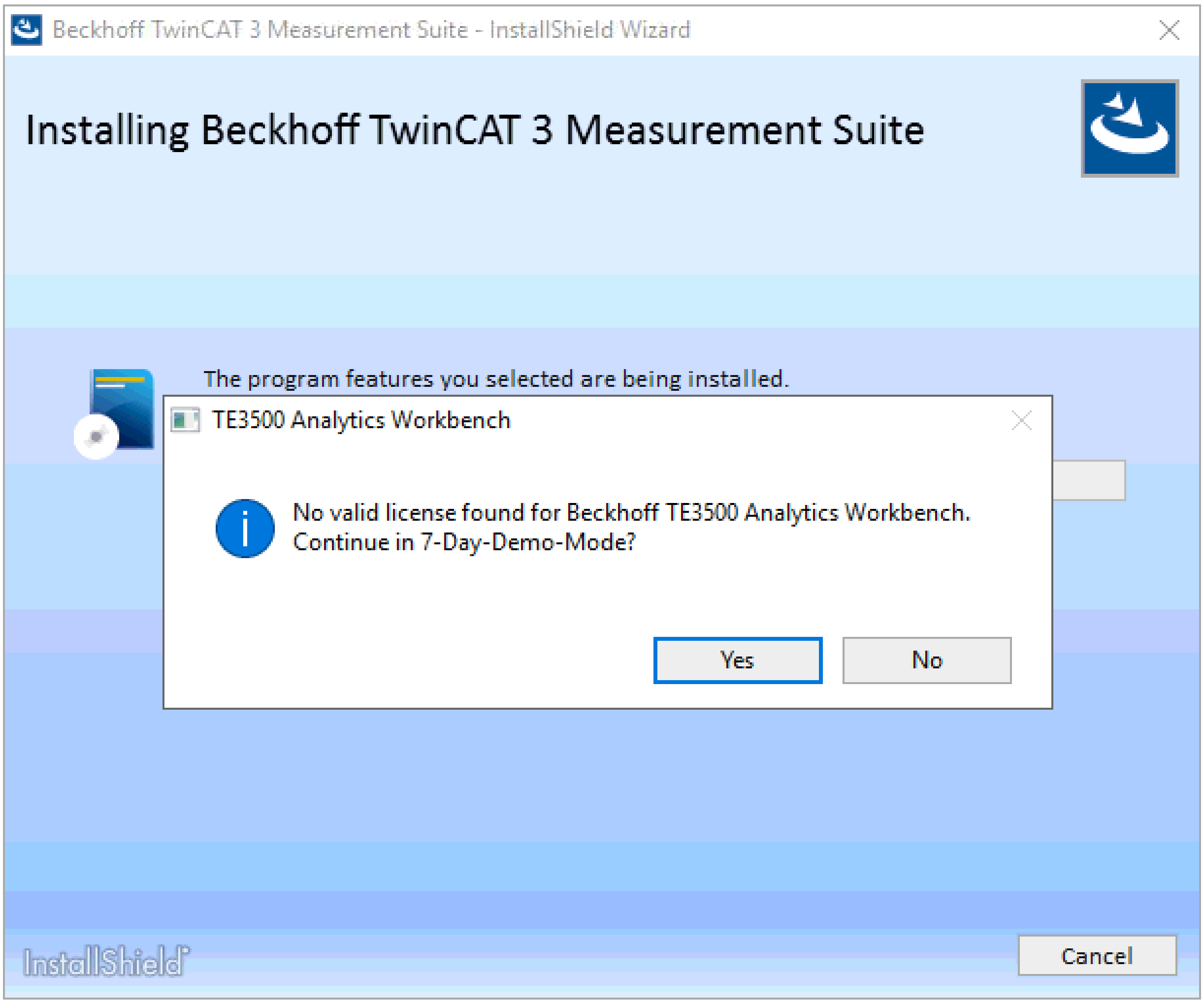
Beckhoff TwinCAT 3 Measurement - InstallShield Wizard

Custom Setup

Select the program features you want installed.

- ⊟☑ Beckhoff TwinCAT 3 Measurement Base (4.48.3.0)
  - ☑ Beckhoff TE130X Scope View (4.48.4.0)
  - ☑ Beckhoff TE1310 Filter Designer (4.48.4.0)
  - ☑ Beckhoff TE132x-Bode-Plot (4.48.4.0)
  - ☑ Beckhoff TF3300 Scope Server (4.48.4.0)
  - ☑ Beckhoff TE35xx Analytics Engineering (4.48.11.0)
  - ☑ Extension for TwinCAT XAE Shell
  - ☑ Beckhoff TwinCAT Application Runtime (1.17.13.0)
  - ☑ Beckhoff TwinCAT Type System (2.7.100.0)
- ☑ Beckhoff Support Info Report (1.4.8.0)
- ☑ Beckhoff TwinCAT Target Browser (2.0.8.0)

Analytics Setup checks whether your system has the required Analytics Engineering licenses during the process. If not, a demo license can be activated. This demo license can be extended as often as you like. However, after the installation in demo mode, no further functional updates can be performed on this system through newer setups. In order to be able to import these updates, a license must first be purchased. An overview of the functional limitations in the demo version can be found here [▶ 9].



Beckhoff TwinCAT 3 Measurement Suite - InstallShield Wizard
Installing Beckhoff TwinCAT 3 Measurement Suite
The program features you selected are being installed.
TE3500 Analytics Workbench
Yes
No
InstallShield
Cancel

No valid license found for Beckhoff TE3500 Analytics Workbench. Continue in 7-Day-Demo-Mode?

The licenses of the Analytics Engineering Tools TE3500 Analytics Workbench and TE3520 Analytics Service Tool are always associated with a maintenance license. After the initial purchase, the maintenance license is valid for 12 months. Functional updates of the software can be executed during this period. After 12 months, the software can still be used without any restrictions. However, new functions can no longer be imported without extending the maintenance license. The installation indicates this with a corresponding message. An extension of the maintenance license can be performed at any time via the products TE3501 and TE3521.

Please contact your Beckhoff sales employee for information about future functions of newer versions.

Setup requires license

Updates of TwinCAT Analytics engineering tools can only be performed with a valid maintenance license!

TwinCAT 3 licenses for non-Beckhoff devices

If you use an IPC from a manufacturer other than Beckhoff (TwinCAT 3 plattform level >= 90), aTwinCAT 3 licencse dongle is highly recommended, if not a prerequisite for successful licensing of TwinCAT Analytics!

TwinCAT Analytics Workflow

To use the complete Analytics Workflow you need the TE2000 HMI engineering setup. Today it is not integrated into the Analytics Workbench setup. This means you must install it individually. It will be integrated in one of the next setup versions. So, this step will be not necessary anymore.

3.3 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

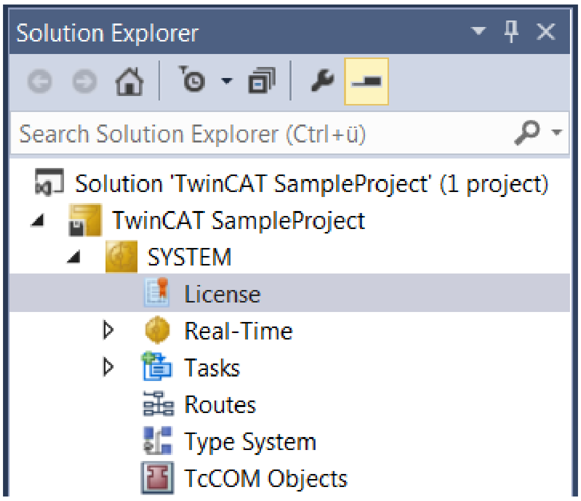
Licensing the full version of a TwinCAT 3 Function

A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "TwinCAT 3 Licensing".

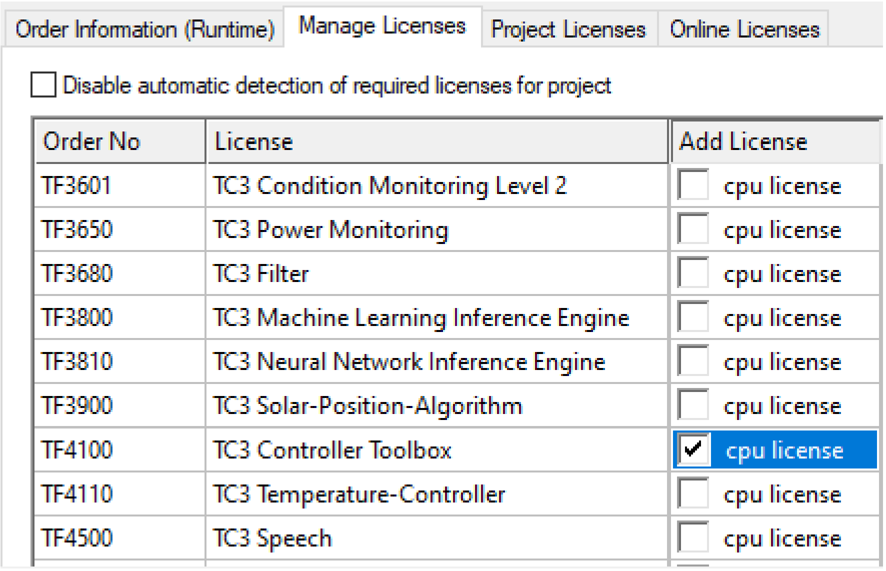
Licensing the 7-day test version of a TwinCAT 3 Function

A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the toolbar.
   - ⇨ The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.
4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



   - ⇨ The TwinCAT 3 license manager opens.
5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF4100 TC3 Controller Toolbox").

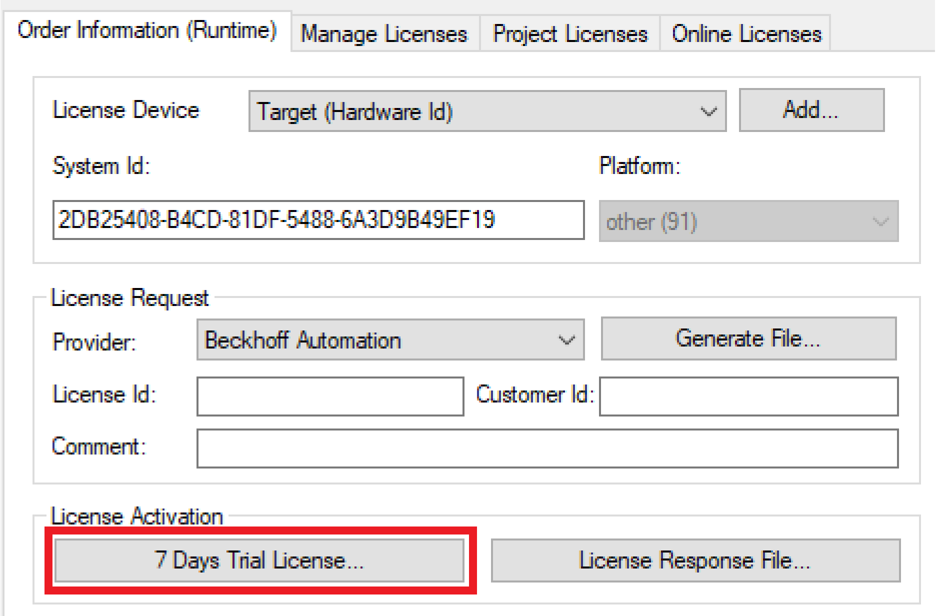


☐ Disable automatic detection of required licenses for project

Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	☐ cpu license
TF3650	TC3 Power Monitoring	☐ cpu license
TF3680	TC3 Filter	☐ cpu license
TF3800	TC3 Machine Learning Inference Engine	☐ cpu license
TF3810	TC3 Neural Network Inference Engine	☐ cpu license
TF3900	TC3 Solar-Position-Algorithm	☐ cpu license
TF4100	TC3 Controller Toolbox	☑ cpu license
TF4110	TC3 Temperature-Controller	☐ cpu license
TF4500	TC3 Speech	☐ cpu license

6. Open the **Order Information (Runtime)** tab.
   - ⇨ In the tabular overview of licenses, the previously selected license is displayed with the status “missing”**.**

7. Click **7-Day Trial License...** to activate the 7-day trial license.



License Device: Target (Hardware Id)

System Id: 2DB25408-B4CD-81DF-5488-6A3D9B49EF19

Platform: other (91)

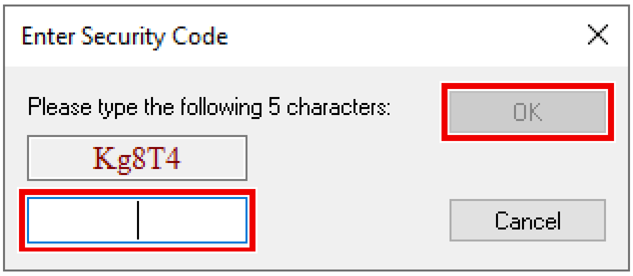
Provider: Beckhoff Automation

License Id:

Customer Id:

Comment:

⇨ A dialog box opens, prompting you to enter the security code displayed in the dialog.



Please type the following 5 characters:

Kg8T4

8. Enter the code exactly as it is displayed and confirm the entry.
9. Confirm the subsequent dialog, which indicates the successful activation.

   ⇨ In the tabular overview of licenses, the license status now indicates the expiry date of the license.
10. Restart the TwinCAT system.

⇨ The 7-day trial version is enabled.

4 Analytics Workflow - First Steps

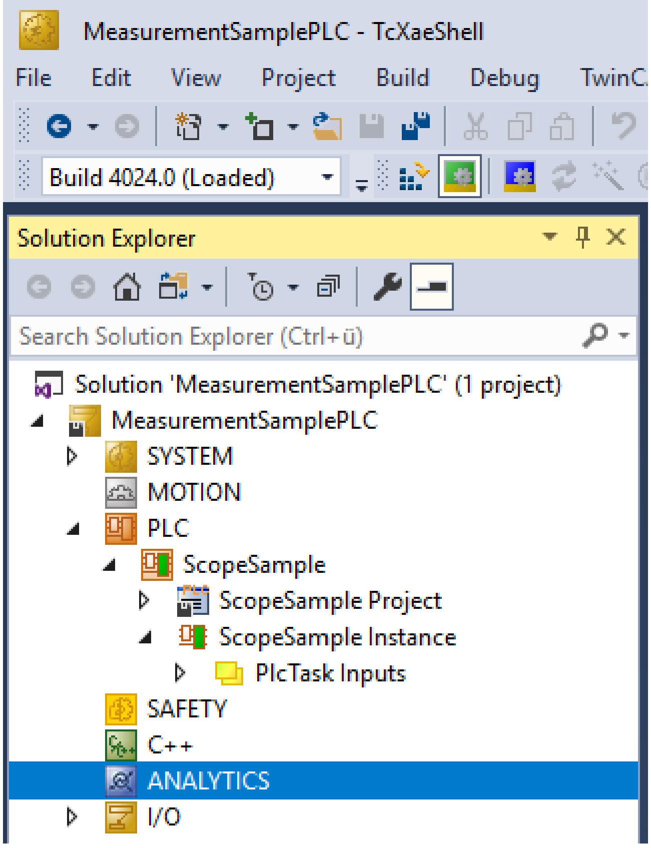
This step by step documentation presents the complete TwinCAT Analytics workflow. From the data acquisition over the communication and historizing up to the evaluation and analysis of the data and to the presentation of the data in web-based dashboard.

4.1 Recording data from the machine

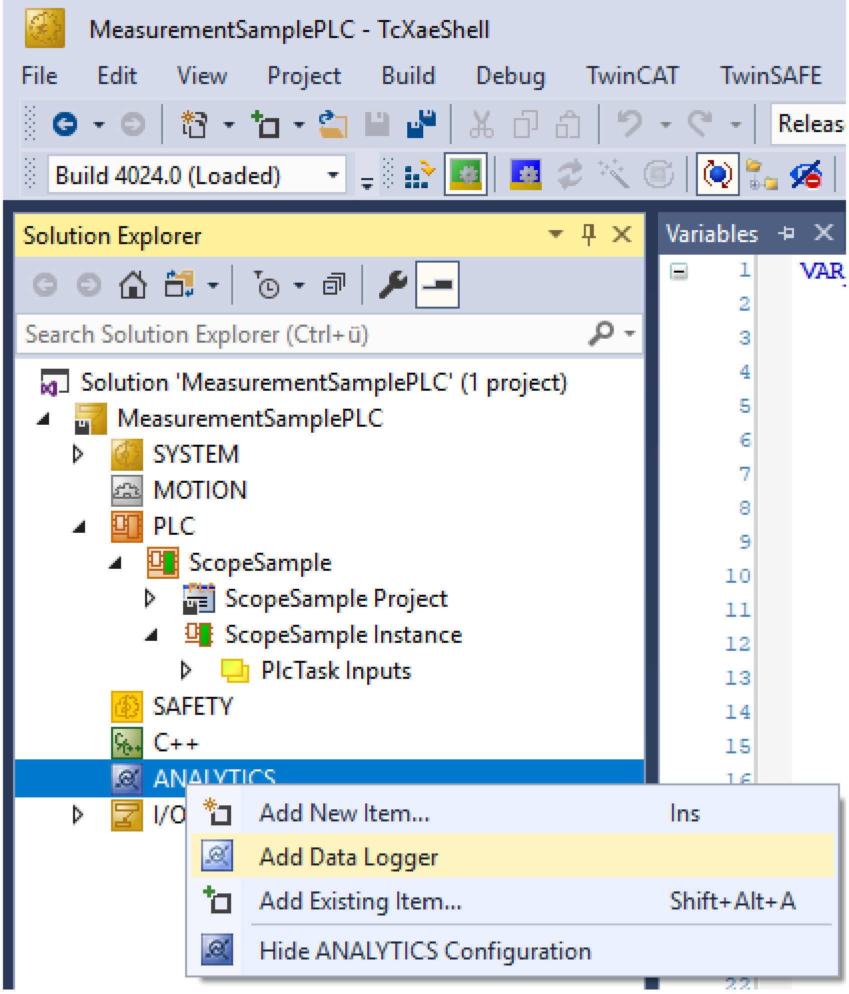
On the machine side is the Analytics Logger the recorder of process data from the machine image, PLC, NC and so on. The Logger is working in the real-time context of TwinCAT.

The TwinCAT Analytics Logger is installed with TwinCAT XAE and XAR. The Logger can act as MQTT Client to communicate the recorded data to a native MQTT Message Broker or store the data in the same data format in a local binary file. By the usage as MQTT Client the Logger is able to bypass short disconnects to the Message Broker with a ring buffer functionality. You can configure a ring buffer as well for the local binary file storage.

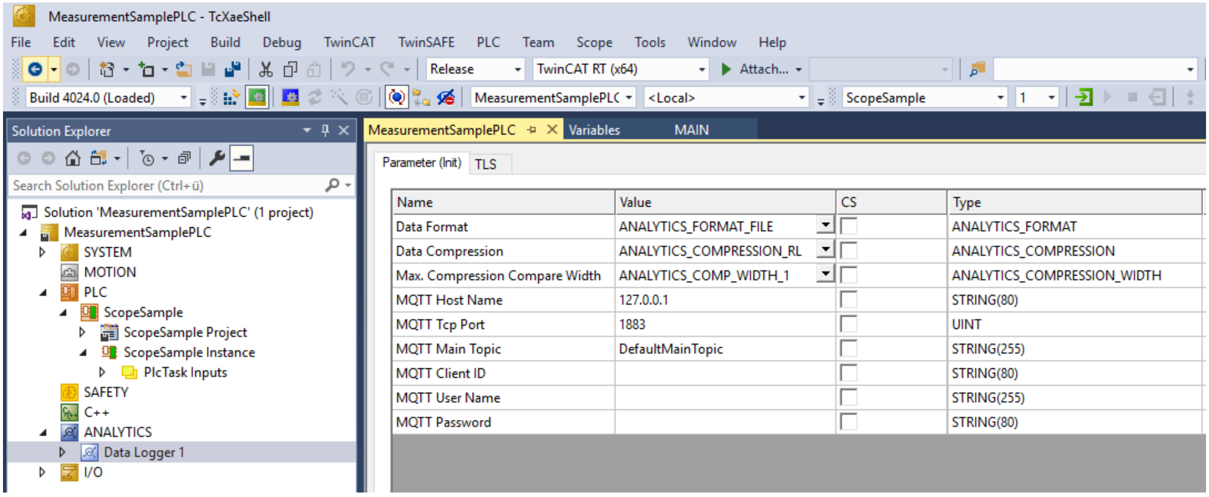
- To configure the Analytics Logger you have to navigate in your existing TwinCAT Project to the Analytics tree node



- Right click on this node and click on “Add Data Logger” to add one new instance to your configuration



- For configuring the base settings, please double click on the new tree item



Name	Value	CS	Type
Data Format	ANALYTICS_FORMAT_FILE	□	ANALYTICS_FORMAT
Data Compression	ANALYTICS_COMPRESSION_RL	□	ANALYTICS_COMPRESSION
Max. Compression Compare Width	ANALYTICS_COMP_WIDTH_1	□	ANALYTICS_COMPRESSION_WIDTH
MQTT Host Name	127.0.0.1	□	STRING(80)
MQTT Tcp Port	1883	□	UINT
MQTT Main Topic	DefaultMainTopic	□	STRING(255)
MQTT Client ID		□	STRING(80)
MQTT User Name		□	STRING(255)
MQTT Password		□	STRING(80)

You can make your specific Analytics Logger settings

-Data Format: Binary file or MQTT stream

-FILE format: Analytics Logger stores the data in local binary files and all other settings are not necessary anymore. The files will be stored in C:\TwinCAT\3.1\Boot\Analytics.

-BINARY: Data will be sent to the configured MQTT Message Broker. You can have multiple Logger in one TwinCAT project to communicate data to different MQTT Message Broker.

-Data Compression: on (default) or off

-Max Compression: mode of the compression

-MQTT host name

-MQTT Tcp port

-MQTT main topic for own hierarchical levels to keep the identification easy

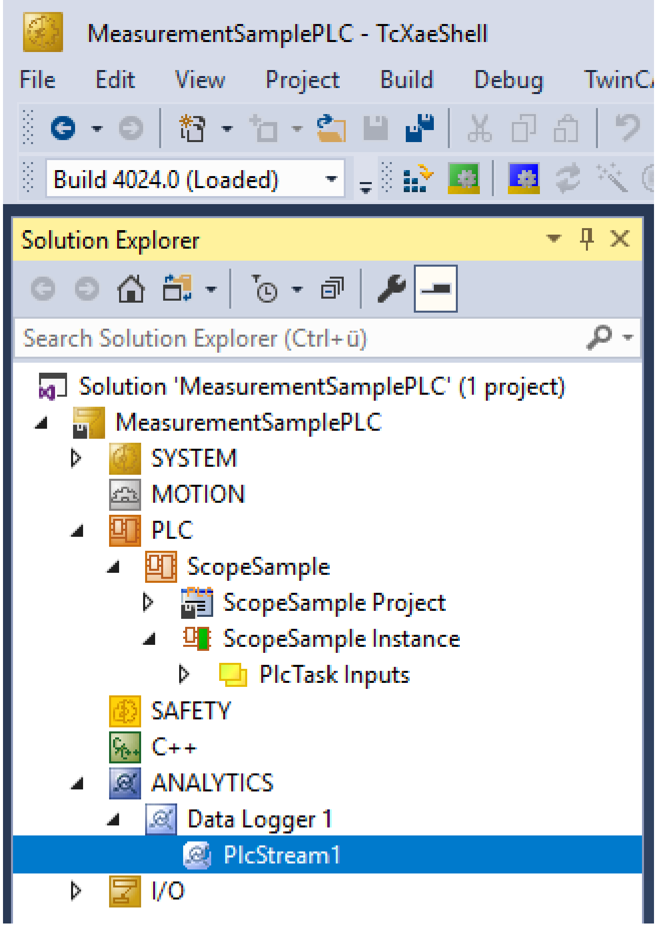
-MQTT Client ID should be unique in the network

-MQTT username

-MQTT password to make authentication at the message broker

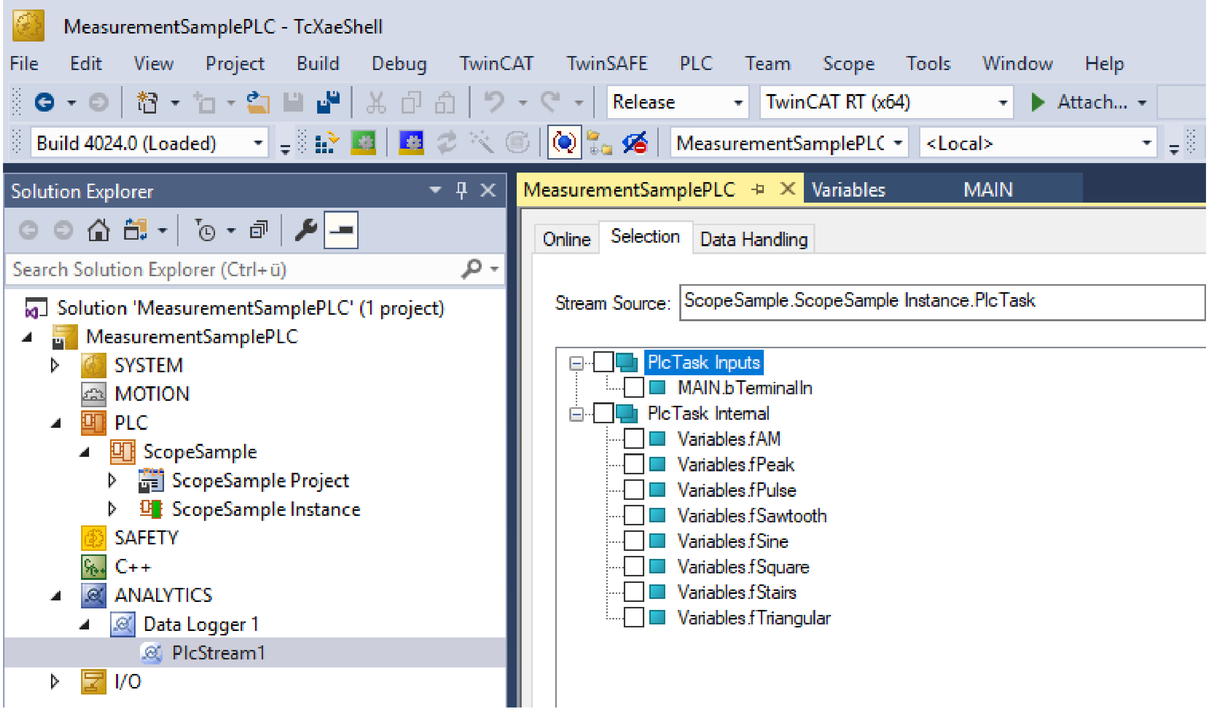
-At the TLS (Transport Layer Security) tab, security settings can be configured. TLS is a secure communication channel between client and server. By the usage of certificates, the TCP port 8883 is exclusively reserved for MQTT over TLS. Analytics Logger is supporting the modes CA Certificates, CA Certificates & Client Certificate and Preshared Key (PSK) mode.

- If variables in your PLC application are marked in the declaration with the attribute {attribute 'TcAnalytics'} they will be shown automatically as a stream below the Data Logger tree node.



An additional device stream will be shown if your configuration provides an EtherCAT Process Image.

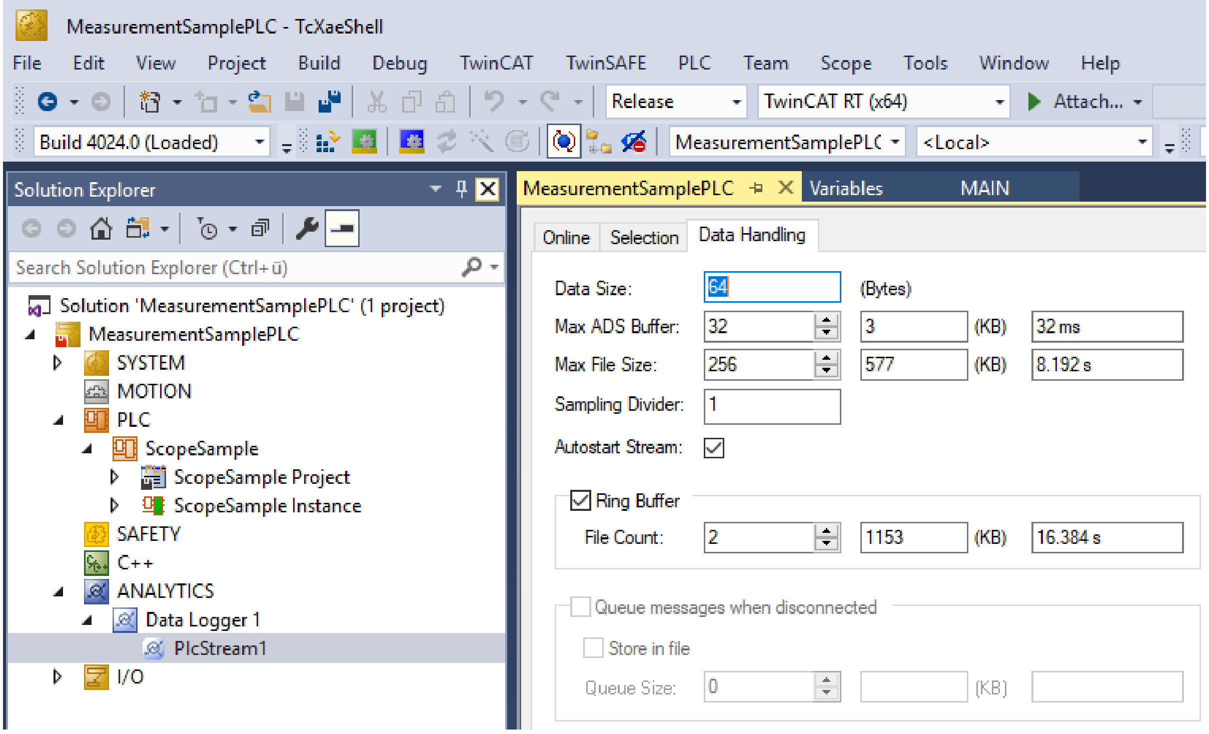
- In the stream a Selection tab is available to choose the variables that should be recorded



Stream Source: ScopeSample.ScopeSample Instance.PlcTask

- □ PlcTask Inputs
  - □ MAIN.bTerminalIn
- □ PlcTask Internal
  - □ Variables.fAM
  - □ Variables.fPeak
  - □ Variables.fPulse
  - □ Variables.fSawtooth
  - □ Variables.fSine
  - □ Variables.fSquare
  - □ Variables.fStairs
  - □ Variables.fTriangular

- Finally it is possible to change the package size for the frames or to configure the ring buffer for disconnects and file in the Data Handling tab.



Data Size: 64 (Bytes)

Max ADS Buffer: 32 | 3 (KB) | 32 ms

Max File Size: 256 | 577 (KB) | 8.192 s

Sampling Divider: 1

Autostart Stream: ☑

☑ Ring Buffer

File Count: 2 | 1153 (KB) | 16.384 s

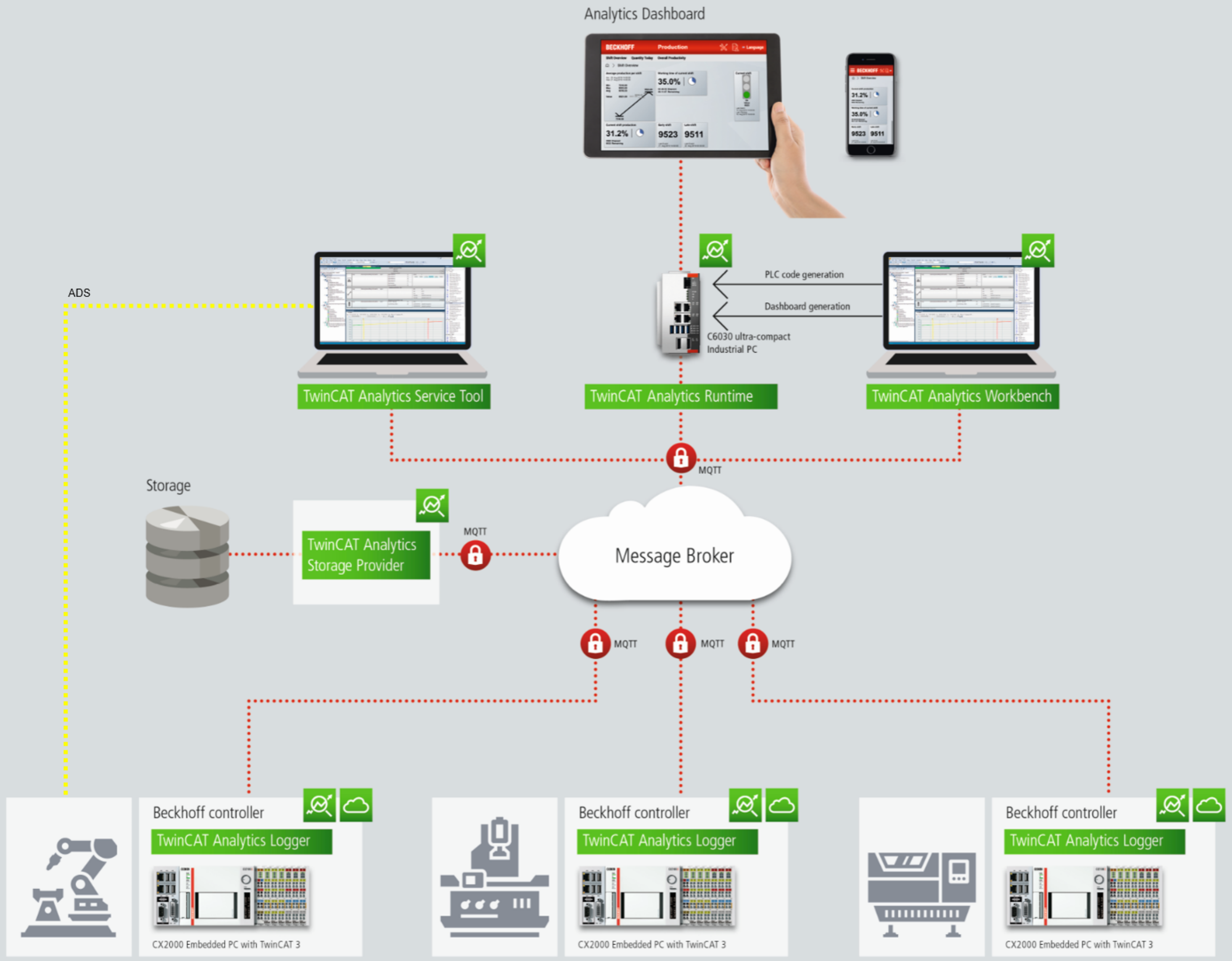
□ Queue messages when disconnected

□ Store in file

Queue Size: 0 | (KB)

4.2 Communication

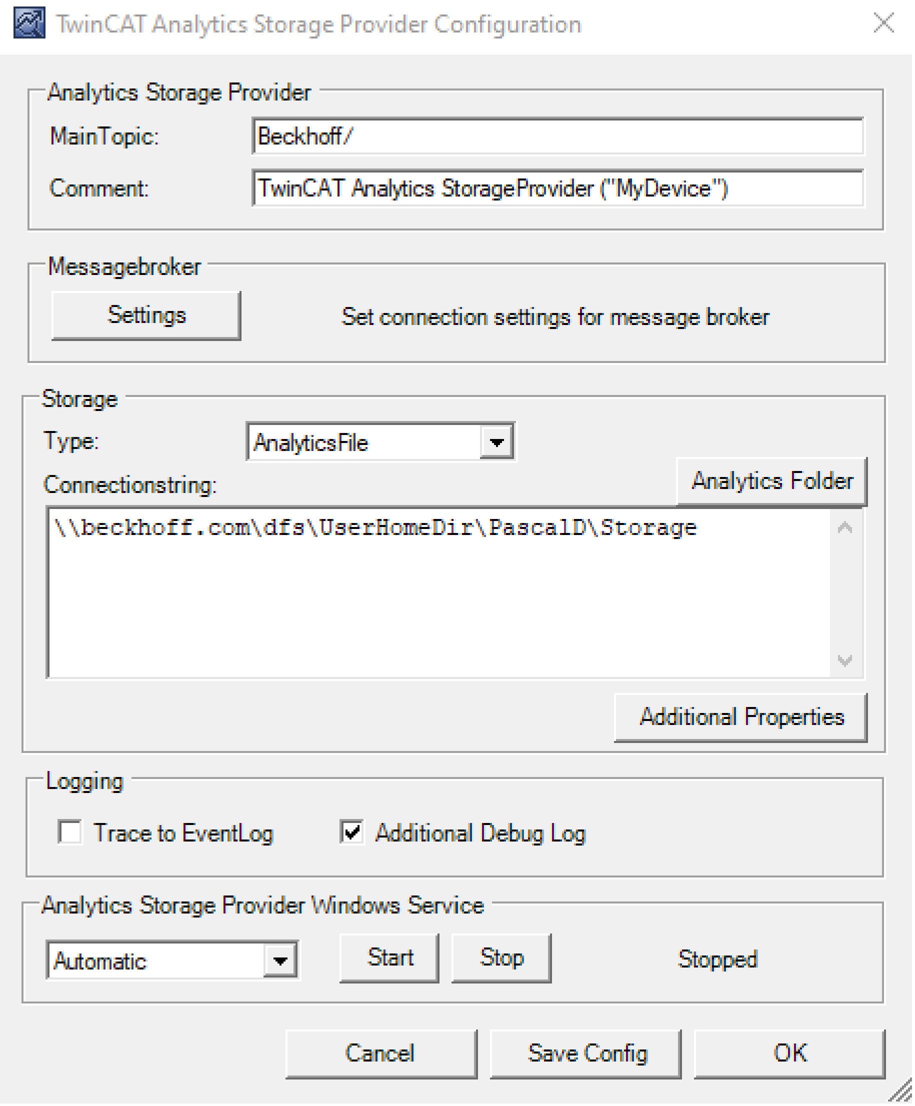
Currently, the Analytics workflow is fully mappable via MQTT. The engineering tools can also access the data of the machines via ADS and carry out analyzes.


Analytics Dashboard
ADS
TwinCAT Analytics Service Tool
PLC code generation
Dashboard generation
C6030 ultra-compact Industrial PC
TwinCAT Analytics Runtime
TwinCAT Analytics Workbench
MQTT
Storage
TwinCAT Analytics Storage Provider
MQTT
Message Broker
MQTT
MQTT
MQTT
Beckhoff controller
TwinCAT Analytics Logger
CX2000 Embedded PC with TwinCAT 3
Beckhoff controller
TwinCAT Analytics Logger
CX2000 Embedded PC with TwinCAT 3
Beckhoff controller
TwinCAT Analytics Logger
CX2000 Embedded PC with TwinCAT 3

If you choose for the IoT communication protocol MQTT you have to setup a native MQTT Message Broker somewhere in the network (VM in a cloud system is also possible). This Message Broker provides a decoupling of the different applications in the Analytics Workflow.

4.3 Historicize data

After the TwinCAT Analytics Storage Provider has been installed, the service running in the background can be configured. You will find the TwinCAT Analytics.StorageProvider.Configurator application in the folder *C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\Tools*.


TwinCAT Analytics Storage Provider Configuration

Analytics Storage Provider

MainTopic: Beckhoff/

Comment: TwinCAT Analytics StorageProvider ("MyDevice")

Messagebroker

Set connection settings for message broker

Storage

Type: AnalyticsFile

Connectionstring: \\beckhoff.com\dfs\UserHomeDir\PascalD\Storage

Logging

□ Trace to EventLog ☑ Additional Debug Log

Analytics Storage Provider Windows Service

Automatic

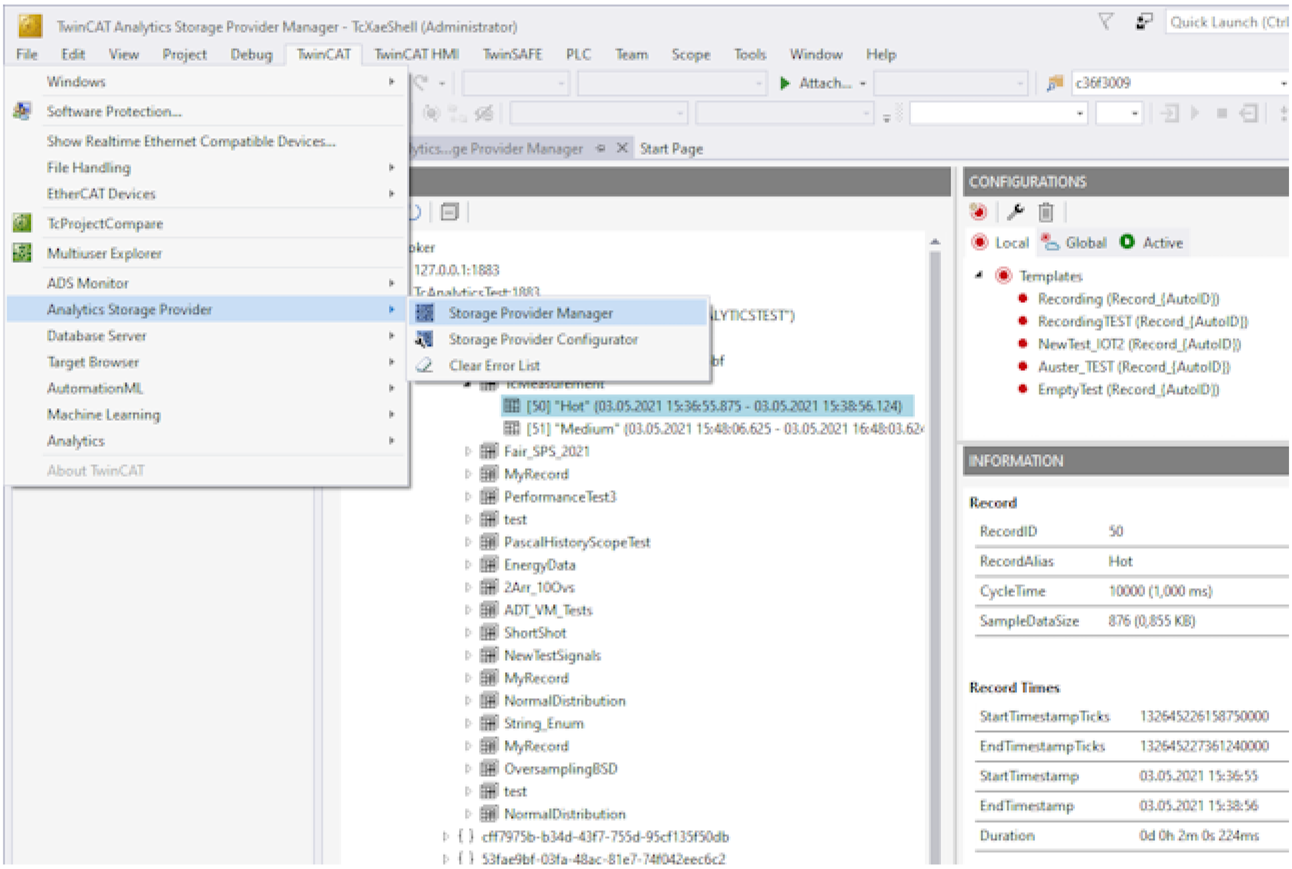
Stopped

The main part of the topic can be defined in the configuration as well as the comment, which is used for identification if more than one Storage Provider is registered with the message broker.

You can make the message broker settings and decide on a storage type:

- Analytics File (binary file)
- CSV file
- Microsoft SQL (binary / plain text)
- InlfuxDB (plain text)
- Microsoft Azure Blob (Azure Cloud required)

At last you can save the configuration and start the service. The next step is to configure the specific recording. For this you should select the **Storage Provider Manager** in your development environment.



oker
127.0.0.1:1883
TcAnalyticsTest:1883
...LYTICSTEST")
...bf
TcMeasurement
[50] "Hot" (03.05.2021 15:36:55.875 - 03.05.2021 15:38:56.124)
[51] "Medium" (03.05.2021 15:48:06.625 - 03.05.2021 16:48:03.62
Fair_SPS_2021
MyRecord
PerformanceTest3
test
PascalHistoryScopeTest
EnergyData
2Arr_10Ovs
ADT_VM_Tests
ShortShot
NewTestSignals
MyRecord
NormalDistribution
String_Enum
MyRecord
OversamplingBSD
test
NormalDistribution
{ } cff7975b-b34d-43f7-755d-95cf135f50db
{ } 53fae9bf-03fa-48ac-81e7-74f042eec6c2

Templates
- Recording (Record_{AutoID})
- RecordingTEST (Record_{AutoID})
- NewTest_IOT2 (Record_{AutoID})
- Auster_TEST (Record_{AutoID})
- EmptyTest (Record_{AutoID})

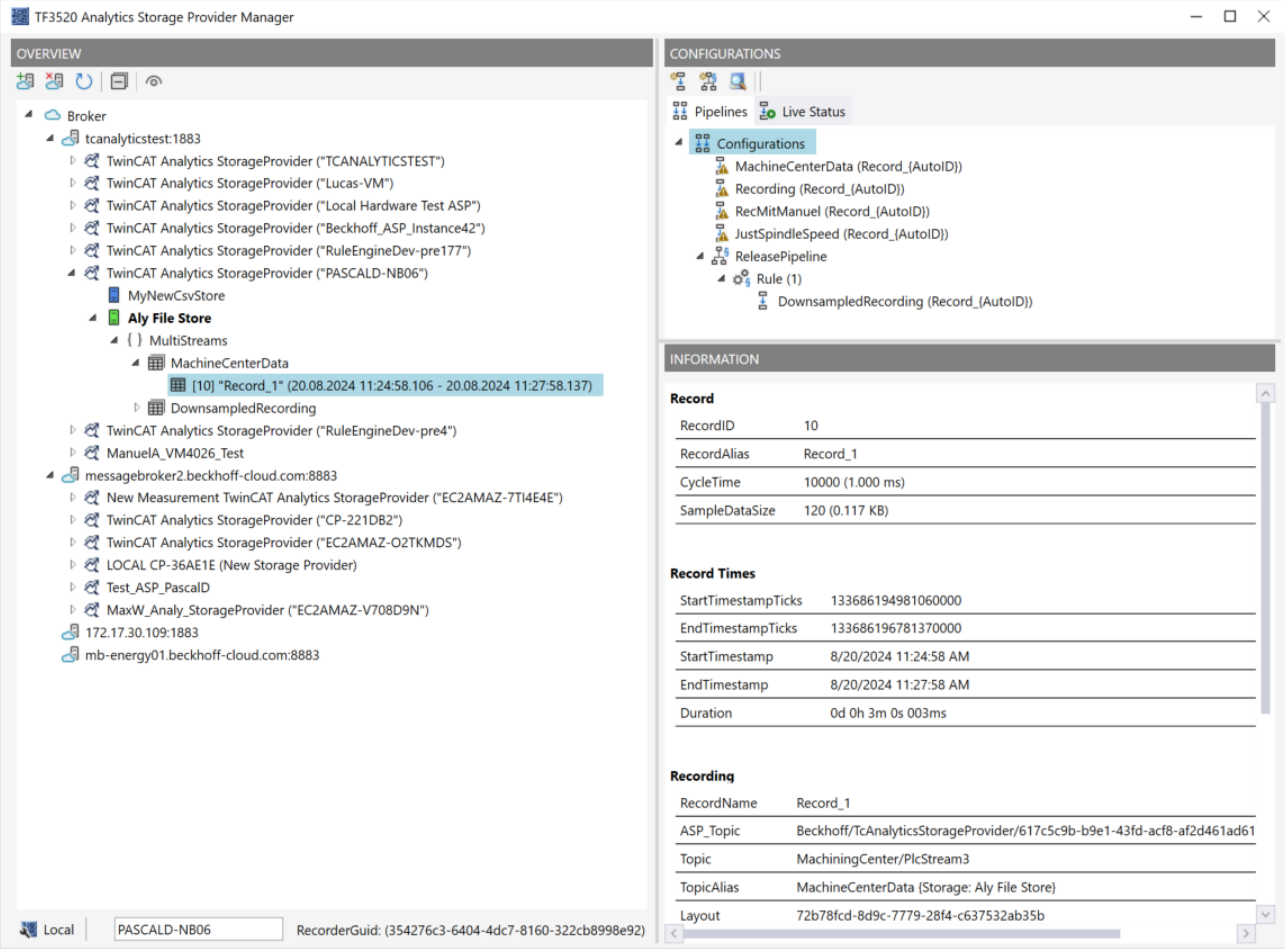
Record

RecordID	50
RecordAlias	Hot
CycleTime	10000 (1,000 ms)
SampleDataSize	876 (0,855 KB)

Record Times

StartTimestampTicks	132645226158750000
EndTimestampTicks	132645227361240000
StartTimestamp	03.05.2021 15:36:55
EndTimestamp	03.05.2021 15:38:56
Duration	0d 0h 2m 0s 224ms

With the Storage Provider Recorder recording definitions can be created, started and managed. In addition, it is possible to manage the data memories of individual Analytics Storage Providers. All important properties of the found Analytics Storage Providers and historized data are clearly displayed.



Broker
tcanalyticstest:1883
TwinCAT Analytics StorageProvider ("TCANALYTICSTEST")
TwinCAT Analytics StorageProvider ("Lucas-VM")
TwinCAT Analytics StorageProvider ("Local Hardware Test ASP")
TwinCAT Analytics StorageProvider ("Beckhoff_ASP_Instance42")
TwinCAT Analytics StorageProvider ("RuleEngineDev-pre177")
TwinCAT Analytics StorageProvider ("PASCALD-NB06")
MyNewCsvStore
Aly File Store
{ } MultiStreams
MachineCenterData
[10] "Record_1" (20.08.2024 11:24:58.106 - 20.08.2024 11:27:58.137)
DownsampledRecording
TwinCAT Analytics StorageProvider ("RuleEngineDev-pre4")
ManuelA_VM4026_Test
messagebroker2.beckhoff-cloud.com:8883
New Measurement TwinCAT Analytics StorageProvider ("EC2AMAZ-7TI4E4E")
TwinCAT Analytics StorageProvider ("CP-221DB2")
TwinCAT Analytics StorageProvider ("EC2AMAZ-O2TKMDS")
LOCAL CP-36AE1E (New Storage Provider)
Test_ASP_PascalD
MaxW_Analy_StorageProvider ("EC2AMAZ-V708D9N")
172.17.30.109:1883
mb-energy01.beckhoff-cloud.com:8883

Local PASCALD-NB06 RecorderGuid: (354276c3-6404-4dc7-8160-322cb8998e92)

Configurations
MachineCenterData (Record_{AutoID})
Recording (Record_{AutoID})
RecMitManuel (Record_{AutoID})
JustSpindleSpeed (Record_{AutoID})
ReleasePipeline
Rule (1)
DownsampledRecording (Record_{AutoID})

Record

RecordID	10
RecordAlias	Record_1
CycleTime	10000 (1.000 ms)
SampleDataSize	120 (0.117 KB)

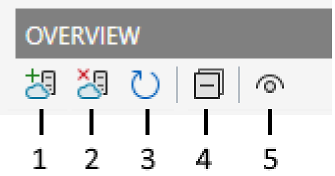
Record Times

StartTimestampTicks	133686194981060000
EndTimestampTicks	133686196781370000
StartTimestamp	8/20/2024 11:24:58 AM
EndTimestamp	8/20/2024 11:27:58 AM
Duration	0d 0h 3m 0s 003ms

Recording

RecordName	Record_1
ASP_Topic	Beckhoff/TcAnalyticsStorageProvider/617c5c9b-b9e1-43fd-acf8-af2d461ad61
Topic	MachiningCenter/PlcStream3
TopicAlias	MachineCenterData (Storage: Aly File Store)
Layout	72b78fcd-8d9c-7779-28f4-c637532ab35b

Toolbar Manager window ("OVERVIEW")

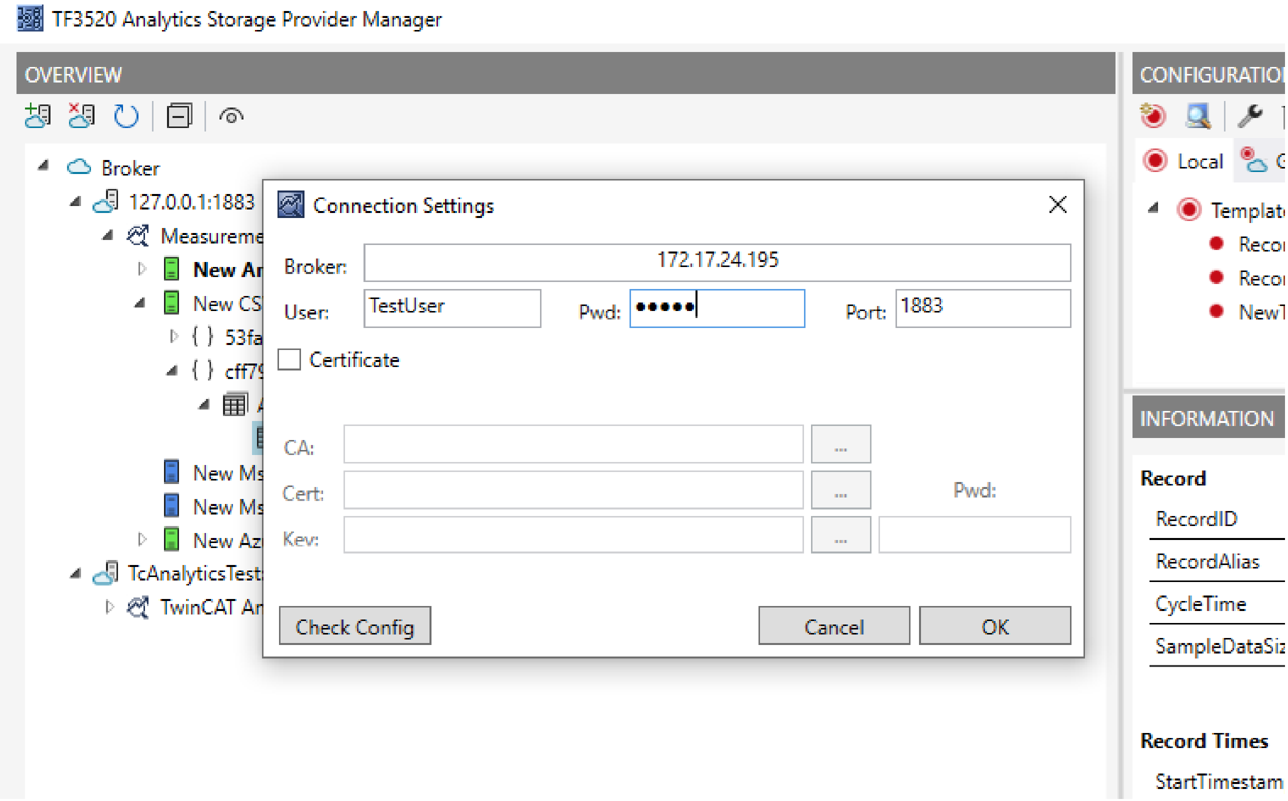


1	Add new broker
2	Remove selected broker
3	Refresh display
4	Collapse all nodes
5	View switch between dark/light mode

Function Manager window ("OVERVIEW")

First assign a **RecorderAlias**. This helps to group the started recordings and to find its self started ones again.

After that, one or more brokers can be set up. This is done via the already known input mask for MQTT connection properties.



Connection Settings

Broker: 172.17.24.195

User: TestUser Pwd: ••••• Port: 1883

□ Certificate

CA:

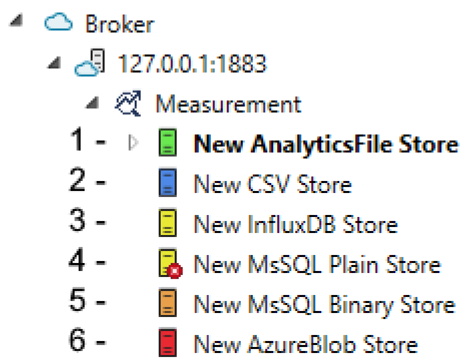
Cert:

Pwd:

Key:

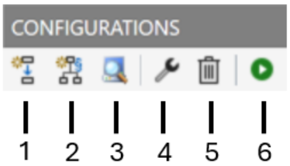
Once a connection to the broker could be established, all Analytics Storage Providers connected to it will be listed.

"Storage" status


Broker
127.0.0.1:1883
Measurement
1 - New AnalyticsFile Store
2 - New CSV Store
3 - New InfluxDB Store
4 - New MsSQL Plain Store
5 - New MsSQL Binary Store
6 - New AzureBlob Store

1	Storage Online
2	Storage Offline
3	Storage starts
4	Storage starts with error. Still trying to start it
5	Storage is shut down
6	Storage is in the error state

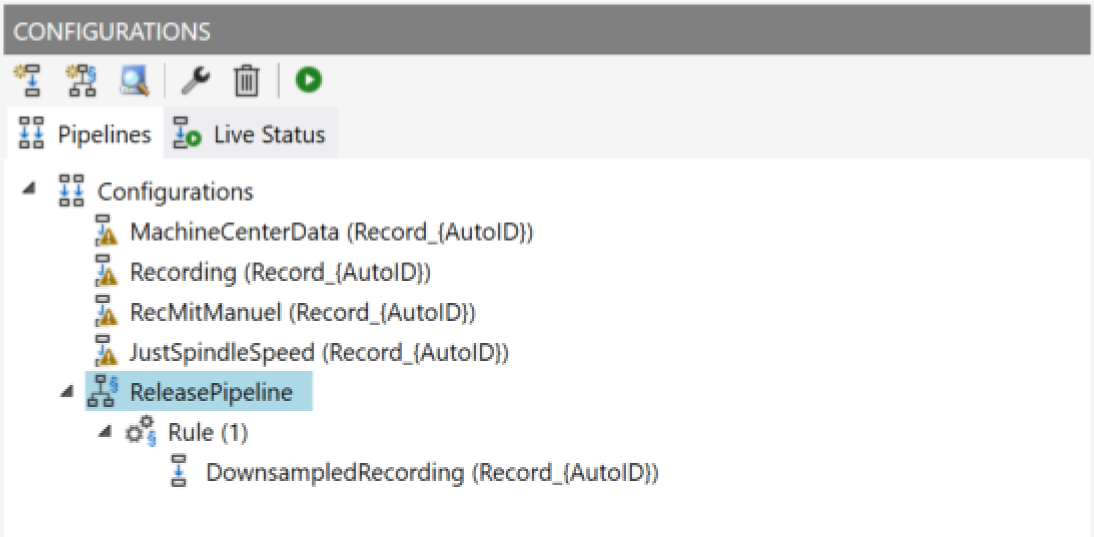
Toolbar Manager window ("CONFIGURATIONS")



1	Create a new pipeline
2	Create a new pipeline with Rule Engine
3	Open Target Browser for connecting simple pipelines
4	Edit a selected pipeline
5	Delete a selected pipeline
6	Start a selected pipeline

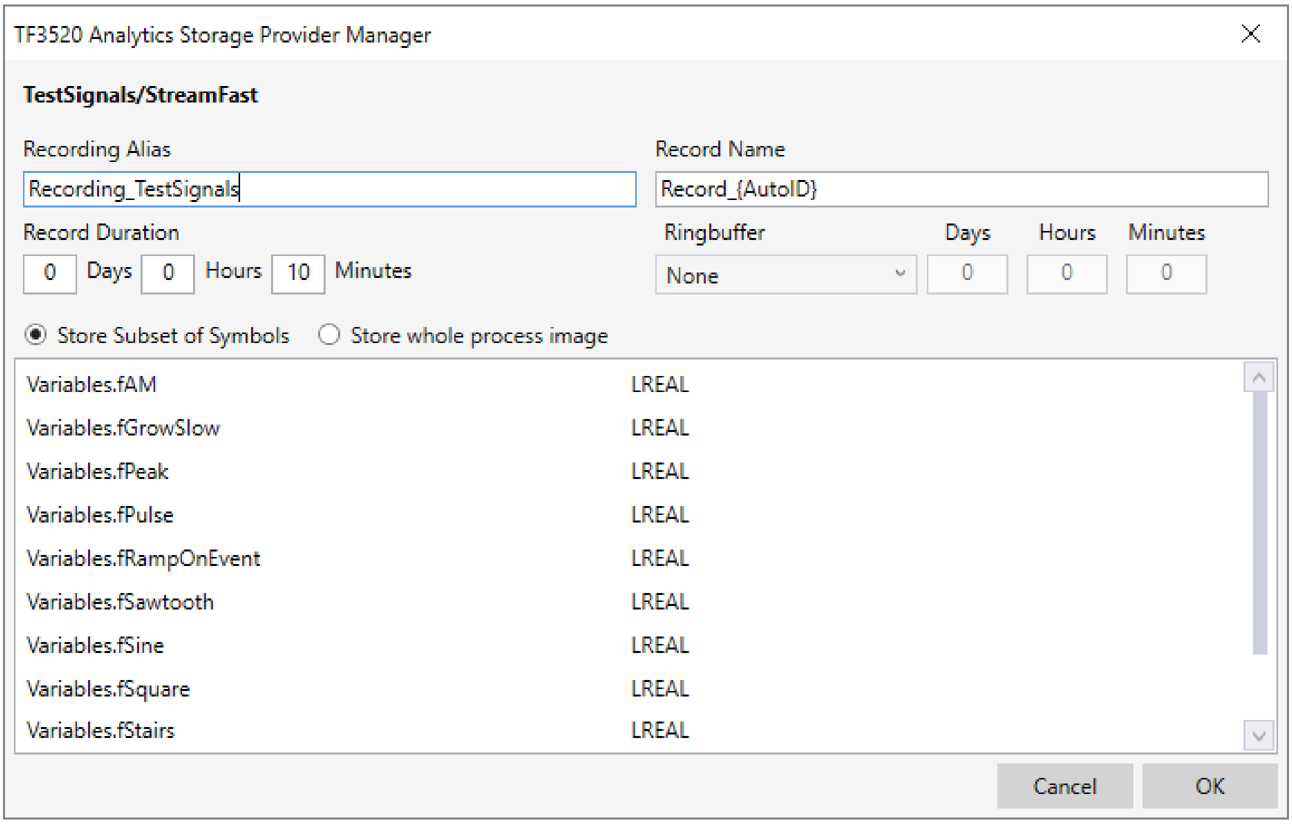
Function Manager window ("CONFIGURATIONS")

The window is divided into two tabs. Pipelines and Live Status. Under Pipelines you will find the configurations of your pipelines. You can define new pipelines from here. Edit existing. Delete or start.



Configurations
MachineCenterData (Record_{AutoID})
Recording (Record_{AutoID})
RecMitManuel (Record_{AutoID})
JustSpindleSpeed (Record_{AutoID})
ReleasePipeline
Rule (1)
DownsampledRecording (Record_{AutoID})

To create a new simple pipeline, click the "Create new pipeline" button. The following dialog opens.



TF3520 Analytics Storage Provider Manager

TestSignals/StreamFast

Recording Alias: Recording_TestSignals

Record Name: Record_{AutoID}

Record Duration: 0 Days 0 Hours 10 Minutes

Ringbuffer: None; Days 0; Hours 0; Minutes 0

● Store Subset of Symbols ○ Store whole process image

Variables.fAM	LREAL
Variables.fGrowSlow	LREAL
Variables.fPeak	LREAL
Variables.fPulse	LREAL
Variables.fRampOnEvent	LREAL
Variables.fSawtooth	LREAL
Variables.fSine	LREAL
Variables.fSquare	LREAL
Variables.fStairs	LREAL

You can now drag and drop the symbols you want to record from the Target Browser into the dialog. You also assign a Recording Alias and a Record Name.

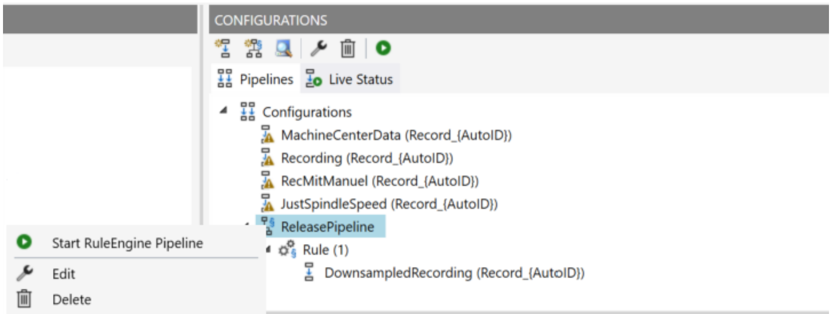
Various placeholders are available for the Record Name:

"{AutoID}"	
"{Topic}"	
"{SystemID}"	
"{Layout}"	
"{CycleTime}"	
"{SampleSize}"	
"{RecordStart}"	

You can also configure recording names and a duration (otherwise the recording will run endlessly until it is stopped manually). A ring buffer can be set according to storage space or time.

The entries are confirmed with **OK** and a new local recording definition is created.

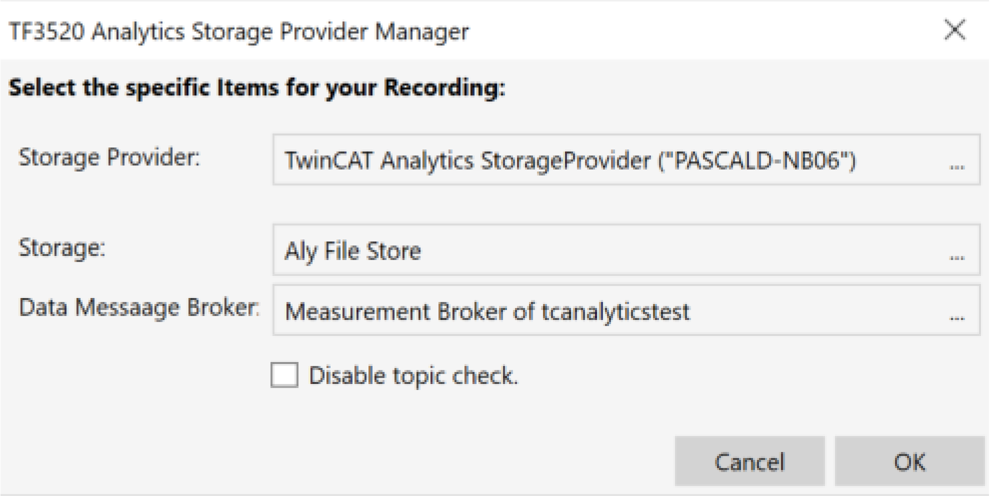
It is now possible to start this definition directly via the toolbar or the context menu.



Configurations
- MachineCenterData (Record_{AutoID})
- Recording (Record_{AutoID})
- RecMitManuel (Record_{AutoID})
- JustSpindleSpeed (Record_{AutoID})
- ReleasePipeline
  - Rule (1)
    - DownsampledRecording (Record_{AutoID})

However, it is also possible to make the definition globally accessible. This can be done via the context menu with the entry **Publish Recording**.

The following dialog then opens:



TF3520 Analytics Storage Provider Manager

Select the specific Items for your Recording:

Storage Provider: TwinCAT Analytics StorageProvider ("PASCALD-NB06")

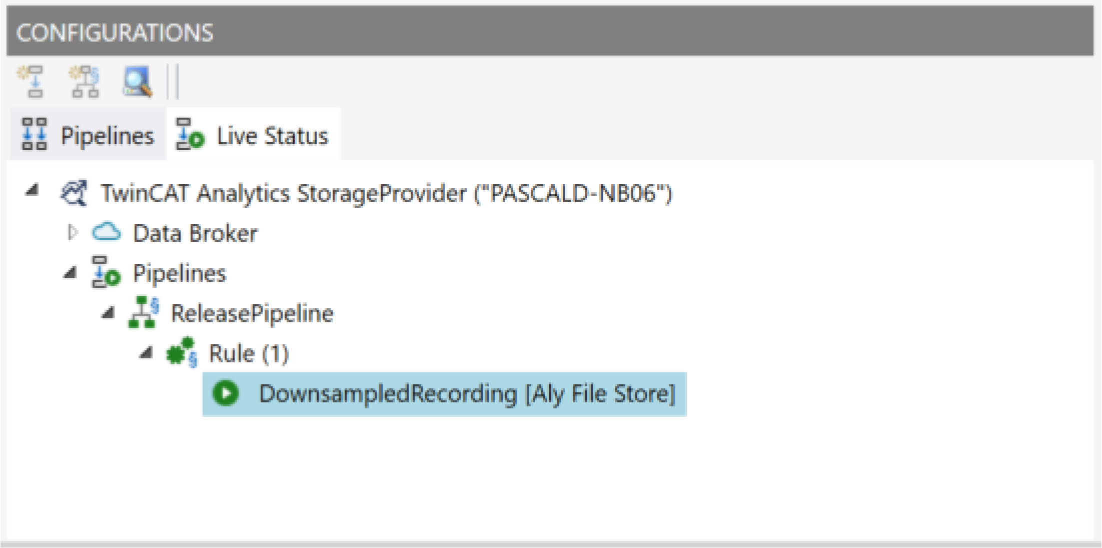
Storage: Aly File Store

Data Messaage Broker: Measurement Broker of tcanalyticstest

□ Disable topic check.

Here you can now select the desired Analytics Storage Provider via which the definition is to be published. In addition, the definition is assigned a Storage and a Data Broker of the selected Analytics Storage Provider. After the selection, the recording definition is confirmed with **OK** and published to the selected Analytics Storage Provider. This means that it can be found by any Storage Provider Manager that is connected to the MQTT Broker.

After starting a pipeline, the view automatically jumps to the second tab, the Live Status.

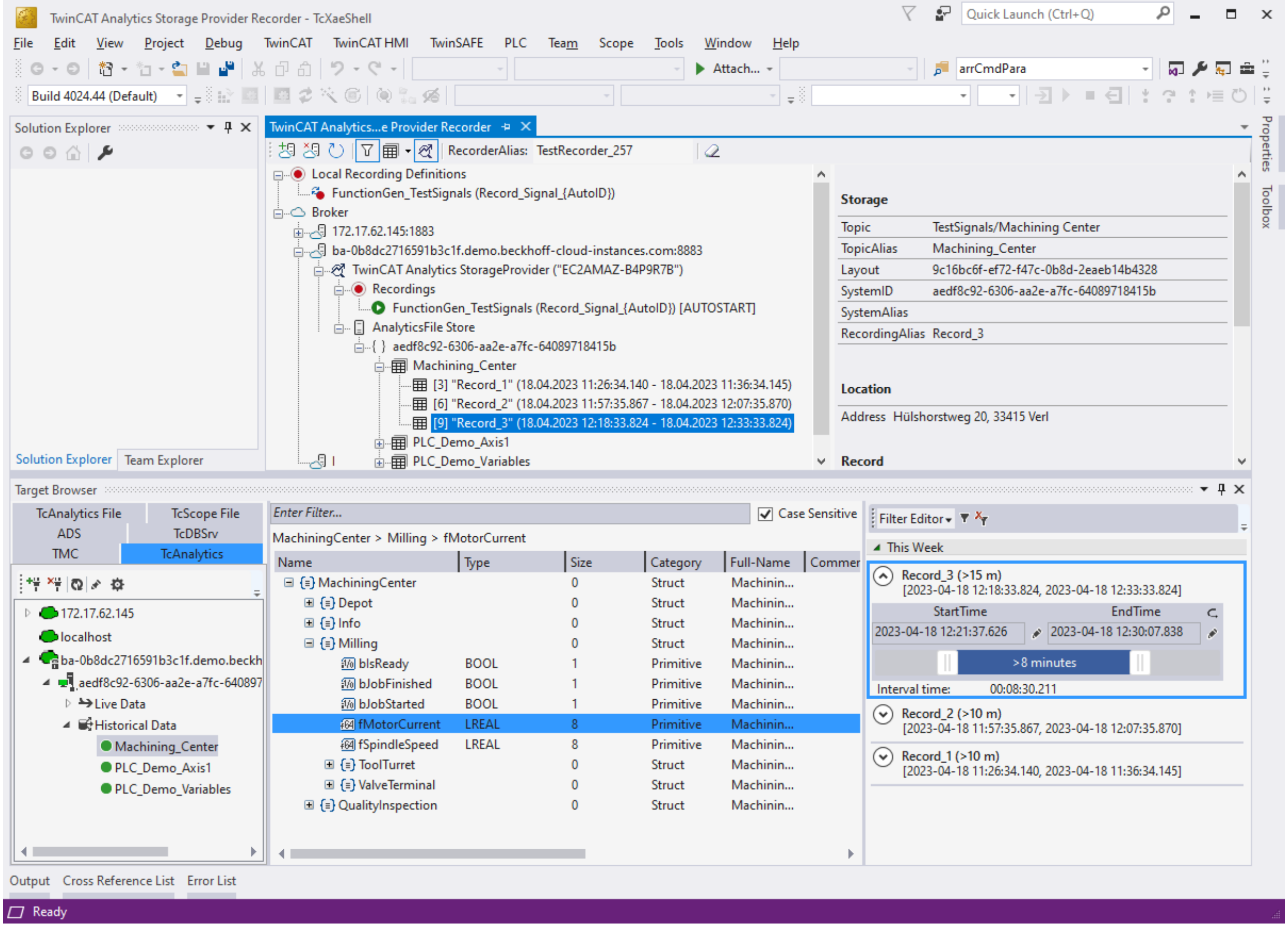


CONFIGURATIONS
Pipelines
Live Status
TwinCAT Analytics StorageProvider ("PASCALD-NB06")
Data Broker
Pipelines
ReleasePipeline
Rule (1)
DownsampledRecording [Aly File Store]

All active recordings from all users are listed here. The recordings can be ended in this tab and it is also possible to jump to the resulting record.

Use historized data

After and also during recording, you can select the historical data as input for your analysis in Target Browser. In the Target Browser, you will find a new control on the right side for the historical data. There you can select the timespan for your data.



RecorderAlias: TestRecorder_257

```
Local Recording Definitions
    FunctionGen_TestSignals (Record_Signal_{AutoID})
Broker
    172.17.62.145:1883
    ba-0b8dc2716591b3c1f.demo.beckhoff-cloud-instances.com:8883
        TwinCAT Analytics StorageProvider ("EC2AMAZ-B4P9R7B")
            Recordings
                FunctionGen_TestSignals (Record_Signal_{AutoID}) [AUTOSTART]
            AnalyticsFile Store
                { } aedf8c92-6306-aa2e-a7fc-64089718415b
                    Machining_Center
                        [3] "Record_1" (18.04.2023 11:26:34.140 - 18.04.2023 11:36:34.145)
                        [6] "Record_2" (18.04.2023 11:57:35.867 - 18.04.2023 12:07:35.870)
                        [9] "Record_3" (18.04.2023 12:18:33.824 - 18.04.2023 12:33:33.824)
                    PLC_Demo_Axis1
                    PLC_Demo_Variables
```

Storage

Topic	TestSignals/Machining Center
TopicAlias	Machining_Center
Layout	9c16bc6f-ef72-f47c-0b8d-2eaeb14b4328
SystemID	aedf8c92-6306-aa2e-a7fc-64089718415b
SystemAlias	
RecordingAlias	Record_3

Location

Address Hülshorstweg 20, 33415 Verl

Record

```
172.17.62.145
localhost
ba-0b8dc2716591b3c1f.demo.beckh
    aedf8c92-6306-aa2e-a7fc-640897
        Live Data
        Historical Data
            Machining_Center
            PLC_Demo_Axis1
            PLC_Demo_Variables
```

MachiningCenter > Milling > fMotorCurrent

Name	Type	Size	Category	Full-Name	Commer
MachiningCenter		0	Struct	Machinin...	
Depot		0	Struct	Machinin...	
Info		0	Struct	Machinin...	
Milling		0	Struct	Machinin...	
bIsReady	BOOL	1	Primitive	Machinin...	
bJobFinished	BOOL	1	Primitive	Machinin...	
bJobStarted	BOOL	1	Primitive	Machinin...	
fMotorCurrent	LREAL	8	Primitive	Machinin...	
fSpindleSpeed	LREAL	8	Primitive	Machinin...	
ToolTurret		0	Struct	Machinin...	
ValveTerminal		0	Struct	Machinin...	
QualityInspection		0	Struct	Machinin...	

This Week

Record_3 (>15 m)
[2023-04-18 12:18:33.824, 2023-04-18 12:33:33.824]

StartTime	EndTime
2023-04-18 12:21:37.626	2023-04-18 12:30:07.838

>8 minutes

Interval time: 00:08:30.211

Record_2 (>10 m)
[2023-04-18 11:57:35.867, 2023-04-18 12:07:35.870]

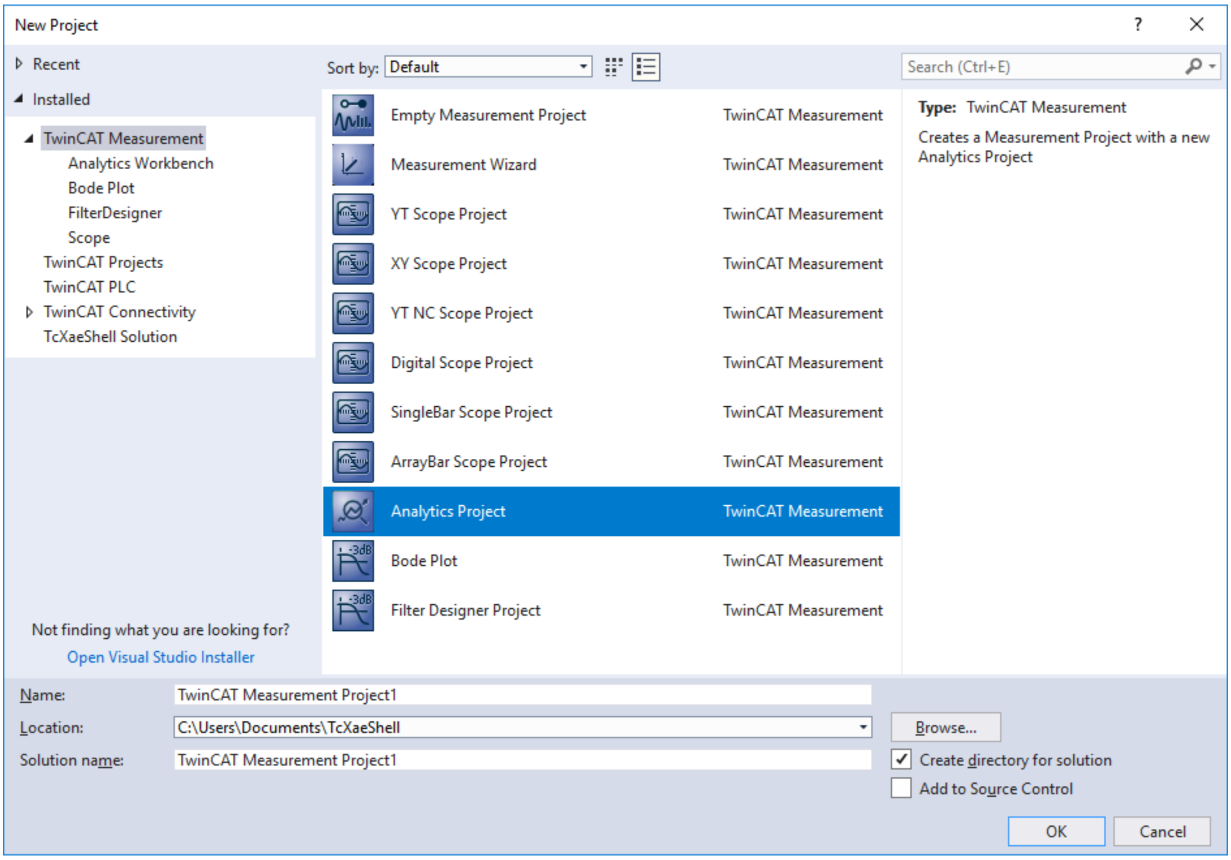
Record_1 (>10 m)
[2023-04-18 11:26:34.140, 2023-04-18 11:36:34.145]

4.4 Analyse data

✓ Open your TwinCAT Engineering environment to start the data analysis.

1. Open **Visual Studio® > File > New > Project…**

2. Select the **Analytics project template** from **TwinCAT Measurement**.



New Project

- Recent
- Installed
  - TwinCAT Measurement
    - Analytics Workbench
    - Bode Plot
    - FilterDesigner
    - Scope
  - TwinCAT Projects
  - TwinCAT PLC
  - TwinCAT Connectivity
  - TcXaeShell Solution

Sort by: Default

Empty Measurement Project	TwinCAT Measurement
Measurement Wizard	TwinCAT Measurement
YT Scope Project	TwinCAT Measurement
XY Scope Project	TwinCAT Measurement
YT NC Scope Project	TwinCAT Measurement
Digital Scope Project	TwinCAT Measurement
SingleBar Scope Project	TwinCAT Measurement
ArrayBar Scope Project	TwinCAT Measurement
Analytics Project	TwinCAT Measurement
Bode Plot	TwinCAT Measurement
Filter Designer Project	TwinCAT Measurement

Type: TwinCAT Measurement

Creates a Measurement Project with a new Analytics Project

Not finding what you are looking for?
Open Visual Studio Installer

Name: TwinCAT Measurement Project1

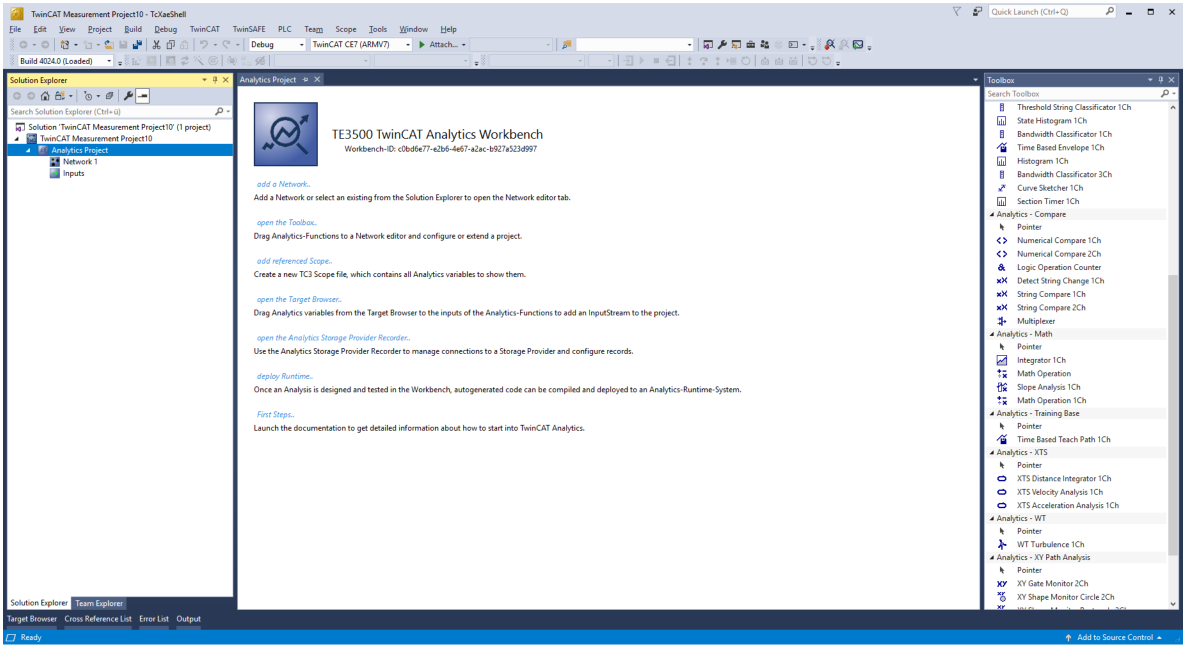
Location: C:\Users\Documents\TcXaeShell

Solution name: TwinCAT Measurement Project1

☑ Create directory for solution

□ Add to Source Control

⇨ The new project is displayed in the Solution Explorer. After clicking the **Analytics Project** tree node element a start window opens where you can select your first action. From here you can add a network, open the **Toolbox**, open the **Target Browser** or open the **Analytics Storage Provider Recorder**. In the following steps you will perform all these actions.



TwinCAT Measurement Project10 - TcXaeShell

Solution 'TwinCAT Measurement Project10' (1 project)
- TwinCAT Measurement Project10
  - Analytics Project
    - Network 1
    - Inputs

TE3500 TwinCAT Analytics Workbench
Workbench-ID: c0bd6e77-e2b6-4e67-a2ac-b927a523d997

add a Network..
Add a Network or select an existing from the Solution Explorer to open the Network editor tab.

open the Toolbox..
Drag Analytics-Functions to a Network editor and configure or extend a project.

add referenced Scope..
Create a new TC3 Scope file, which contains all Analytics variables to show them.

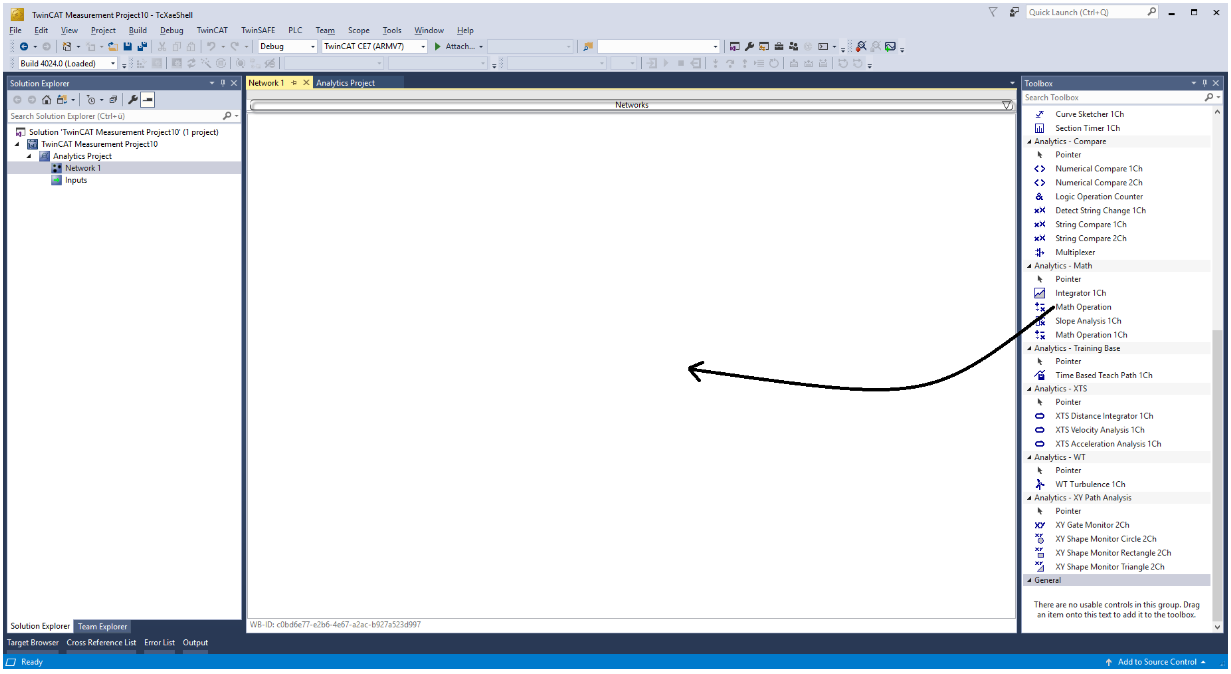
open the Target Browser..
Drag Analytics variables from the Target Browser to the inputs of the Analytics-Functions to add an InputStream to the project.

open the Analytics Storage Provider Recorder..
Use the Analytics Storage Provider Recorder to manage connections to a Storage Provider and configure records.

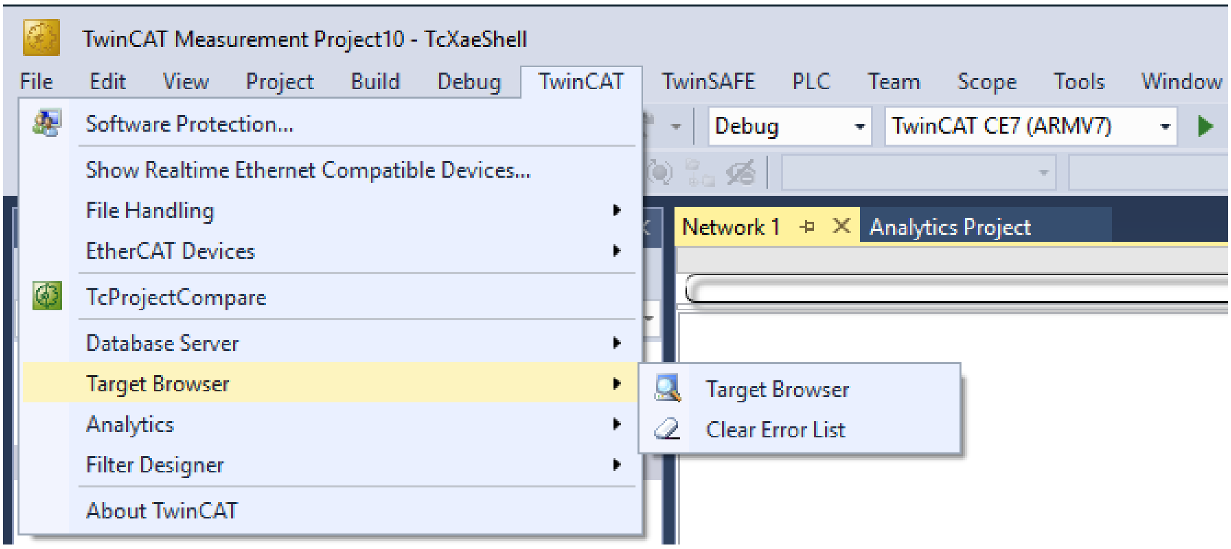
deploy Runtime..
Once an Analysis is designed and tested in the Workbench, autogenerated code can be compiled and deployed to an Analytics-Runtime-System.

First Steps..
Launch the documentation to get detailed information about how to start into TwinCAT Analytics.

3. It makes sense to open the **Toolbox** of Visual Studio® first. There you will find all the algorithms supported by TwinCAT Analytics. Algorithms need to be grouped and organized into networks. Right-click **Analytics Project** to add a new network, or add a network using the start page. The first network is always generated by default.

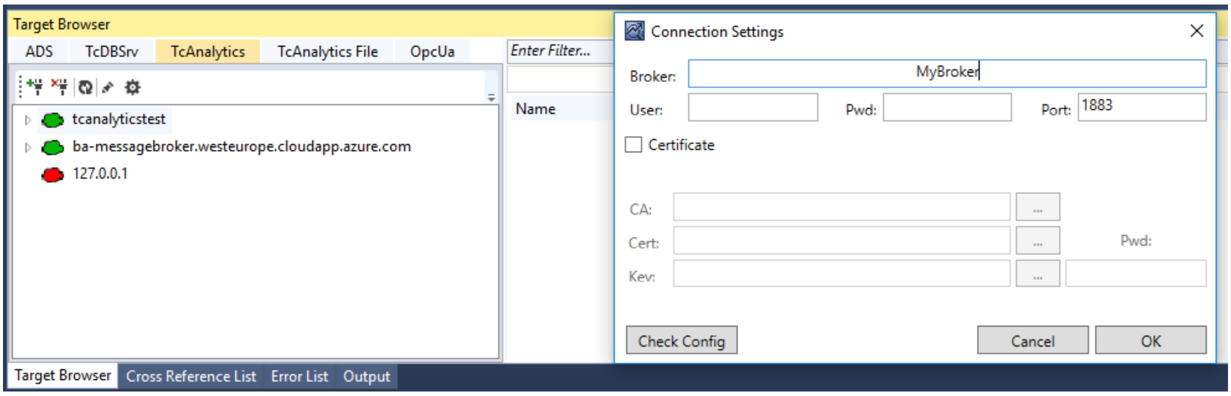


4. When you click on the network, an editor opens. Now you can drag and drop the desired algorithm into the editor interface.
5. After selecting the algorithm, you need to connect input variables to the modules (algorithm). To do this, open the **Target Browser**.
TwinCAT > Target Browser > Target Browser



6. Now select the **TcAnalytics** or **TcAnalyticsFile** tab in the Target Browser. Continue with the tab **TcAnalytics** (MQTT).

7. Click the icon highlighted in green in the toolbar of this Analytics extension. A window opens in which you can specify the connectivity data of your message broker.



Connection Settings

Broker: MyBroker

User:

Pwd:

Port: 1883

□ Certificate

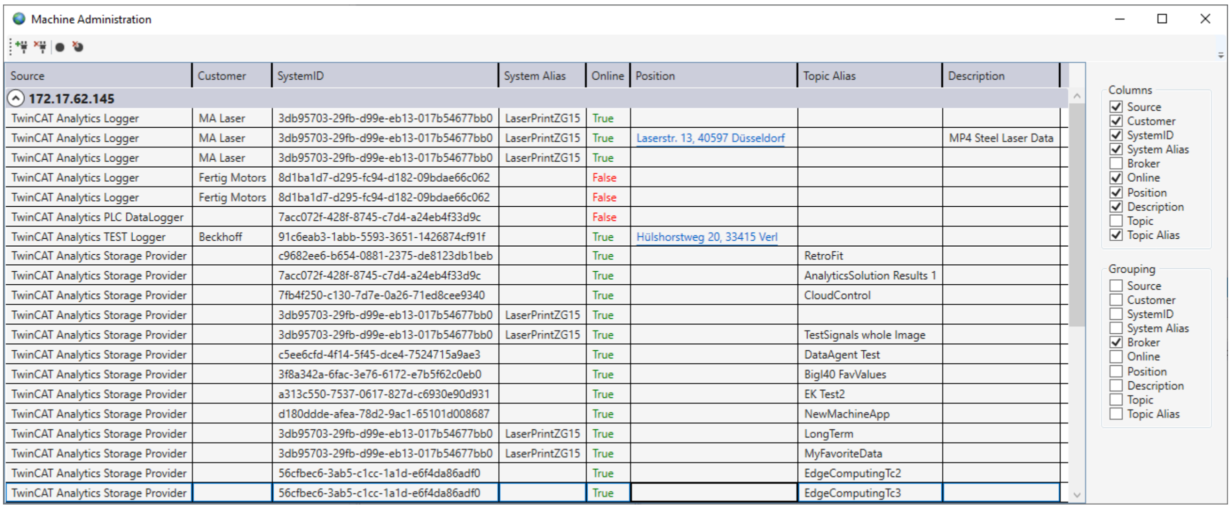
CA:

Cert:

Kev:

Pwd:

8. Select your MQTT Analytics client (TwinCAT Analytics Logger, TwinCAT IoT Data Agent or Beckhoff EK9160). There is a unique ID for each control. This ID is displayed in the Target Browser.
9. Clicking on the **gear icon**, you will get to the Machine Administration page. Here you can assign a system alias name that will be displayed in the Target Browser instead of the ID.

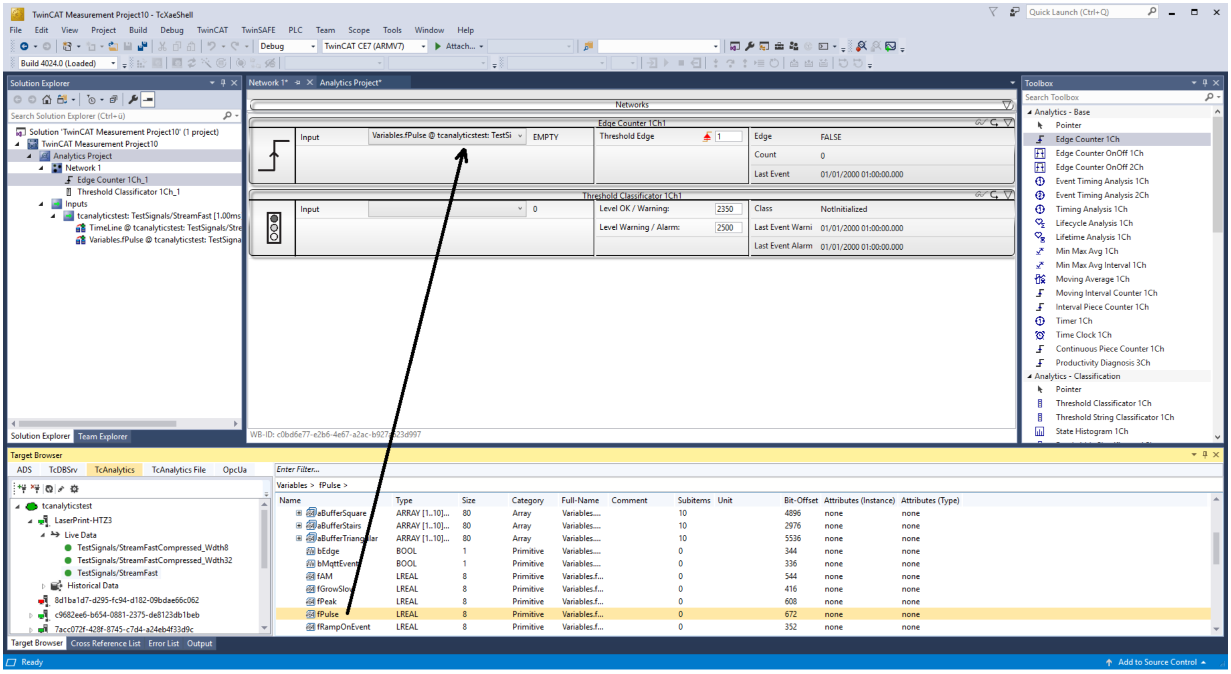

Machine Administration

Source	Customer	SystemID	System Alias	Online	Position	Topic Alias	Description
172.17.62.145							
TwinCAT Analytics Logger	MA Laser	3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True			
TwinCAT Analytics Logger	MA Laser	3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True	Laserstr. 13, 40597 Düsseldorf		MP4 Steel Laser Data
TwinCAT Analytics Logger	MA Laser	3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True			
TwinCAT Analytics Logger	Fertig Motors	8d1ba1d7-d295-fc94-d182-09bdae66c062		False			
TwinCAT Analytics Logger	Fertig Motors	8d1ba1d7-d295-fc94-d182-09bdae66c062		False			
TwinCAT Analytics PLC DataLogger		7acc072f-428f-8745-c7d4-a24eb4f33d9c		False			
TwinCAT Analytics TEST Logger	Beckhoff	91c6eab3-1abb-5593-3651-1426874cf91f		True	Hülshorstweg 20, 33415 Verl		
TwinCAT Analytics Storage Provider		c9682ee6-b654-0881-2375-de8123db1beb		True		RetroFit	
TwinCAT Analytics Storage Provider		7acc072f-428f-8745-c7d4-a24eb4f33d9c		True		AnalyticsSolution Results 1	
TwinCAT Analytics Storage Provider		7fb4f250-c130-7d7e-0a26-71ed8cee9340		True		CloudControl	
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True			
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True		TestSignals whole Image	
TwinCAT Analytics Storage Provider		c5ee6cfd-4f14-5f45-dce4-7524715a9ae3		True		DataAgent Test	
TwinCAT Analytics Storage Provider		3f8a342a-6fac-3e76-6172-e7b5f62c0eb0		True		BigI40 FavValues	
TwinCAT Analytics Storage Provider		a313c550-7537-0617-827d-c6930e90d931		True		EK Test2	
TwinCAT Analytics Storage Provider		d180ddde-afea-78d2-9ac1-65101d008687		True		NewMachineApp	
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True		LongTerm	
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True		MyFavoriteData	
TwinCAT Analytics Storage Provider		56cfbec6-3ab5-c1cc-1a1d-e6f4da86adf0		True		EdgeComputingTc2	
TwinCAT Analytics Storage Provider		56cfbec6-3ab5-c1cc-1a1d-e6f4da86adf0		True		EdgeComputingTc3	

Columns
☑ Source
☑ Customer
☑ SystemID
☑ System Alias
□ Broker
☑ Online
☑ Position
☑ Description
□ Topic
☑ Topic Alias

Grouping
□ Source
□ Customer
□ SystemID
□ System Alias
☑ Broker
□ Online
□ Position
□ Description
□ Topic
□ Topic Alias

10. In the next step, you can choose between live data and historical data for each MQTT Analytics client. In this case, the historical data is provided by the TwinCAT Analytics Storage Provider.


TwinCAT Measurement Project10 - TcXaeShell
File Edit View Project Build Debug TwinCAT TwinSAFE PLC Team Scope Tools Window Help
Debug
TwinCAT CE7 (ARMV7)
Attach...
Build 4024.0 (Loaded)
Solution Explorer
Solution 'TwinCAT Measurement Project10' (1 project)
TwinCAT Measurement Project10
Analytics Project
Network 1
Edge Counter 1Ch_1
Threshold Classificator 1Ch_1
Inputs
tcanalyticstest: TestSignals/StreamFast [1.00ms]
TimeLine @ tcanalyticstest: TestSignals/Stre
Variables.fPulse @ tcanalyticstest: TestSigna
Solution Explorer
Team Explorer
Network 1*
Analytics Project*
Networks
Toolbox
Search Toolbox
Analytics - Base
Pointer
Edge Counter 1Ch
Edge Counter OnOff 1Ch
Edge Counter OnOff 2Ch
Event Timing Analysis 1Ch
Event Timing Analysis 2Ch
Timing Analysis 1Ch
Lifecycle Analysis 1Ch
Lifetime Analysis 1Ch
Min Max Avg 1Ch
Min Max Avg Interval 1Ch
Moving Average 1Ch
Moving Interval Counter 1Ch
Interval Piece Counter 1Ch
Timer 1Ch
Time Clock 1Ch
Continuous Piece Counter 1Ch
Productivity Diagnosis 3Ch
Analytics - Classification
Pointer
Threshold Classificator 1Ch
Threshold String Classificator 1Ch
State Histogram 1Ch
Target Browser
ADS
TcDBSrv
TcAnalytics
TcAnalytics File
OpcUa
tcanalyticstest
LaserPrint-HTZ3
Live Data
TestSignals/StreamFastCompressed_Wdth8
TestSignals/StreamFastCompressed_Wdth32
TestSignals/StreamFast
Historical Data
8d1ba1d7-d295-fc94-d182-09bdae66c062
c9682ee6-b654-0881-2375-de8123db1beb
7acc072f-428f-8745-c7d4-a24eb4f33d9c
Target Browser
Cross Reference List
Error List
Output
Ready
Add to Source Control

Edge Counter 1Ch1

Input: Variables.fPulse @ tcanalyticstest: TestSi EMPTY

Threshold Edge: 1

Edge: FALSE

Count: 0

Last Event: 01/01/2000 01:00:00.000

Threshold Classificator 1Ch1

Input: 0

Level OK / Warning: 2350

Level Warning / Alarm: 2500

Class: NotInitialized

Last Event Warni: 01/01/2000 01:00:00.000

Last Event Alarm: 01/01/2000 01:00:00.000

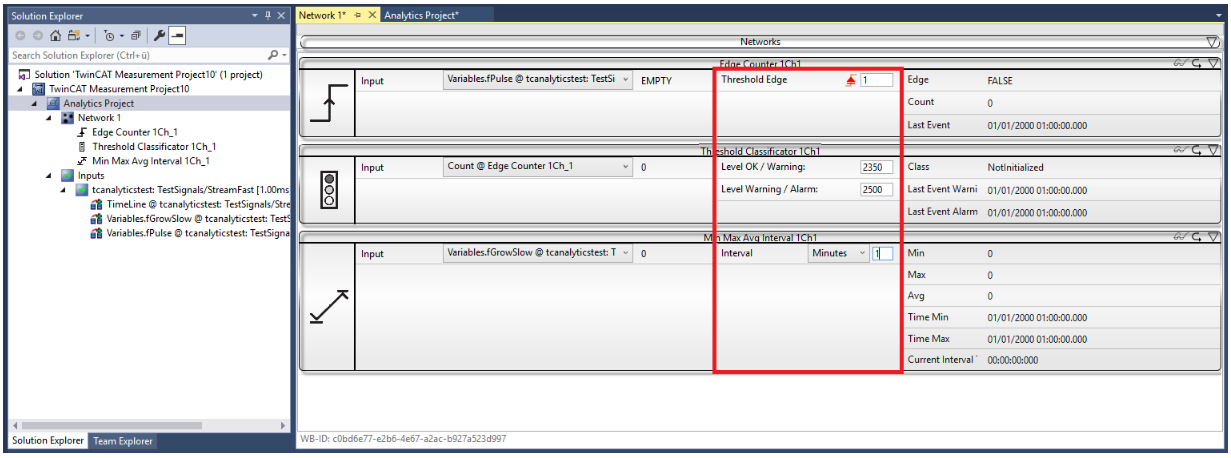
WB-ID: c0bd6e77-e2b6-4e67-a2ac-b927a23d997

Enter Filter...

Variables > fPulse >

Name	Type	Size	Category	Full-Name	Comment	Subitems	Unit	Bit-Offset	Attributes (Instance)	Attributes (Type)
aBufferSquare	ARRAY [1..10]...	80	Array	Variables....		10		4896	none	none
aBufferStairs	ARRAY [1..10]...	80	Array	Variables....		10		2976	none	none
aBufferTriangular	ARRAY [1..10]...	80	Array	Variables....		10		5536	none	none
bEdge	BOOL	1	Primitive	Variables....		0		344	none	none
bMqttEvent	BOOL	1	Primitive	Variables....		0		336	none	none
fAM	LREAL	8	Primitive	Variables.f...		0		544	none	none
fGrowSlow	LREAL	8	Primitive	Variables.f...		0		416	none	none
fPeak	LREAL	8	Primitive	Variables.f...		0		608	none	none
fPulse	LREAL	8	Primitive	Variables.f...		0		672	none	none
fRampOnEvent	LREAL	8	Primitive	Variables.f...		0		352	none	none

11. You can drag and drop the variables into the inputs of the specific algorithm. In most algorithms, conditions such as thresholds, time intervals, logical operators etc. can be specified. These settings are made in the middle of each module.



Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'TwinCAT Measurement Project10' (1 project)
TwinCAT Measurement Project10
Analytics Project
Network 1
Edge Counter 1Ch_1
Threshold Classificator 1Ch_1
Min Max Avg Interval 1Ch_1
Inputs
tcanalyticstest: TestSignals/StreamFast [1.00ms
TimeLine @ tcanalyticstest: TestSignals/Stre
Variables.fGrowSlow @ tcanalyticstest: TestS
Variables.fPulse @ tcanalyticstest: TestSigna
Solution Explorer
Team Explorer
Network 1*
Analytics Project*
Networks

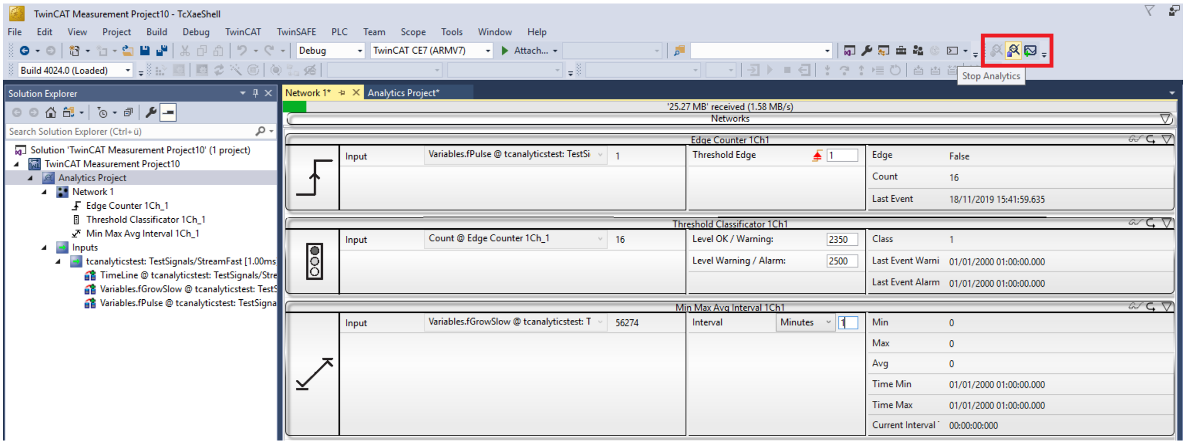
Edge Counter 1Ch1
Input: Variables.fPulse @ tcanalyticstest: TestSi EMPTY
Threshold Edge: 1
Edge: FALSE
Count: 0
Last Event: 01/01/2000 01:00:00.000

Threshold Classificator 1Ch1
Input: Count @ Edge Counter 1Ch_1 0
Level OK / Warning: 2350
Level Warning / Alarm: 2500
Class: NotInitialized
Last Event Warni: 01/01/2000 01:00:00.000
Last Event Alarm: 01/01/2000 01:00:00.000

Min Max Avg Interval 1Ch1
Input: Variables.fGrowSlow @ tcanalyticstest: T 0
Interval: Minutes 1
Min: 0
Max: 0
Avg: 0
Time Min: 01/01/2000 01:00:00.000
Time Max: 01/01/2000 01:00:00.000
Current Interval: 00:00:00:000

WB-ID: c0bd6e77-e2b6-4e67-a2ac-b927a523d997

⇨ Finally, your first Analytics Project is complete. To start the analysis, click **Start Analytics**. To stop the analysis, click **Stop Analytics**.



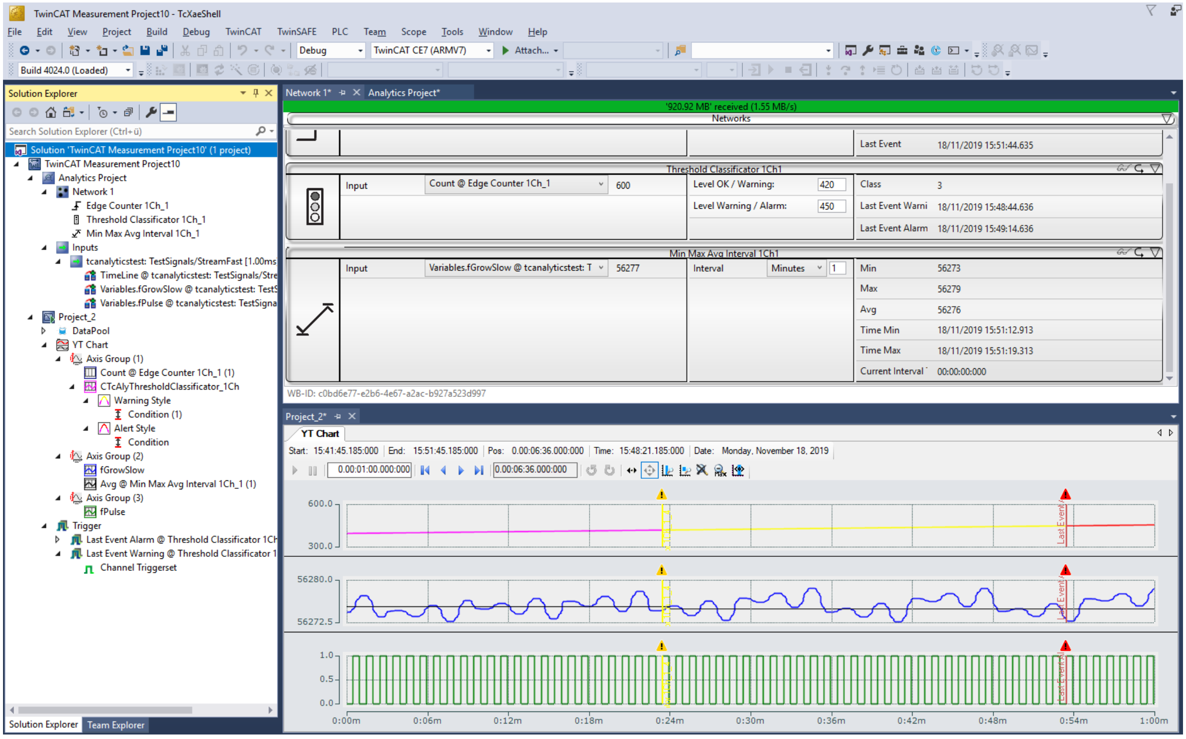
'25.27 MB' received (1.58 MB/s)

Edge Counter 1Ch1
Input: Variables.fPulse @ tcanalyticstest: TestSi 1
Threshold Edge: 1
Edge: False
Count: 16
Last Event: 18/11/2019 15:41:59.635

Threshold Classificator 1Ch1
Input: Count @ Edge Counter 1Ch_1 16
Level OK / Warning: 2350
Level Warning / Alarm: 2500
Class: 1
Last Event Warni: 01/01/2000 01:00:00.000
Last Event Alarm: 01/01/2000 01:00:00.000

Min Max Avg Interval 1Ch1
Input: Variables.fGrowSlow @ tcanalyticstest: T 56274
Interval: Minutes 1
Min: 0
Max: 0
Avg: 0
Time Min: 01/01/2000 01:00:00.000
Time Max: 01/01/2000 01:00:00.000
Current Interval: 00:00:00:000

⇨ Before starting the analysis or during runtime, you can click the **Add Reference Scope** button. This will automatically create a Scope configuration that matches your Analytics project.



TwinCAT Measurement Project10 - TcXaeShell
Solution Explorer
Solution 'TwinCAT Measurement Project10' (1 project)
TwinCAT Measurement Project10
Analytics Project
Network 1
Edge Counter 1Ch_1
Threshold Classificator 1Ch_1
Min Max Avg Interval 1Ch_1
Inputs
Project_2
DataPool
YT Chart
Axis Group (1)
Count @ Edge Counter 1Ch_1 (1)
CTcAlyThresholdClassificator_1Ch
Warning Style
Condition (1)
Alert Style
Condition
Axis Group (2)
fGrowSlow
Avg @ Min Max Avg Interval 1Ch_1 (1)
Axis Group (3)
fPulse
Trigger
Last Event Alarm @ Threshold Classificator 1Ch
Last Event Warning @ Threshold Classificator 1
Channel Triggerset
Project_2*
YT Chart
600.0
300.0
56280.0
56272.5
1.0
0.5
0.0
0:00m
0:06m
0:12m
0:18m
0:24m
0:30m
0:36m
0:42m
0:48m
0:54m
1:00m
Last Event

'920.92 MB' received (1.55 MB/s)

Edge Counter: Last Event: 18/11/2019 15:51:44.635

Threshold Classificator 1Ch1
Input: Count @ Edge Counter 1Ch_1 600
Level OK / Warning: 420
Level Warning / Alarm: 450
Class: 3
Last Event Warni: 18/11/2019 15:48:44.636
Last Event Alarm: 18/11/2019 15:49:14.636

Min Max Avg Interval 1Ch1
Input: Variables.fGrowSlow @ tcanalyticstest: T 56277
Interval: Minutes 1
Min: 56273
Max: 56279
Avg: 56276
Time Min: 18/11/2019 15:51:12.913
Time Max: 18/11/2019 15:51:19.313
Current Interval: 00:00:00:000

WB-ID: c0bd6e77-e2b6-4e67-a2ac-b927a523d997

Start: 15:41:45.185:000 | End: 15:51:45.185:000 | Pos: 0.00:06:36.000:000 | Time: 15:48:21.185:000 | Date: Monday, November 18, 2019

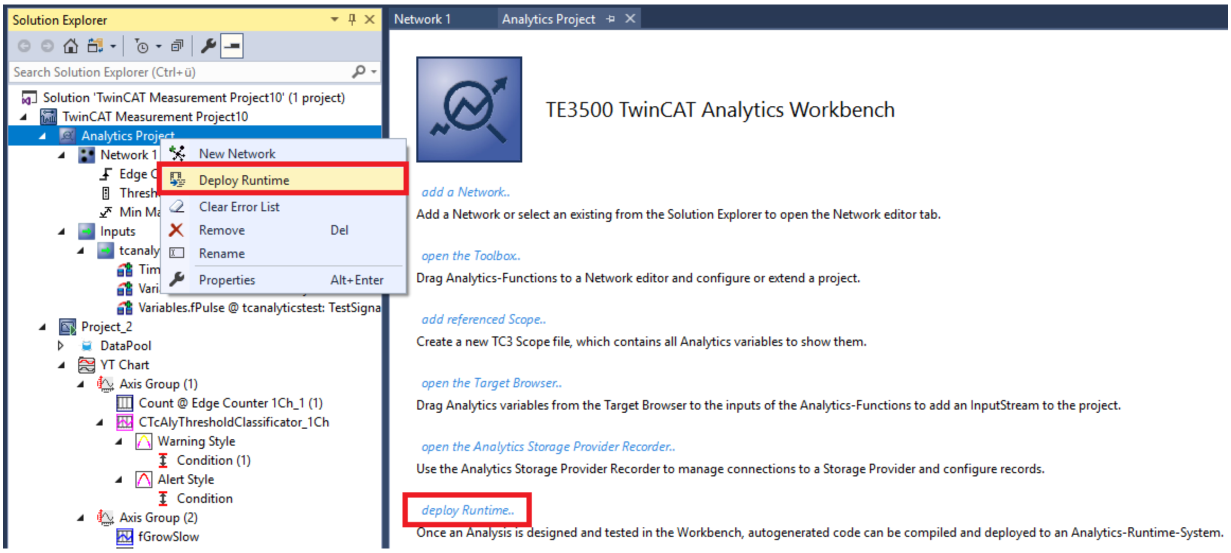
0.00:01:00.000:000 | 0.00:06:36.000:000

⇨ The analysis results can be displayed in the Scope View graphs using drag-and-drop. For example, a mean value can be displayed as a new channel in the view. Timestamps as markers on the X-axes show significant values.

4.5 24h Analytics application

The last major step in the TwinCAT Analytics workflow is the continuous 24-hour machine analysis. It runs in parallel with the machine applications in the field. To make this very easy, the TwinCAT Analytics Workbench can automatically generate PLC code and an HTML5-based dashboard of your Analytics configuration. Both can be downloaded into a TwinCAT Analytics Runtime (TC3 PLC and HMI Server) and provide the same analysis results as the configurator tool in the engineering environment.

✓ First, save your configuration and open the Analytics Deploy Runtime Wizard. This can be done from the context menu in the Analytics Project tree item or from the start page.



TE3500 TwinCAT Analytics Workbench

add a Network..
Add a Network or select an existing from the Solution Explorer to open the Network editor tab.

open the Toolbox..
Drag Analytics-Functions to a Network editor and configure or extend a project.

add referenced Scope..
Create a new TC3 Scope file, which contains all Analytics variables to show them.

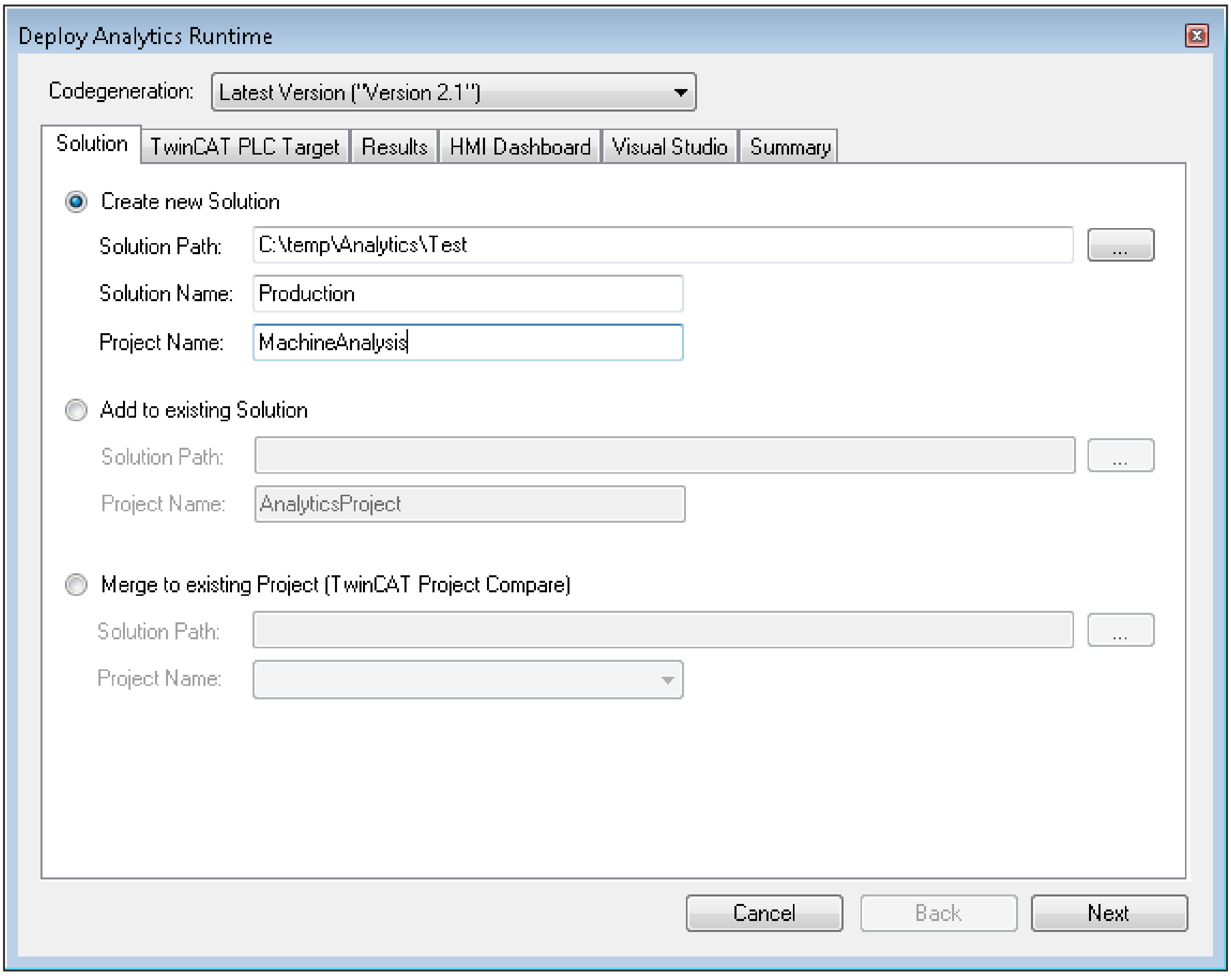
open the Target Browser..
Drag Analytics variables from the Target Browser to the inputs of the Analytics-Functions to add an InputStream to the project.

open the Analytics Storage Provider Recorder..
Use the Analytics Storage Provider Recorder to manage connections to a Storage Provider and configure records.

deploy Runtime..
Once an Analysis is designed and tested in the Workbench, autogenerated code can be compiled and deployed to an Analytics-Runtime-System.

1. When the wizard is open, you can click through some tabs. The first one is called Solution. Here you can decide how your Analytics project should be used in the PLC code: As...
completely new solution.

part of an existing solution.
update of an existing Analytics solution.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

● Create new Solution

Solution Path: C:\temp\Analytics\Test

Solution Name: Production

Project Name: MachineAnalysis

○ Add to existing Solution

Solution Path:

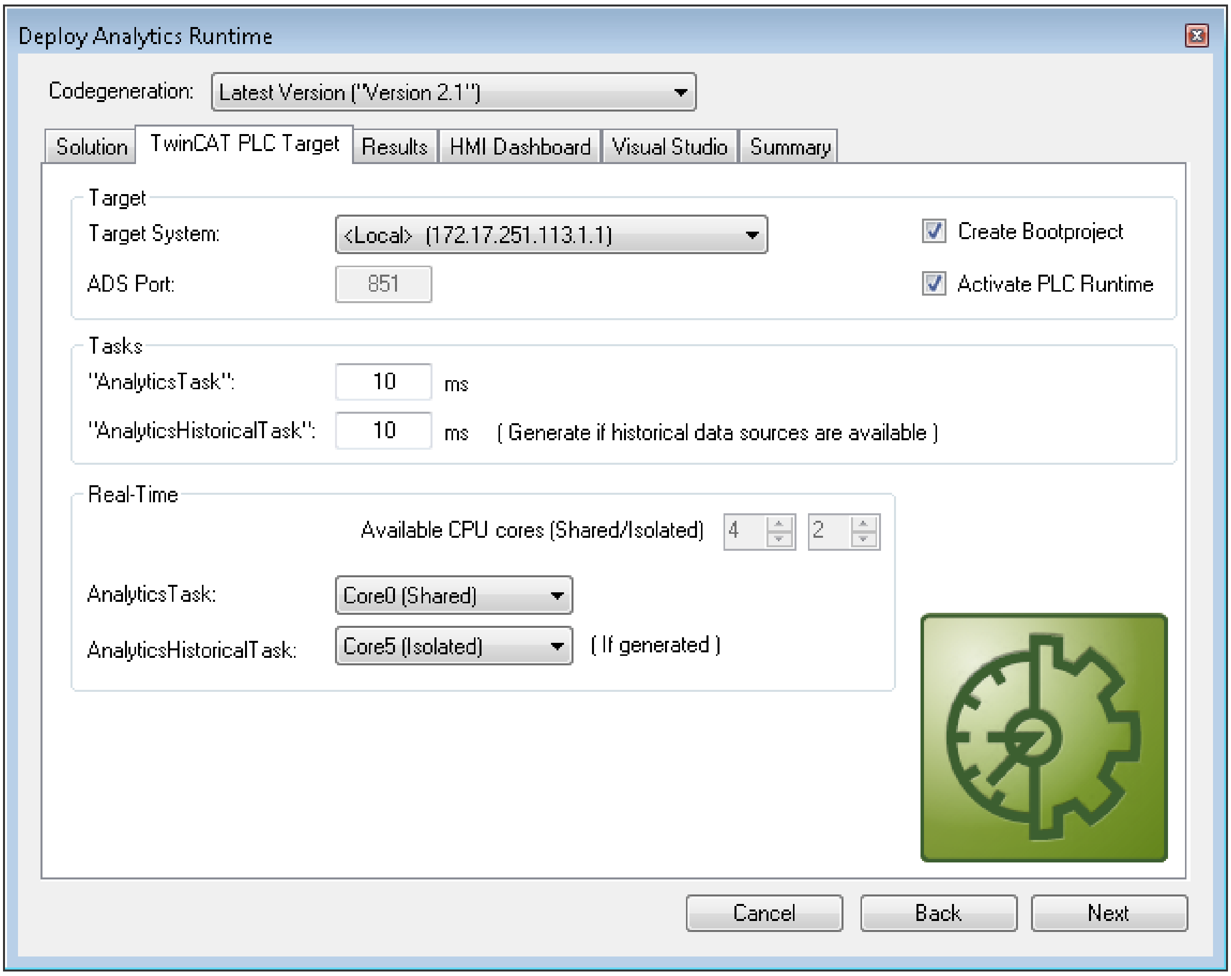
Project Name: AnalyticsProject

○ Merge to existing Project (TwinCAT Project Compare)

Solution Path:

Project Name:

2. In the **TwinCAT PLC Target** tab you can select the ADS target system that runs the TwinCAT Analytics Runtime (TF3550). The created project is immediately executable. For this purpose you can set the Activate PLC Runtime option. In addition, it can be selected that a boot project is created directly.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

Target

Target System: <Local> (172.17.251.113.1.1)

ADS Port: 851

☑ Create Bootproject

☑ Activate PLC Runtime

Tasks

"AnalyticsTask": 10 ms

"AnalyticsHistoricalTask": 10 ms (Generate if historical data sources are available)

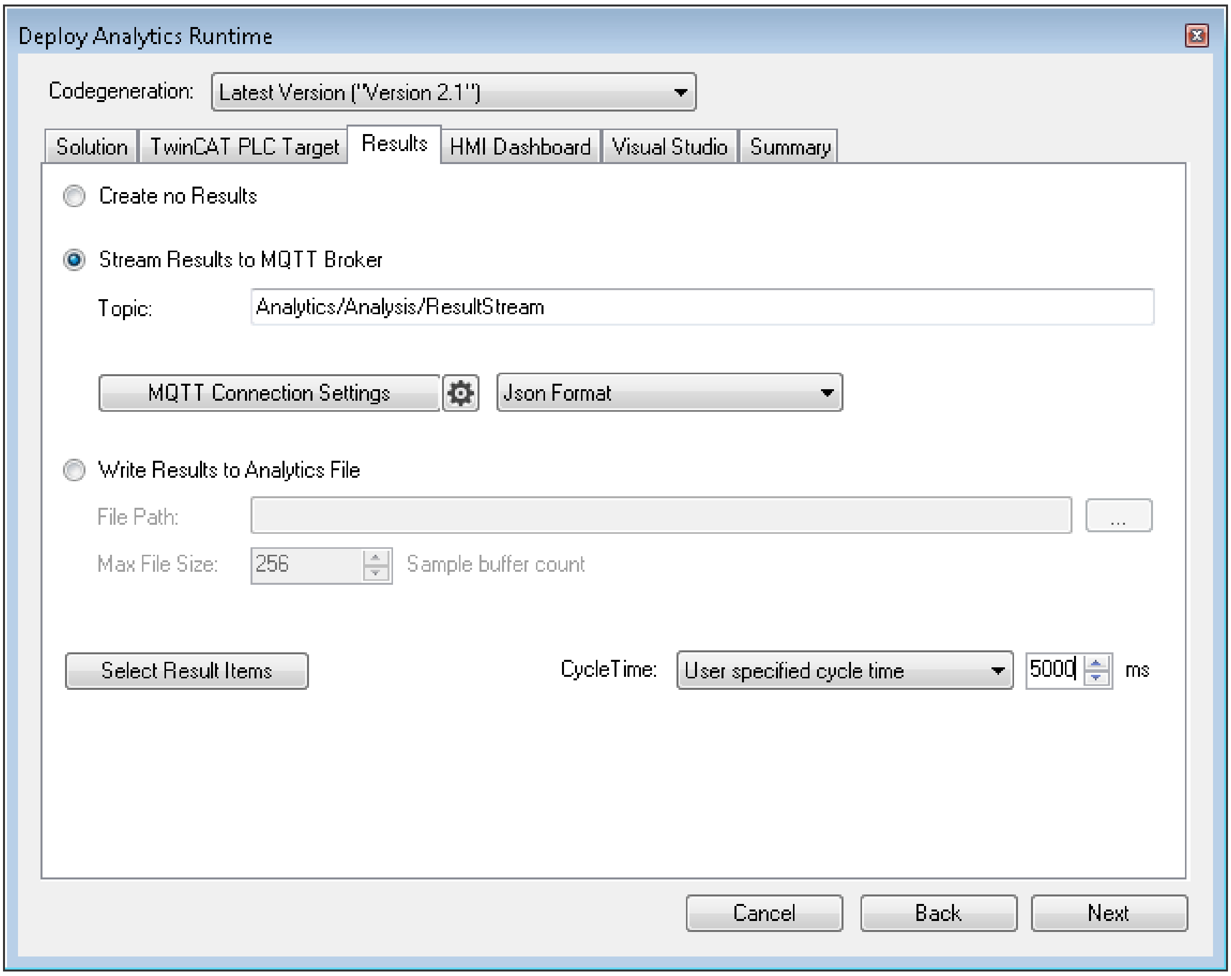
Real-Time

Available CPU cores (Shared/Isolated) 4 2

AnalyticsTask: Core0 (Shared)

AnalyticsHistoricalTask: Core5 (Isolated) (If generated)

3. Especially for virtual machines, it is important to run the project on isolated cores, which is also an option in this tab. The next tab **Results** is needed only if you have selected the **Stream Results** option in the algorithm properties. If you want to send results, you can decide here in which way (locally in a file/ through MQTT) and which format (binary/JSON) this should be done. This is also generated automatically and executed immediately after activation.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

○ Create no Results

● Stream Results to MQTT Broker

Topic: Analytics/Analysis/ResultStream

Json Format

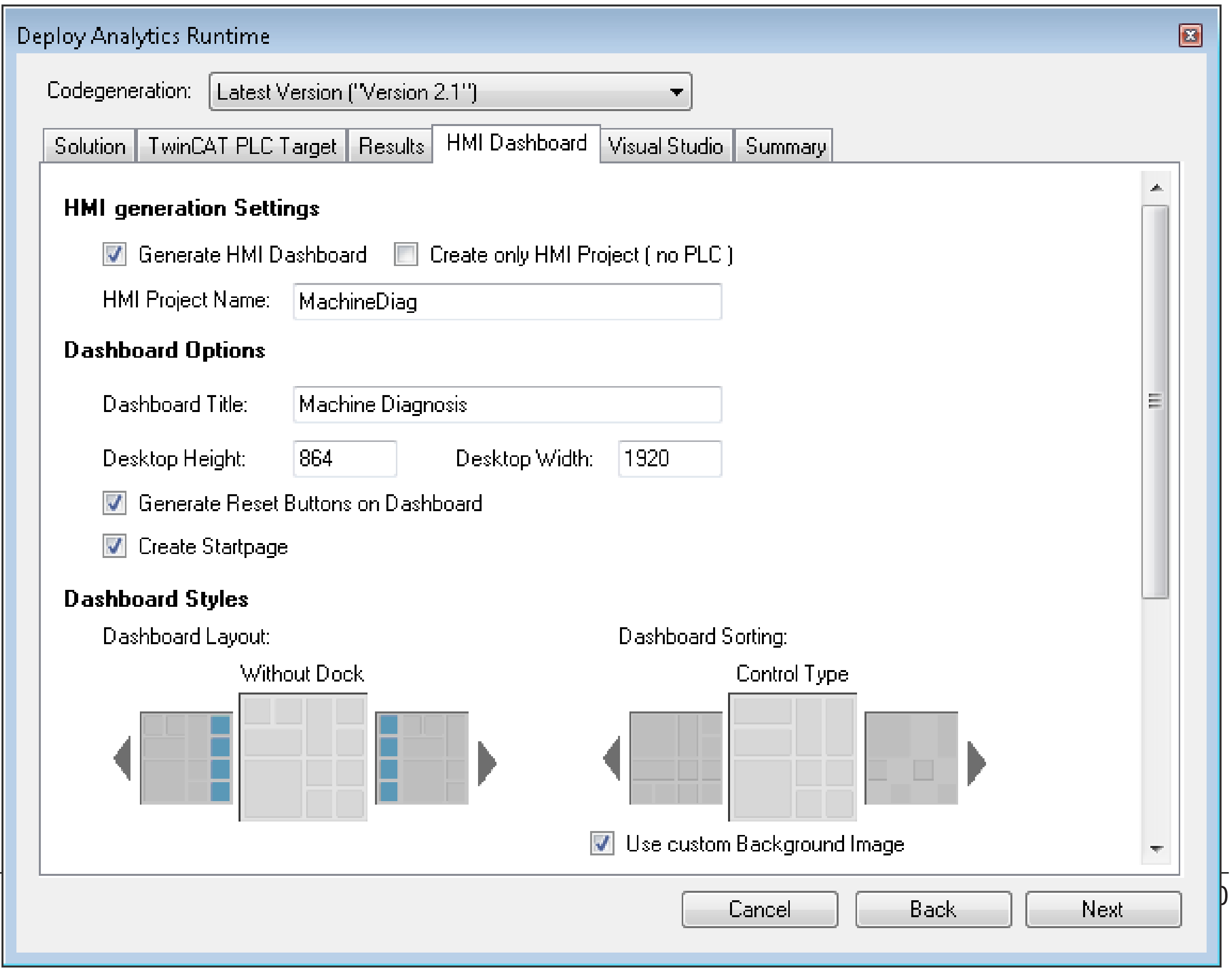
○ Write Results to Analytics File

File Path:

Max File Size: 256 Sample buffer count

CycleTime: User specified cycle time 5000 ms

Downsampling of the results is possible by specifying a cycle time. The next tab is for the **HMI Dashboard**. A prerequisite for the automatic generation of the dashboard is the selection of HMI Controls for the corresponding algorithms whose results are to be displayed in the dashboard.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

HMI generation Settings

☑ Generate HMI Dashboard ☐ Create only HMI Project (no PLC)

HMI Project Name: MachineDiag

Dashboard Options

Dashboard Title: Machine Diagnosis

Desktop Height: 864 Desktop Width: 1920

☑ Generate Reset Buttons on Dashboard

☑ Create Startpage

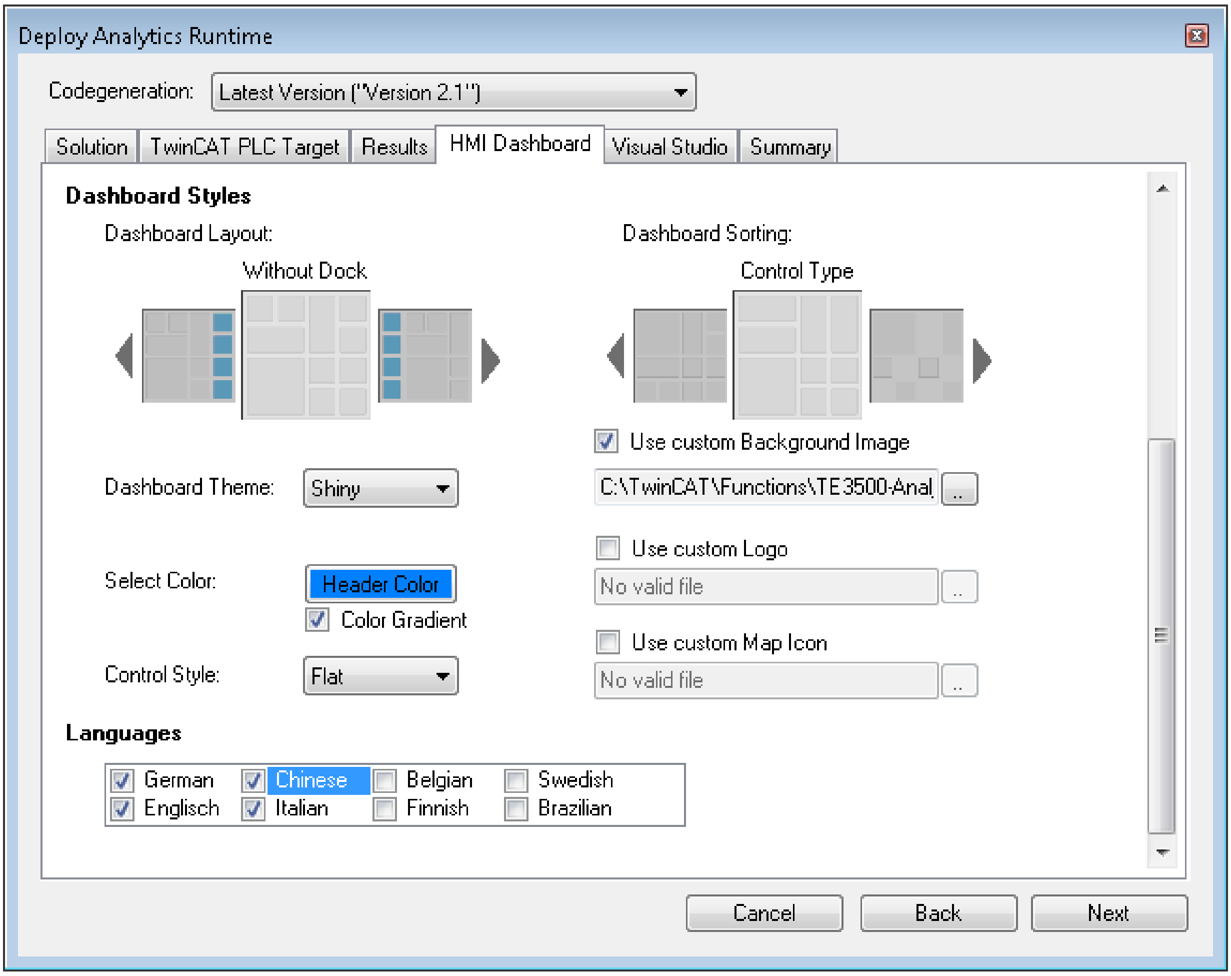
Dashboard Styles

Dashboard Layout: Without Dock

Dashboard Sorting: Control Type

☑ Use custom Background Image

4. You can choose different options for your Analytics Dashboard, such as a start page with a map, layouts, sorting algorithms, custom colors and logos. If you select multiple languages for the Analytics Controls, a language switching menu will also be generated.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

Dashboard Styles

Dashboard Layout: Without Dock

Dashboard Sorting: Control Type

☑ Use custom Background Image

C:\TwinCAT\Functions\TE3500-Anal

Dashboard Theme: Shiny

□ Use custom Logo

No valid file

Select Color: Header Color

☑ Color Gradient

□ Use custom Map Icon

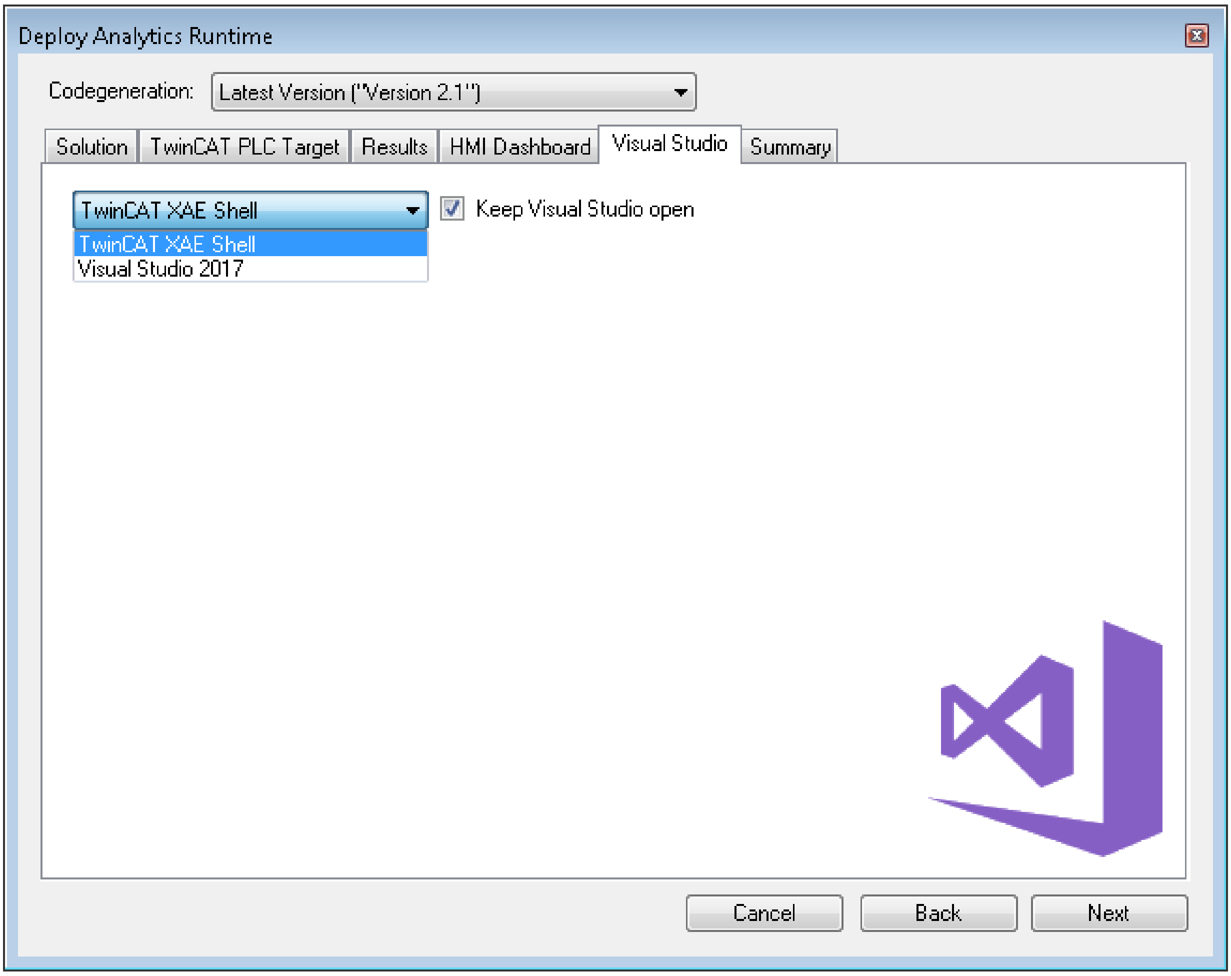
No valid file

Control Style: Flat

Languages

☑ German
☑ Englisch
☑ Chinese
☑ Italian
□ Belgian
□ Finnish
□ Swedish
□ Brazilian

5. Select one of the installed versions of Visual Studio® and, whether the instance should start visibly or just be set up and activated in the background.



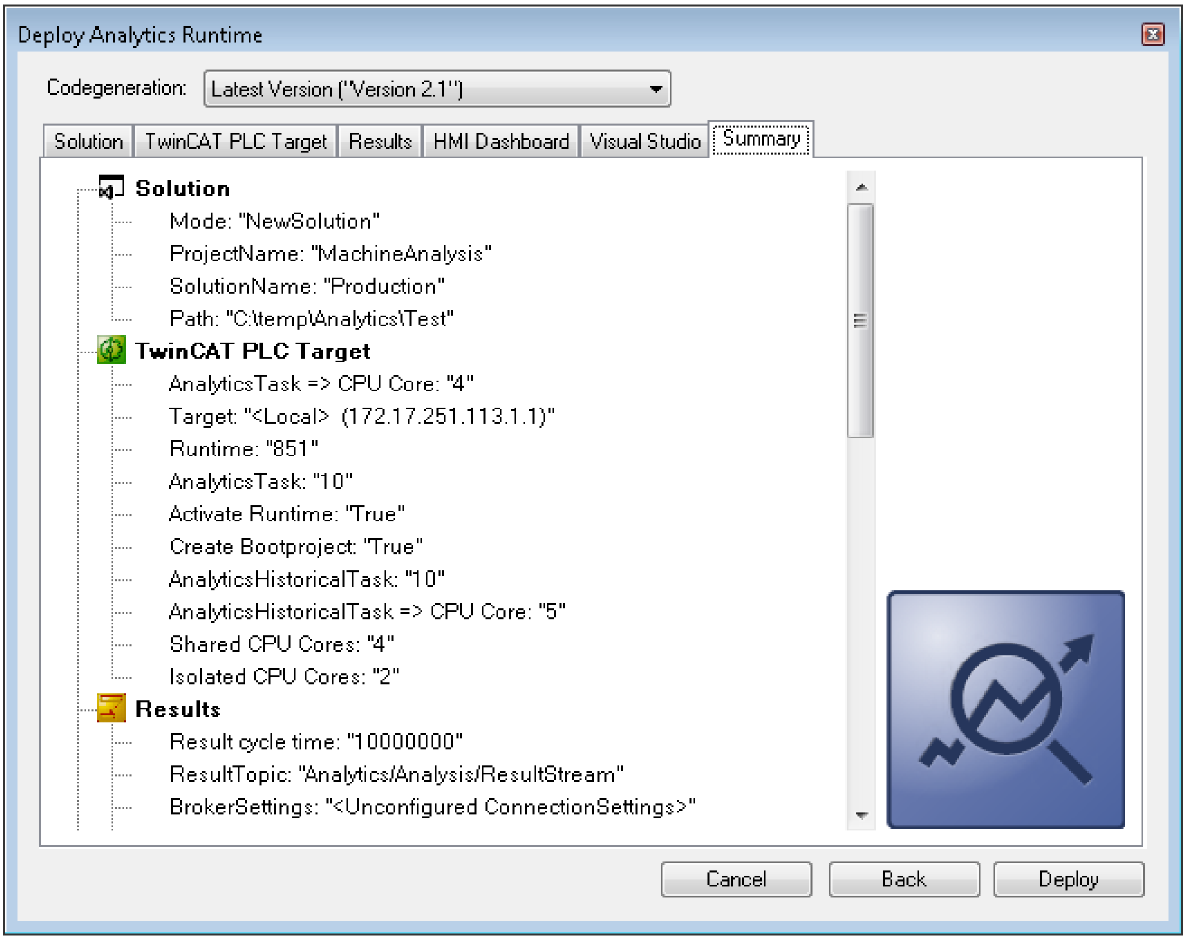
Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

TwinCAT XAE Shell

☑ Keep Visual Studio open

⇨ At last you can find an overview.



Deploy Analytics Runtime

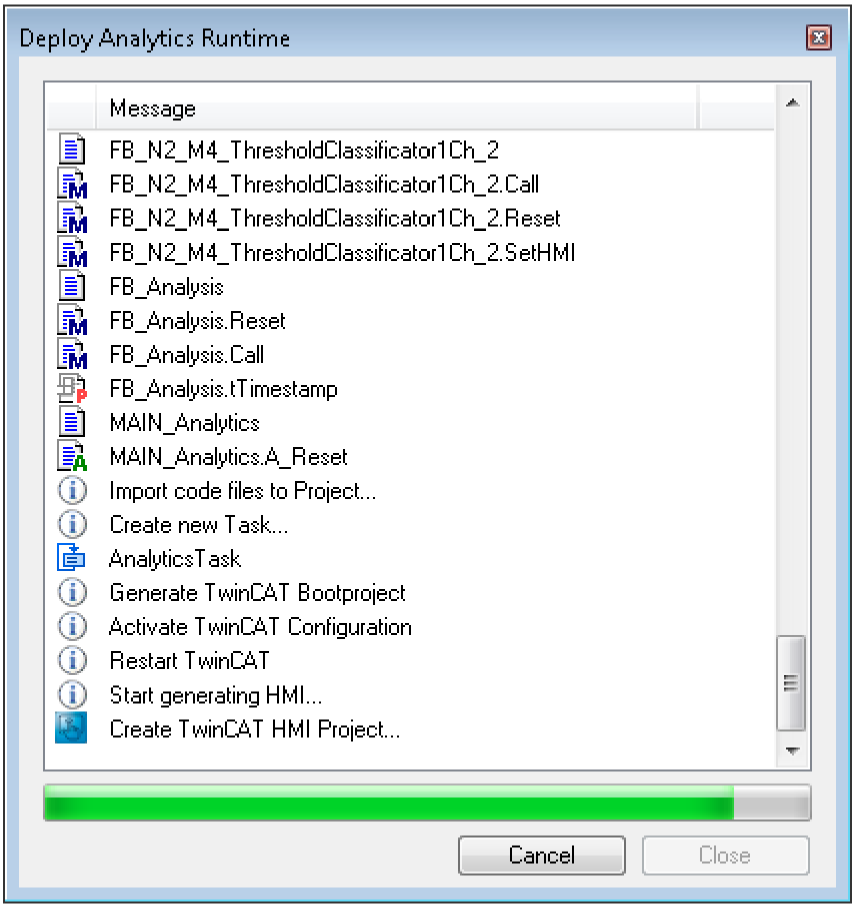
Codegeneration: Latest Version ("Version 2.1")

Solution
- Mode: "NewSolution"
- ProjectName: "MachineAnalysis"
- SolutionName: "Production"
- Path: "C:\temp\Analytics\Test"

TwinCAT PLC Target
- AnalyticsTask => CPU Core: "4"
- Target: "<Local> (172.17.251.113.1.1)"
- Runtime: "851"
- AnalyticsTask: "10"
- Activate Runtime: "True"
- Create Bootproject: "True"
- AnalyticsHistoricalTask: "10"
- AnalyticsHistoricalTask => CPU Core: "5"
- Shared CPU Cores: "4"
- Isolated CPU Cores: "2"

Results
- Result cycle time: "10000000"
- ResultTopic: "Analytics/Analysis/ResultStream"
- BrokerSettings: "<Unconfigured ConnectionSettings>"

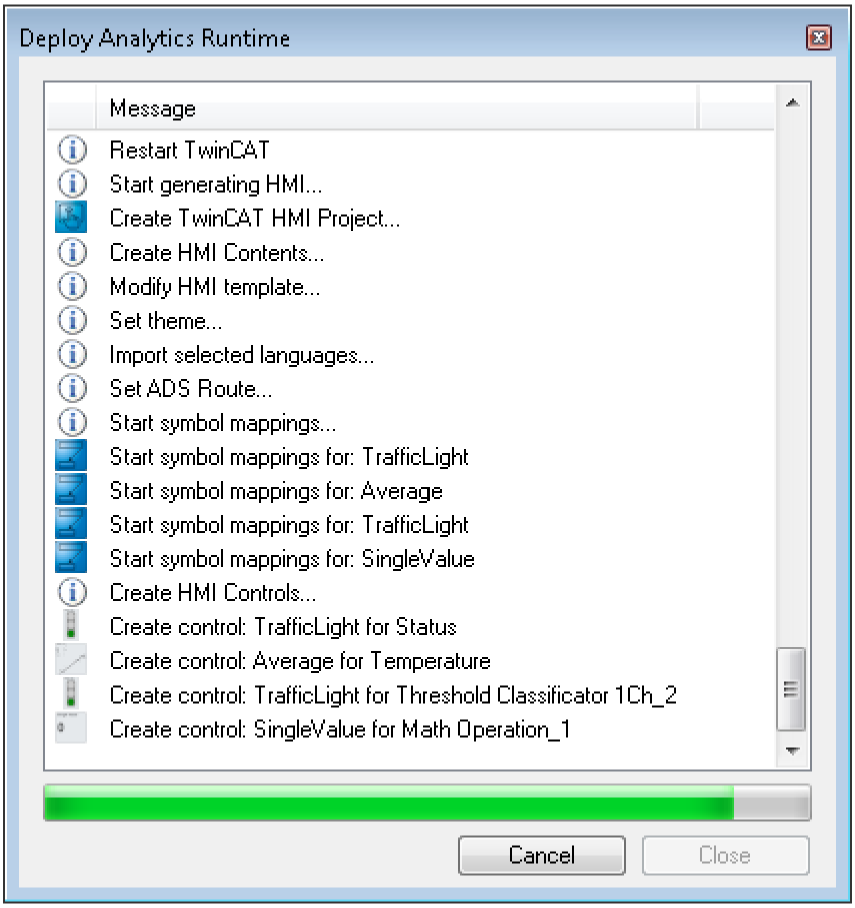
6. Now you can click the **Deploy** button to start the generation process. The PLC project and the HMI dashboard are now generated.



Deploy Analytics Runtime

Message

FB_N2_M4_ThresholdClassificator1Ch_2
FB_N2_M4_ThresholdClassificator1Ch_2.Call
FB_N2_M4_ThresholdClassificator1Ch_2.Reset
FB_N2_M4_ThresholdClassificator1Ch_2.SetHMI
FB_Analysis
FB_Analysis.Reset
FB_Analysis.Call
FB_Analysis.tTimestamp
MAIN_Analytics
MAIN_Analytics.A_Reset
Import code files to Project...
Create new Task...
AnalyticsTask
Generate TwinCAT Bootproject
Activate TwinCAT Configuration
Restart TwinCAT
Start generating HMI...
Create TwinCAT HMI Project...

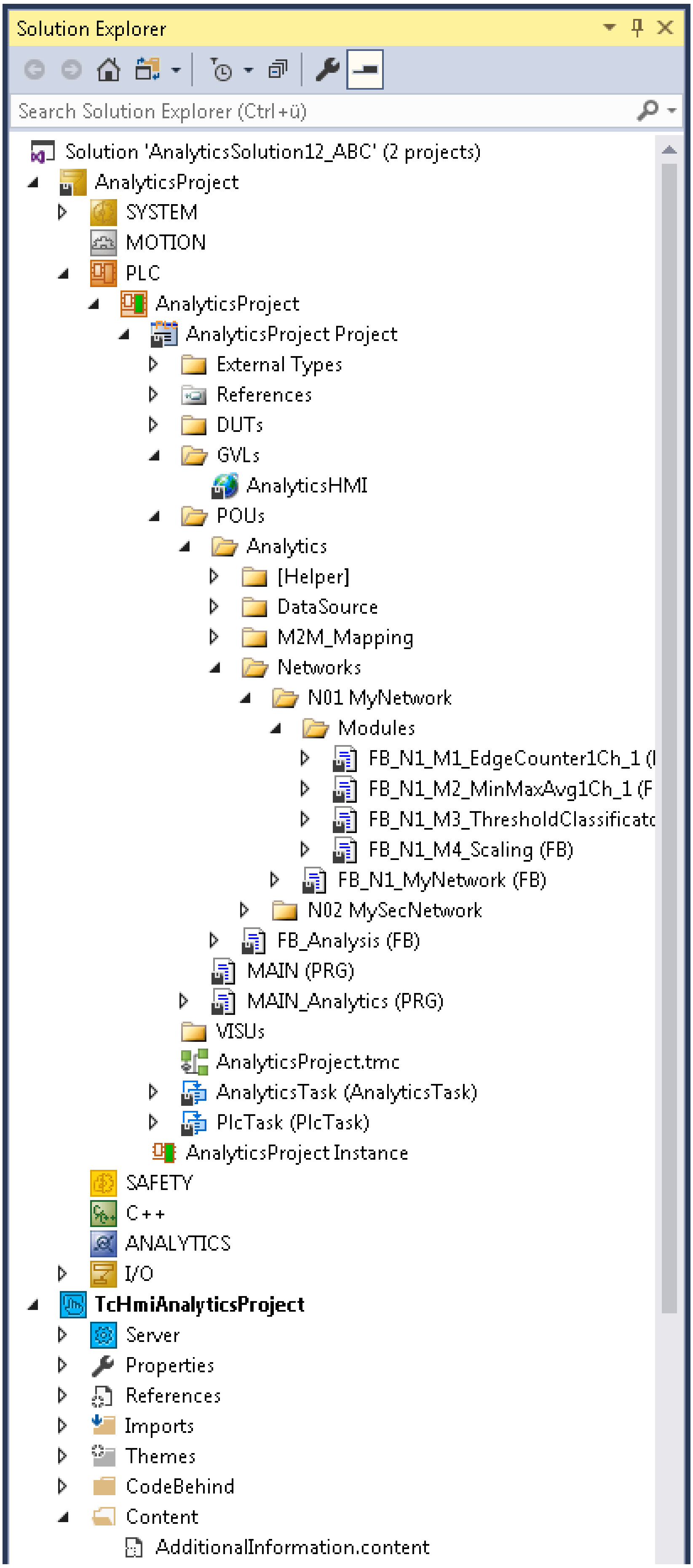


Deploy Analytics Runtime

Message

Restart TwinCAT
Start generating HMI...
Create TwinCAT HMI Project...
Create HMI Contents...
Modify HMI template...
Set theme...
Import selected languages...
Set ADS Route...
Start symbol mappings...
Start symbol mappings for: TrafficLight
Start symbol mappings for: Average
Start symbol mappings for: TrafficLight
Start symbol mappings for: SingleValue
Create HMI Controls...
Create control: TrafficLight for Status
Create control: Average for Temperature
Create control: TrafficLight for Threshold Classificator 1Ch_2
Create control: SingleValue for Math Operation_1

⇨ After the "Deploy Runtime succeeded" message, you will find a new Visual Studio®/XAE shell instance on your desktop. The new Solution and both projects are created.



```
Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'AnalyticsSolution12_ABC' (2 projects)
◢ AnalyticsProject
    ▷ SYSTEM
      MOTION
    ◢ PLC
        ◢ AnalyticsProject
            ◢ AnalyticsProject Project
                ▷ External Types
                ▷ References
                ▷ DUTs
                ◢ GVLs
                      AnalyticsHMI
                ◢ POUs
                    ◢ Analytics
                        ▷ [Helper]
                        ▷ DataSource
                        ▷ M2M_Mapping
                        ◢ Networks
                            ◢ N01 MyNetwork
                                ◢ Modules
                                    ▷ FB_N1_M1_EdgeCounter1Ch_1 (I
                                    ▷ FB_N1_M2_MinMaxAvg1Ch_1 (F
                                    ▷ FB_N1_M3_ThresholdClassificatc
                                    ▷ FB_N1_M4_Scaling (FB)
                                ▷ FB_N1_MyNetwork (FB)
                            ▷ N02 MySecNetwork
                        ▷ FB_Analysis (FB)
                      MAIN (PRG)
                    ▷ MAIN_Analytics (PRG)
                  VISUs
                  AnalyticsProject.tmc
                ▷ AnalyticsTask (AnalyticsTask)
                ▷ PlcTask (PlcTask)
              AnalyticsProject Instance
      SAFETY
      C++
      ANALYTICS
    ▷ I/O
◢ TcHmiAnalyticsProject
    ▷ Server
    ▷ Properties
    ▷ References
    ▷ Imports
    ▷ Themes
    ▷ CodeBehind
    ◢ Content
          AdditionalInformation.content
```

5 Technical introduction

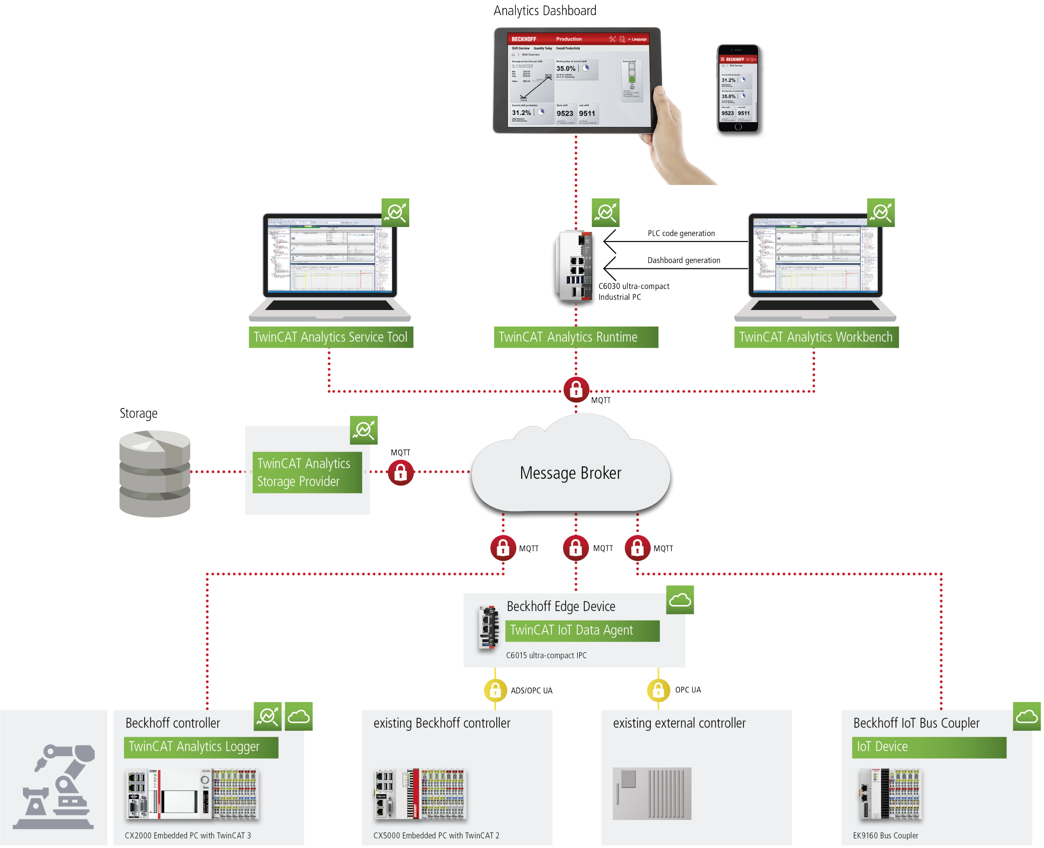
5.1 Basic concept

The following figure shows the basic concept of TwinCAT Analytics from the data source to the Analytics Dashboard based on TwinCAT 3 HMI. The communication in an Analytics scene is realized by the IoT communication protocol MQTT.

Data sources:

Currently there are three different data sources for TwinCAT Analytics. All these sources can communicate with the specific binary data format of TwinCAT Analytics. This format is necessary to achieve high performance.

- TwinCAT 3 control with TF3500 TwinCAT Analytics Logger
- TwinCAT 2, TwinCAT 3 and external control together with a gateway of the TF6720 TwinCAT IoT Data Agent
- All EK9160 IoT Bus Coupler



Storage:

With TwinCAT Analytics it is possible to analyze live and historical data. The TwinCAT Analytics Storage Provider is the interface between native MQTT Message Broker to different stores. As storage TwinCAT Analytics is supporting an Azure Blob store and a Microsoft SQL database. The configuration of the stores is done automatically by the Storage Provider. Thus, it is not necessary to use classic SQL commandos to implement the communication. The user also does not need to setup a special table structure.

Analysis:

For service technicians and machine commissioning

The TE3520 TwinCAT Analytics Service Tool is the perfect tool for experts who like to analyze TwinCAT Analytics data sources. It is integrated into the Microsoft Visual Studio®. The user is able to make his analytics configuration in a graphical configurator choosing from a wide pool of different algorithms. A parallel interaction with the Scope View is also possible. The user is able to find significant values easily by drag and drop from the configurator into the data stream of our Scope View.

For continues 24/7 machine analysis

The TE3500 TwinCAT Analytics Workbench has the same functionality as the Service Tool. In addition, it is possible to automatically generate a PLC code with associated HMI dashboard based on the realized analytics configuration in the configurator. The PLC code is ready to use, so you can start data analysis immediately as in the configurator. But now for 24 hours 7 days per week if necessary. The automatically generated code can be downloaded into the TF3550 TwinCAT Analytics Runtime. Alternatively, a download of the pure PLC project, when created without HMI dashboard, into the Analytics Runtime Base TF3551 (without HMI Server) is possible. Both runtime products are essentially license bundles and can run on a classic IPC or Embedded PC, but also in a virtual machine.

Products:

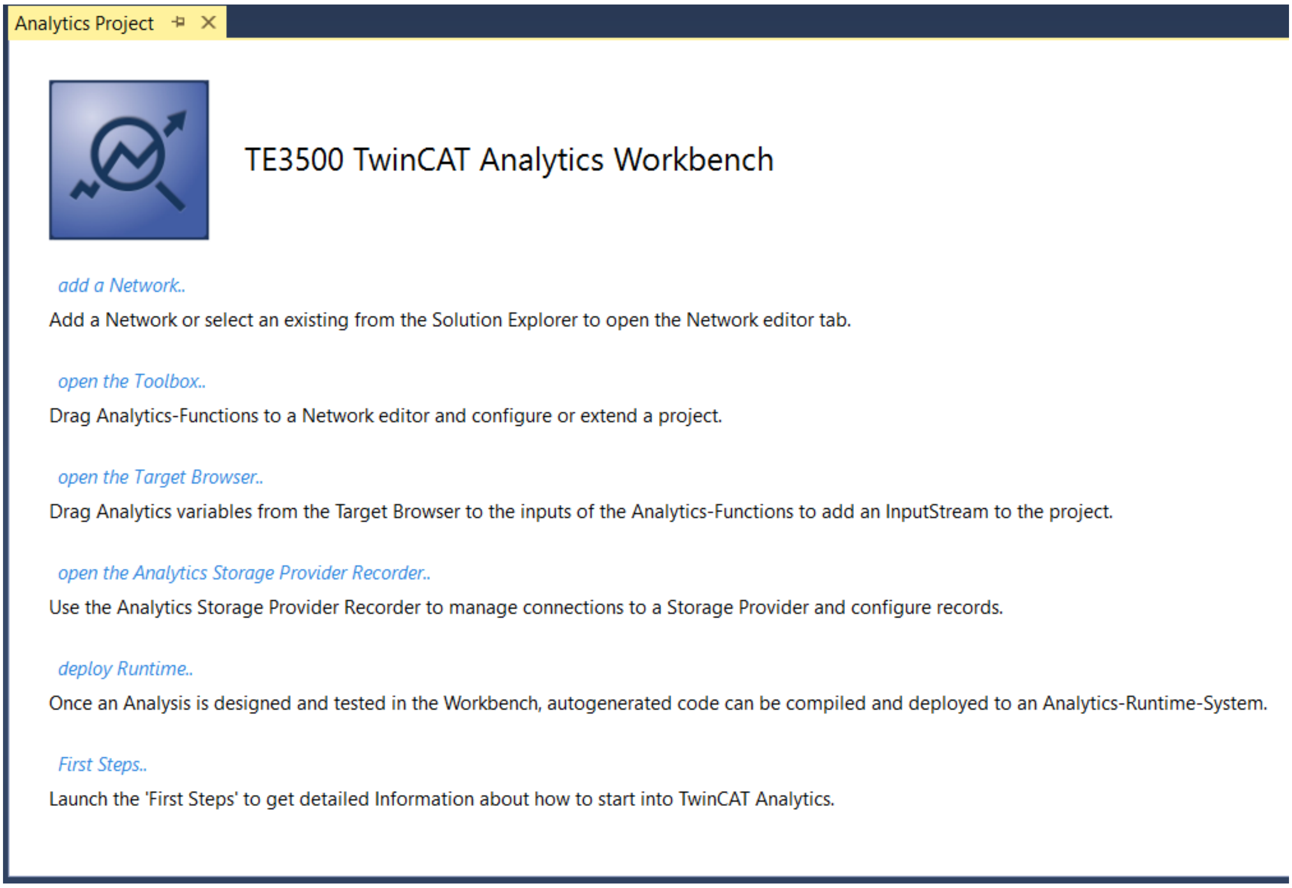
We have different single products in the TwinCAT Analytics Workflow. See therefore the following list with all products.

Product number	Product name
TE3500	Analytics Workbench
TE3520	Analytics Service Tool
TF3500	Analytics Logger
TF3510	Analytics Library
TF3520	Analytics Storage Provider
TF3550	Analytics Runtime - including HMI Server and Client Pack
TF3551	Analytics Runtime Base - without HMI
TF3560	Analytics Controller Pack 4
TF3561	Analytics Controller Pack 8
TF3562	Analytics Controller Pack 16
TF3563	Analytics Controller Pack 32
TF3564	Analytics Controller Pack 64
TF3565	Analytics Controller Pack 128
TF6720	IoT Data Agent
EK9160	IoT Coupler

The TwinCAT Analytics Service Tool can be used as a kind of Scope ++ via an ADS channel. This is automatically the most sensible minimum configuration from the basic concept shown. The Analytics Logger can also be used to decouple data collection and analysis. It can store data locally on the machine computer. The data can be evaluated via the service tool. If you want to organize data storage centrally rather than decentrally, you can use the Analytics Storage Provider via MQTT. The data sources available here are TF3500/TF6720/EK9160. Also possible is just to use the TF3510 Analytics Library in a TwinCAT system.

6 Configuration

If you want to create an Analytics configuration, you will start at the Analytics Project start page, which you can see in the following screenshot.



Analytics Project

TE3500 TwinCAT Analytics Workbench

add a Network..

Add a Network or select an existing from the Solution Explorer to open the Network editor tab.

open the Toolbox..

Drag Analytics-Functions to a Network editor and configure or extend a project.

open the Target Browser..

Drag Analytics variables from the Target Browser to the inputs of the Analytics-Functions to add an InputStream to the project.

open the Analytics Storage Provider Recorder..

Use the Analytics Storage Provider Recorder to manage connections to a Storage Provider and configure records.

deploy Runtime..

Once an Analysis is designed and tested in the Workbench, autogenerated code can be compiled and deployed to an Analytics-Runtime-System.

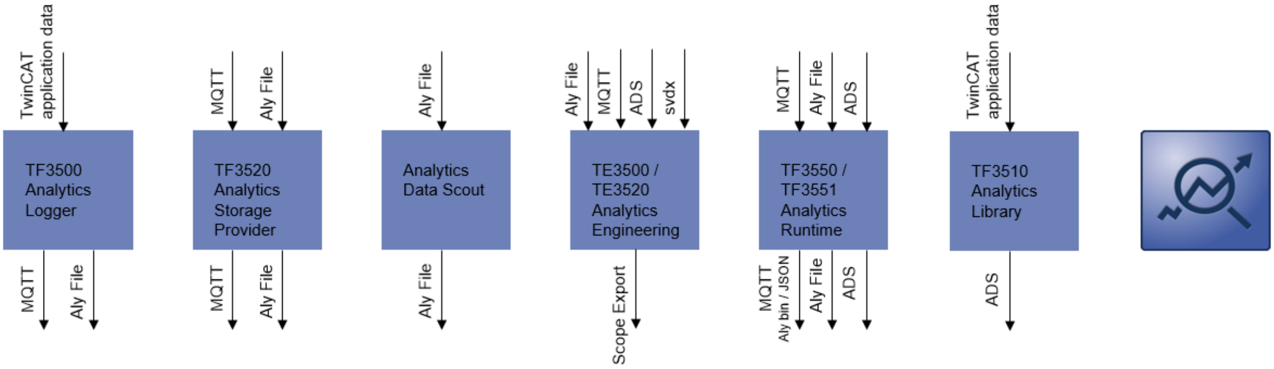
First Steps..

Launch the 'First Steps' to get detailed Information about how to start into TwinCAT Analytics.

On this start page you have several options to continue. These options are described below under the respective option.

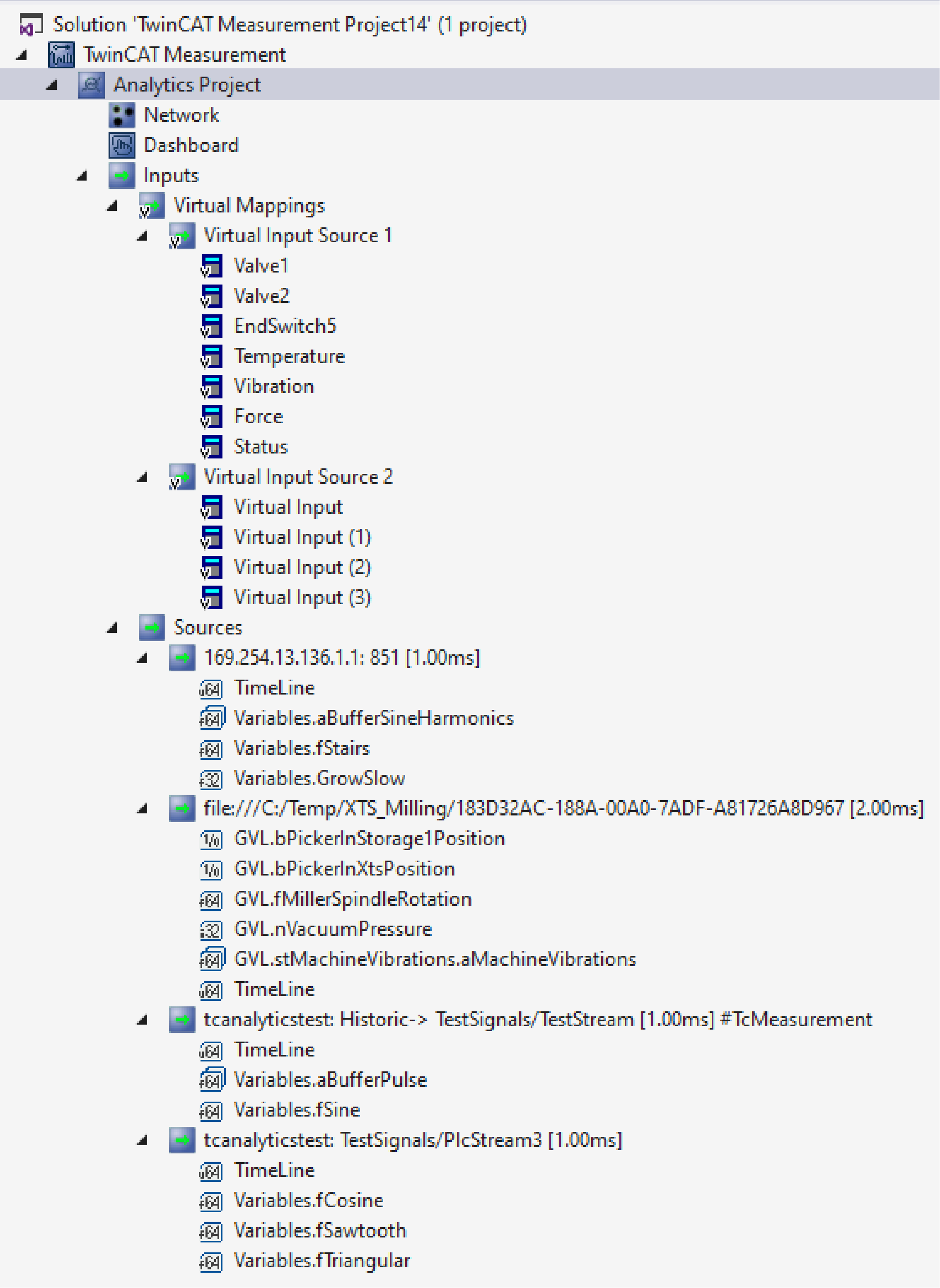
6.1 Data Source

There are many data sources for the TwinCAT Analytics products. Likewise, there are many output formats, thus ensuring a seamless transition from product to product and from tool to tool. The following graphic is intended to provide a general overview.



The most universal are the Analytics binary formats in the form of the Analytics File and an MQTT stream.

In Engineering itself, you can see the data sources in the Solution Explorer in the Inputs area divided into Virtual Input Source and Source. The Source embodies the actual data source as shown in the diagram above. For the engineering products TE3500 and TE3520 these are MQTT as live and historical stream, Analytics File and ADS.



Solution 'TwinCAT Measurement Project14' (1 project)
- TwinCAT Measurement
  - Analytics Project
    - Network
    - Dashboard
    - Inputs
      - Virtual Mappings
        - Virtual Input Source 1
          - Valve1
          - Valve2
          - EndSwitch5
          - Temperature
          - Vibration
          - Force
          - Status
        - Virtual Input Source 2
          - Virtual Input
          - Virtual Input (1)
          - Virtual Input (2)
          - Virtual Input (3)
      - Sources
        - 169.254.13.136.1.1: 851 [1.00ms]
          - TimeLine
          - Variables.aBufferSineHarmonics
          - Variables.fStairs
          - Variables.GrowSlow
        - file:///C:/Temp/XTS_Milling/183D32AC-188A-00A0-7ADF-A81726A8D967 [2.00ms]
          - GVL.bPickerInStorage1Position
          - GVL.bPickerInXtsPosition
          - GVL.fMillerSpindleRotation
          - GVL.nVacuumPressure
          - GVL.stMachineVibrations.aMachineVibrations
          - TimeLine
        - tcanalyticstest: Historic-> TestSignals/TestStream [1.00ms] #TcMeasurement
          - TimeLine
          - Variables.aBufferPulse
          - Variables.fSine
        - tcanalyticstest: TestSignals/PlcStream3 [1.00ms]
          - TimeLine
          - Variables.fCosine
          - Variables.fSawtooth
          - Variables.fTriangular

The Virtual Input Source is an abstract mapping of the actual source. The Virtual Input Source is used for linking at the inputs of the algorithms. It is very convenient that through this mapping the actual data source can be exchanged very quickly and conveniently without having to re-link the algorithms.

Configurations for different sources can be created at the Virtual Input Sources. A configuration contains the mapping of the individual symbols of a source with a virtual input. It is also possible to create different configurations for a source, where the mapping of the symbols differs with a Virtual Input.

When adding sources and replacing them, you can optionally use the Source Wizard, which is explained on the following pages.

6.1.1 Wizard

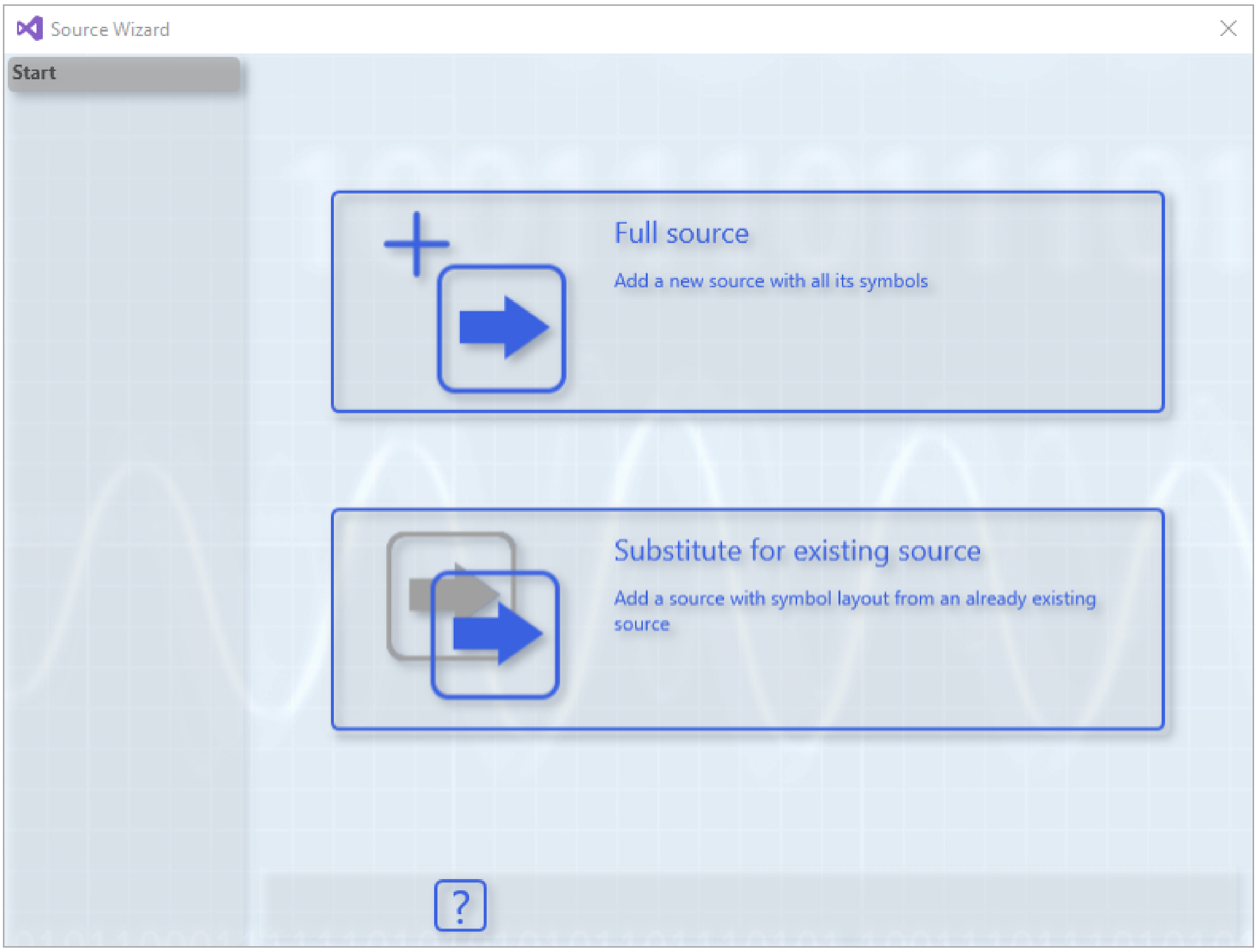
The Source Wizard can be used to add new data sources. This window provides the option to add to entire data sources or find a replacement for an existing data source.

The TwinCAT Target Browser must be installed in order to use the Source Wizard.

Pages:

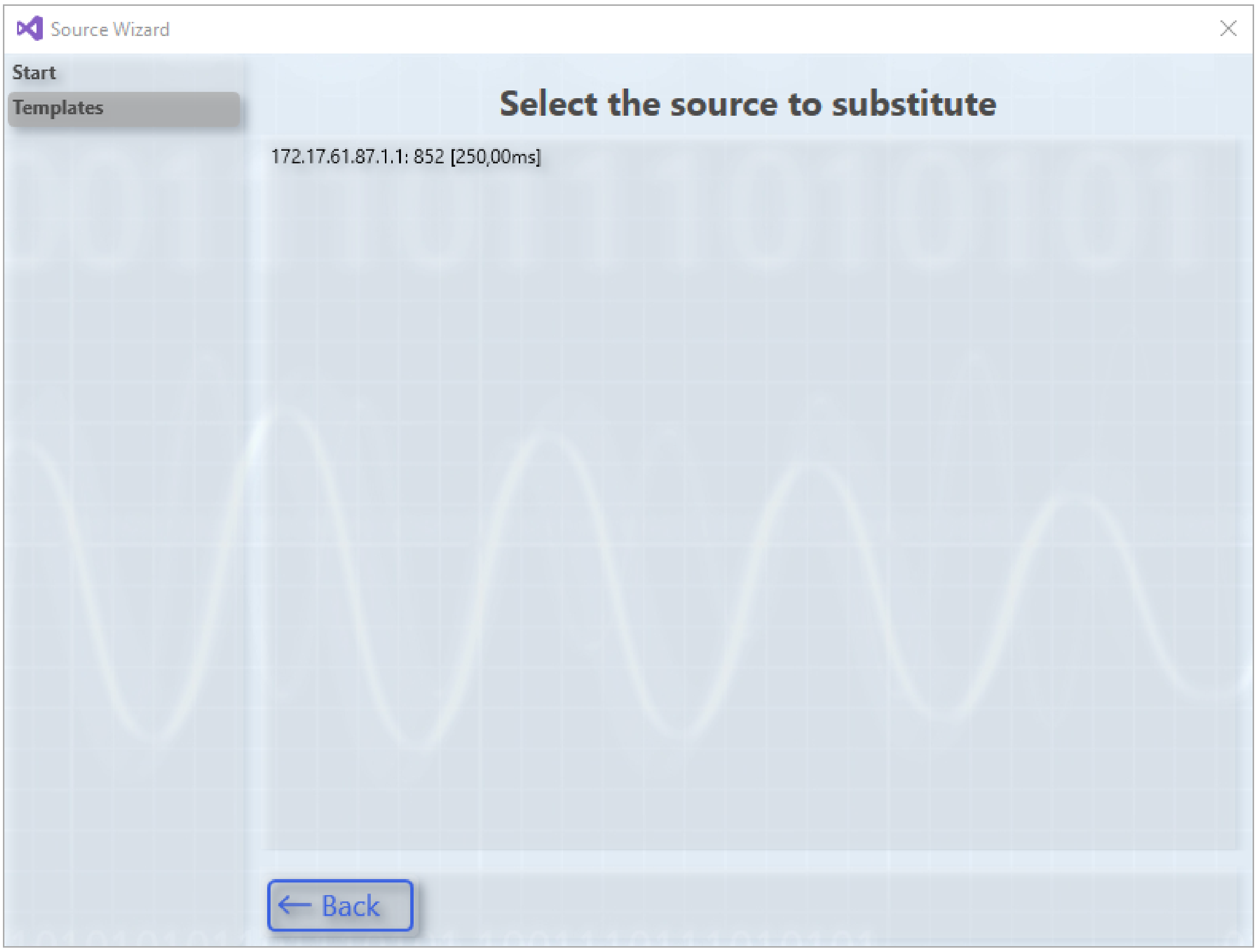
The Source Wizard is divided into several pages. The pages already shown are listed on the left-hand side. Previously opened pages can be displayed again via this list or the **Back** and **Next** buttons.

Start


Source Wizard
Start
Full source
Add a new source with all its symbols
Substitute for existing source
Add a source with symbol layout from an already existing source
?

On this page, you can choose whether all symbols of a source are to be added or only symbols that were sought on the basis of an existing source. The option to replace a source can only be selected if a source already exists in the project.

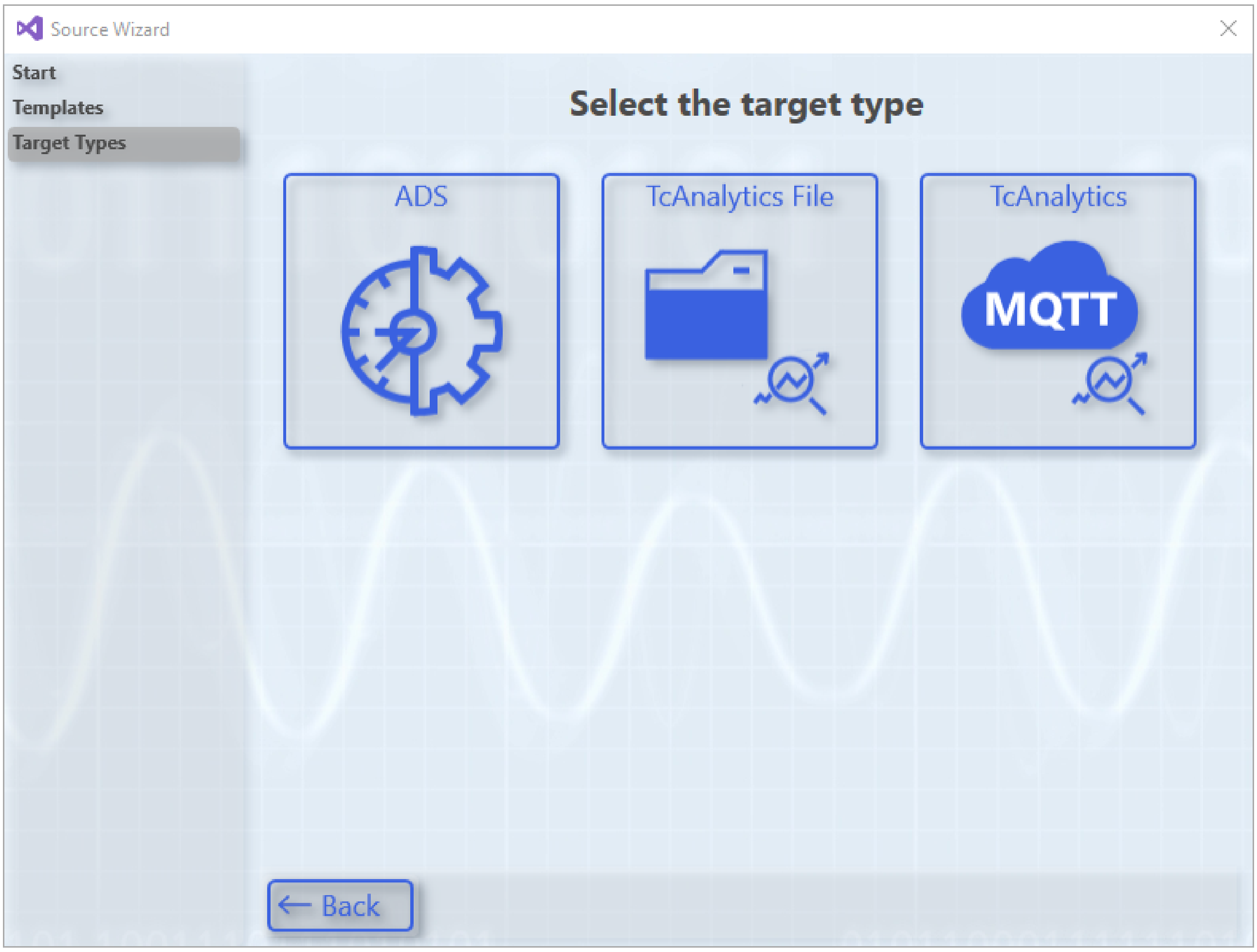
Templates



172.17.61.87.1.1: 852 [250,00ms]

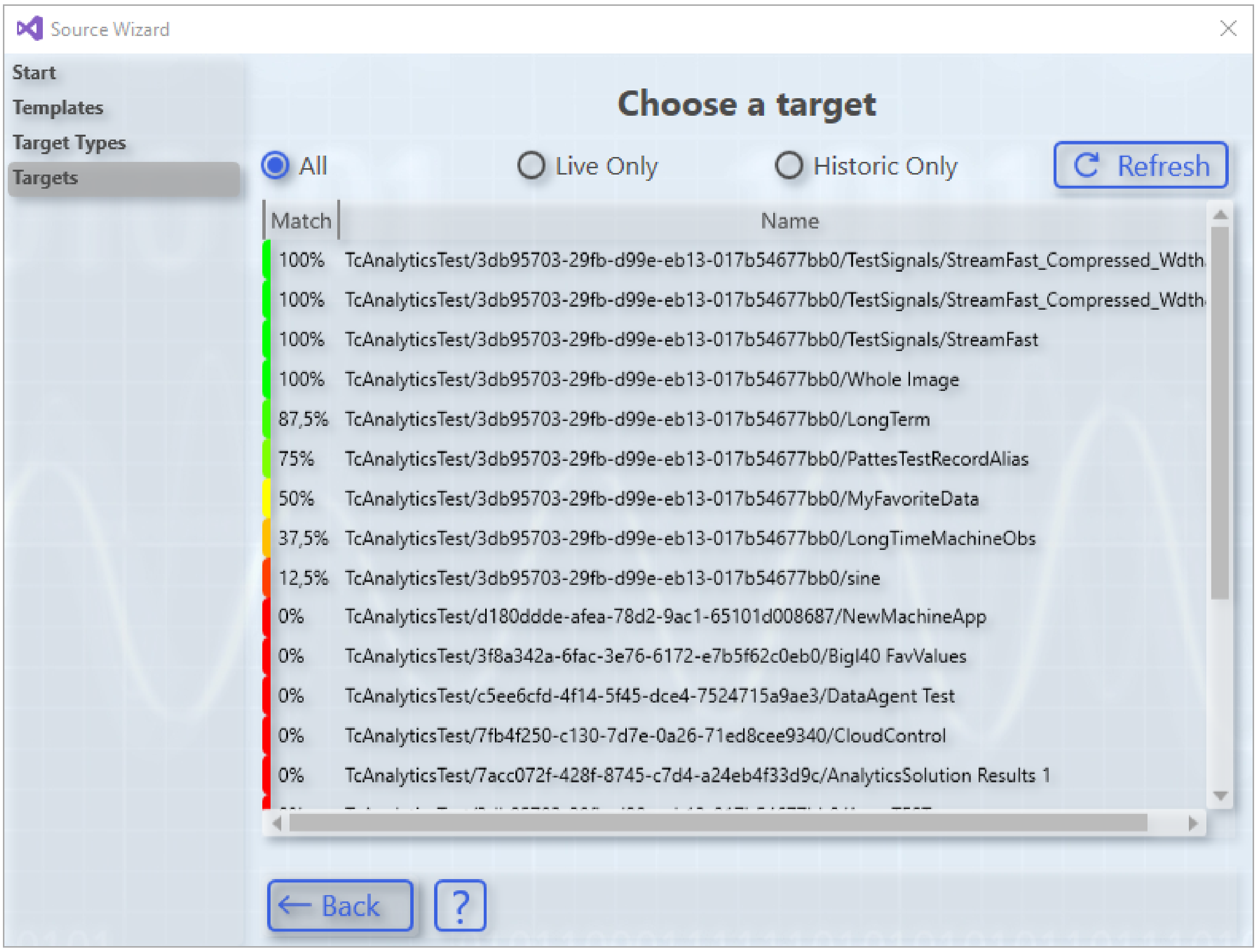
This page is only displayed if the Replace option has been selected on the Start page. Already existing sources of the project are displayed here. To continue, a source must be selected that will then serve as a template for the new one.

Source types



This page displays different source types, such as ADS or Analytics File. The number of types is different to the extensions installed in the Target Browser that are compatible with the Source Wizard.

Sources



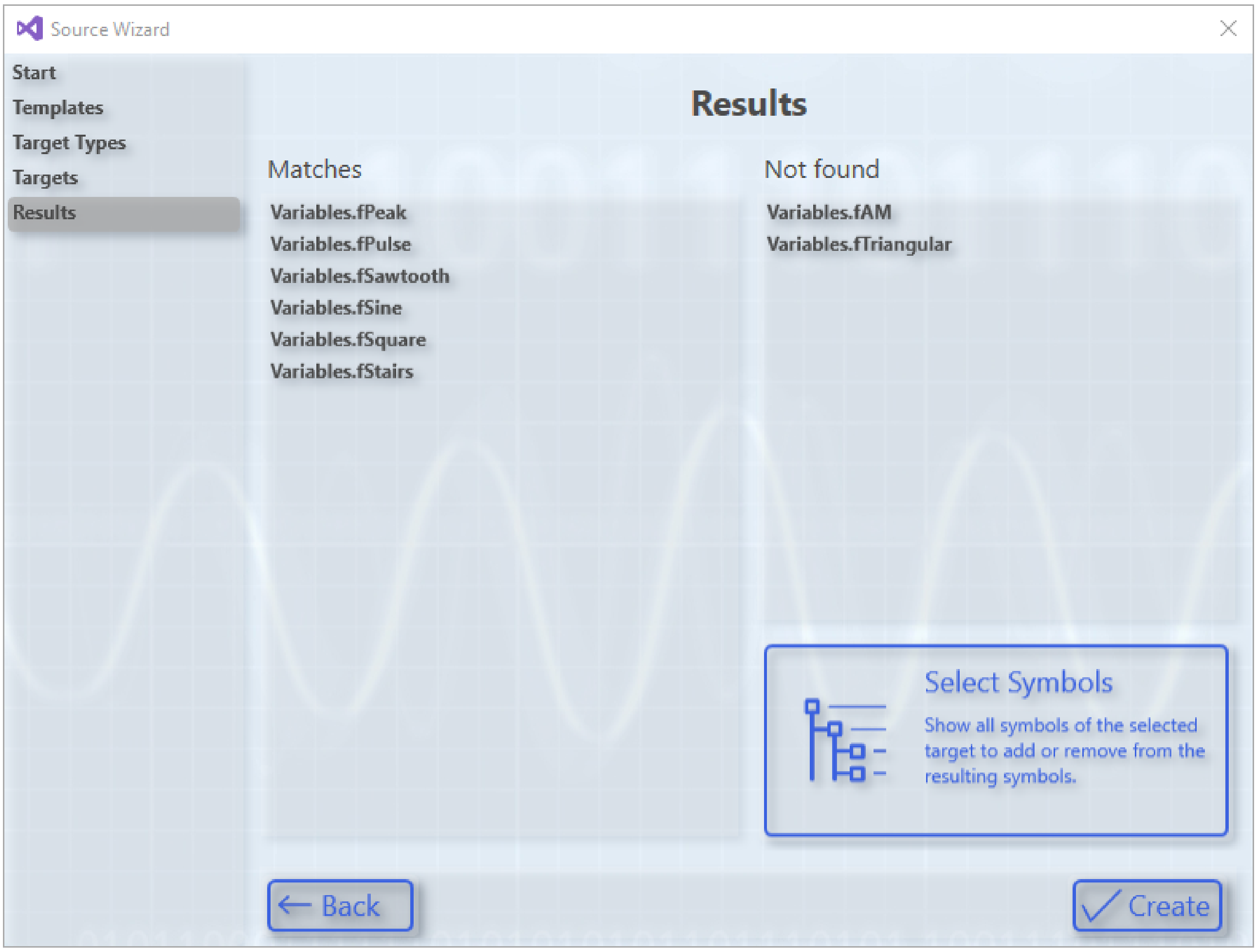
Choose a target

● All ○ Live Only ○ Historic Only

Match	Name
100%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/TestSignals/StreamFast_Compressed_Wdth
100%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/TestSignals/StreamFast_Compressed_Wdth
100%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/TestSignals/StreamFast
100%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/Whole Image
87,5%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/LongTerm
75%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/PattesTestRecordAlias
50%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/MyFavoriteData
37,5%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/LongTimeMachineObs
12,5%	TcAnalyticsTest/3db95703-29fb-d99e-eb13-017b54677bb0/sine
0%	TcAnalyticsTest/d180ddde-afea-78d2-9ac1-65101d008687/NewMachineApp
0%	TcAnalyticsTest/3f8a342a-6fac-3e76-6172-e7b5f62c0eb0/Bigl40 FavValues
0%	TcAnalyticsTest/c5ee6cfd-4f14-5f45-dce4-7524715a9ae3/DataAgent Test
0%	TcAnalyticsTest/7fb4f250-c130-7d7e-0a26-71ed8cee9340/CloudControl
0%	TcAnalyticsTest/7acc072f-428f-8745-c7d4-a24eb4f33d9c/AnalyticsSolution Results 1

After selection of a type, its available sources are displayed. In order to find additional sources, new sources must be stored in the Target Browser. If the Replace option is selected on the Start page, the correlation with the template selected on the templates page is displayed in addition to the name of the sources. The correlation indicates how many of the symbols from the template have been found. In addition, the list can be filtered so that only live or only historical sources are displayed, unless there are only historical or only live sources.

Results



Results

Matches

Variables.fPeak
Variables.fPulse
Variables.fSawtooth
Variables.fSine
Variables.fSquare
Variables.fStairs

Not found

Variables.fAM
Variables.fTriangular

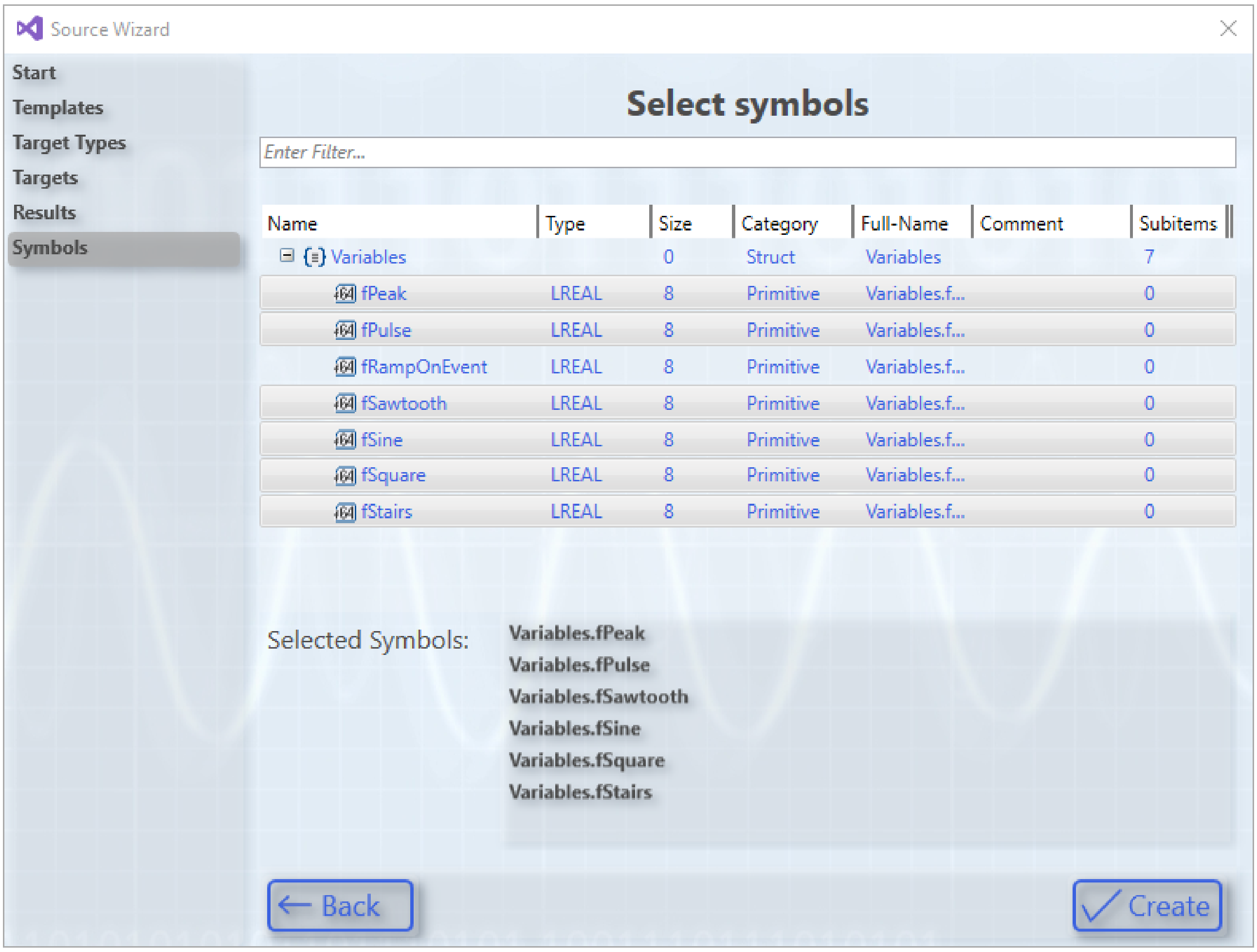
Select Symbols

Show all symbols of the selected target to add or remove from the resulting symbols.

Result page when using Replace

The Result page lists all symbols that would be added. It also displays the icons of the templates that were not found (this list is omitted if the option to add whole sources has been selected on the Start page). Click **Create** to close the wizard. All symbols in the "Matches" list are added to the project. If the selected symbols do not meet the requirements, click "Select Symbols", which will open the Symbols page.

Symbols



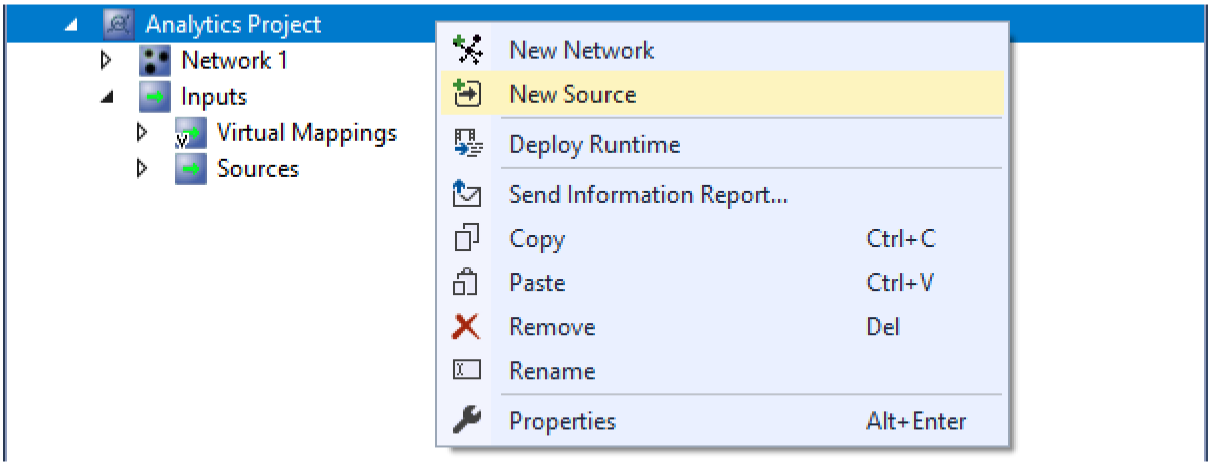
Name	Type	Size	Category	Full-Name	Comment	Subitems
⊟ {≡} Variables		0	Struct	Variables		7
fPeak	LREAL	8	Primitive	Variables.f...		0
fPulse	LREAL	8	Primitive	Variables.f...		0
fRampOnEvent	LREAL	8	Primitive	Variables.f...		0
fSawtooth	LREAL	8	Primitive	Variables.f...		0
fSine	LREAL	8	Primitive	Variables.f...		0
fSquare	LREAL	8	Primitive	Variables.f...		0
fStairs	LREAL	8	Primitive	Variables.f...		0

Selected Symbols:
Variables.fPeak
Variables.fPulse
Variables.fSawtooth
Variables.fSine
Variables.fSquare
Variables.fStairs

If this page is opened, all symbols of the selected source are listed, with the symbols that are also listed as found on the Results page already highlighted. In addition, all currently highlighted symbols are listed below the Tree symbol. Here, you can now select the desired symbols or deselect unnecessary symbols. Clicking the “Create” button terminates the wizard and adds the selected symbols to the project.

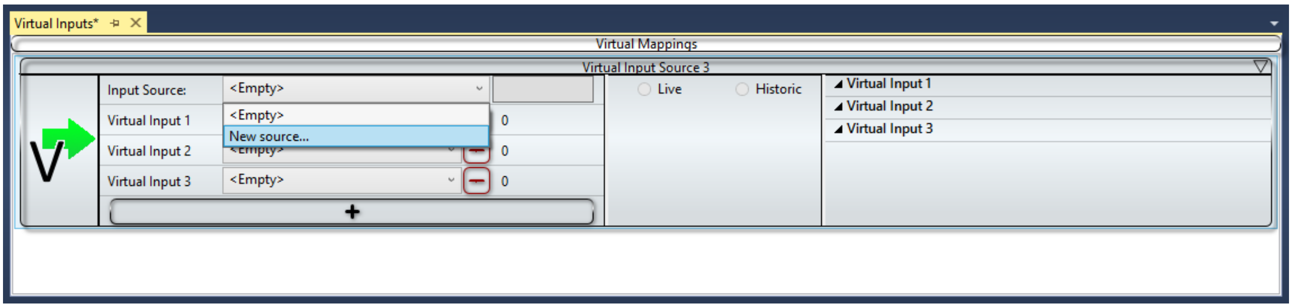
Call:

Context menu



Right-clicking the Analytics Project or a node below it in Solution Explorer opens the context menu, where the entry **New source** can be found. If this is selected, the wizard starts. Upon successful completion, the desired source is added to the project.

Virtual source



In a module for a virtual source, the entry **New source...** can be found in the **Input Source** combo box. If this is selected, the wizard starts. This way to open the wizard does not start at the Start page, but on the Templates page, where the currently selected template is the source of the virtual source. Upon successful completion, the new source is selected in the virtual source and the symbols can be assigned to the virtual inputs in selection mode.

Query

If a symbol is added to a function by drag and drop and this symbol does not belong to any source in the project, then you can choose between three options. A new virtual source can be added and, in order to use the source at the same time as the other sources, a source from an already existing virtual source can be changed or the source wizard can be opened. When the wizard is opened, there is a different action, depending on the selection on the Start page, which is performed after the wizard is completed. If an entire source has been added, a virtual source is created in addition to the source so that it can be used with others at the same time. If Replace has been selected, the new source is added and selected in a virtual source. The symbols can then be assigned to the virtual inputs in selection mode.

6.2 Data Scout

The TwinCAT Analytics Data Scout is a very important tool in the TwinCAT Analytics processing flow. It is used for data viewing and can manipulate data or its recordings. For example, different recordings can be merged into one large recording or unneeded data sections can be removed. The artificially created new image can be used as a data source for the actual analysis. The Data Scout is only responsible for processing data that has already been recorded. The input format is Analytics File, which is also the output format. The engineering tool is available with TE3500 Analytics Workbench and TE3520 Analytics Service Tool.

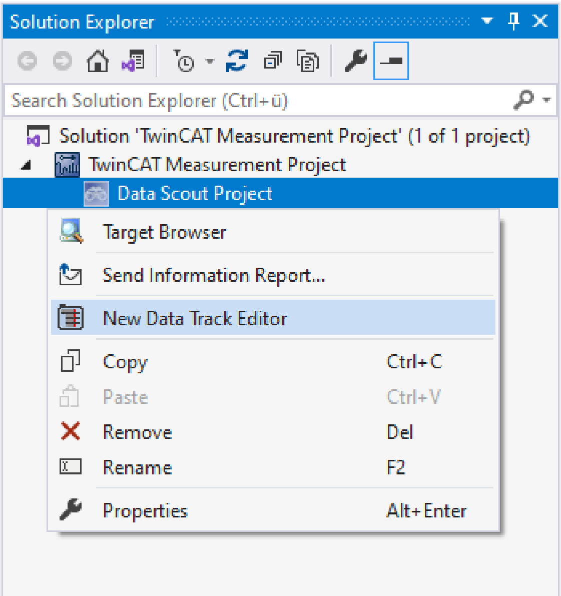
The Data Scout project can be created in the Engineering environment below the Measurement project and is explained in more detail on the following pages.

6.2.1 Data Track Editor

The Data Track Editor is part of the Analytics Data Scout. It can be used to view data, merge data from different sources and to cut out parts of the data or whole symbols.

Create

A new track editor can be added to an Analytics Data Scout by clicking New Data Track Editor in the context menu of a Data Scout project.



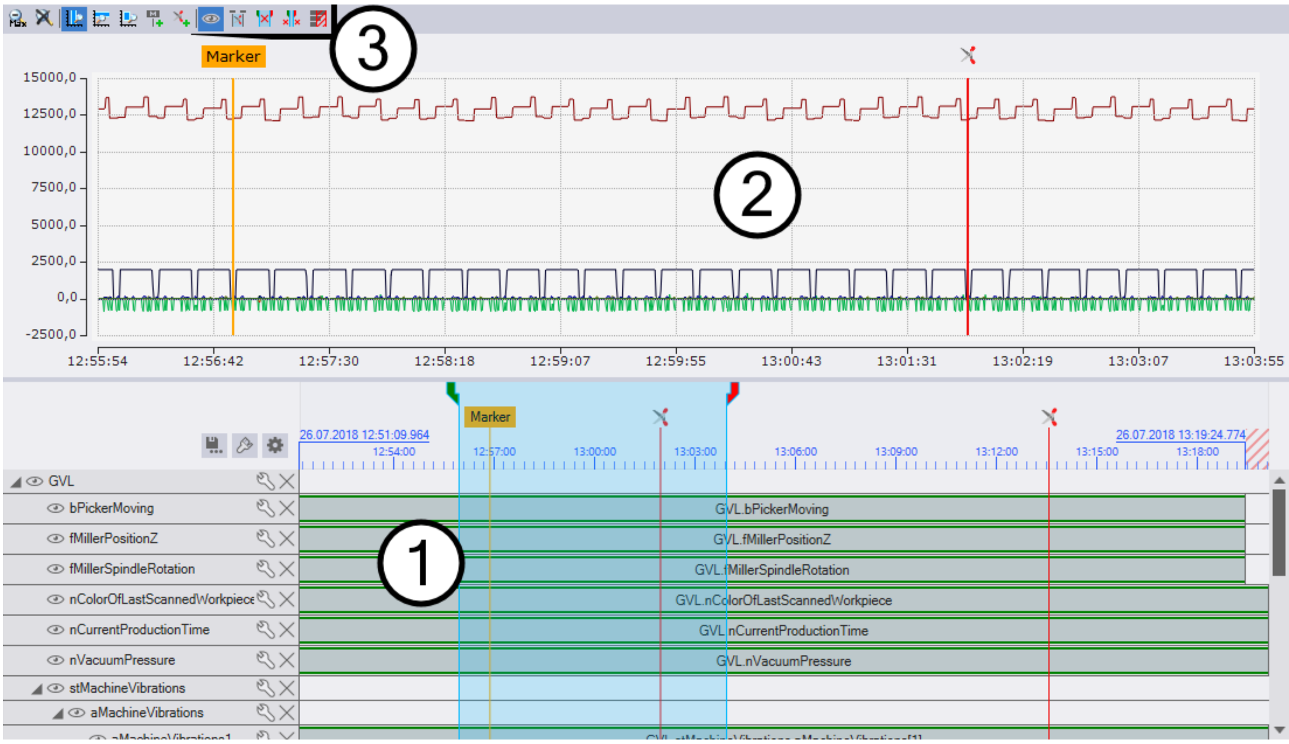
Solution Explorer

Search Solution Explorer (Ctrl+ü)

Solution 'TwinCAT Measurement Project' (1 of 1 project)
TwinCAT Measurement Project
Data Scout Project

Target Browser	
Send Information Report...	
New Data Track Editor	
Copy	Ctrl+C
Paste	Ctrl+V
Remove	Del
Rename	F2
Properties	Alt+Enter

Structure

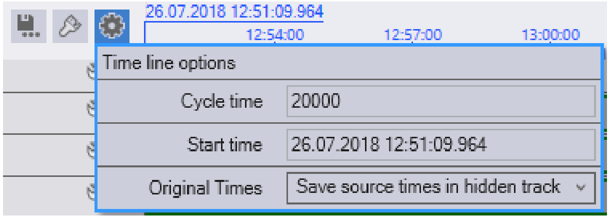


1. Tracks [▶ 54] represent the symbolism and data in the Data Track Editor.
2. The Chart [▶ 60] is used to visualize the data lying in the tracks as graphs.
3. The toolbar [▶ 61] contains functions, for example, to change the displayed area or to eliminate areas.

To export [▶ 62] the 💾 button must be clicked.

Timeline

As different sources can be added in the Track Editor, new times are assigned. The start time and the cycle time can be set in the Timeline Options. These times are used in the export and in the chart. The start and cycle time of the first added source will be taken over if they have not been set yet.



Time line options

Cycle time: 20000

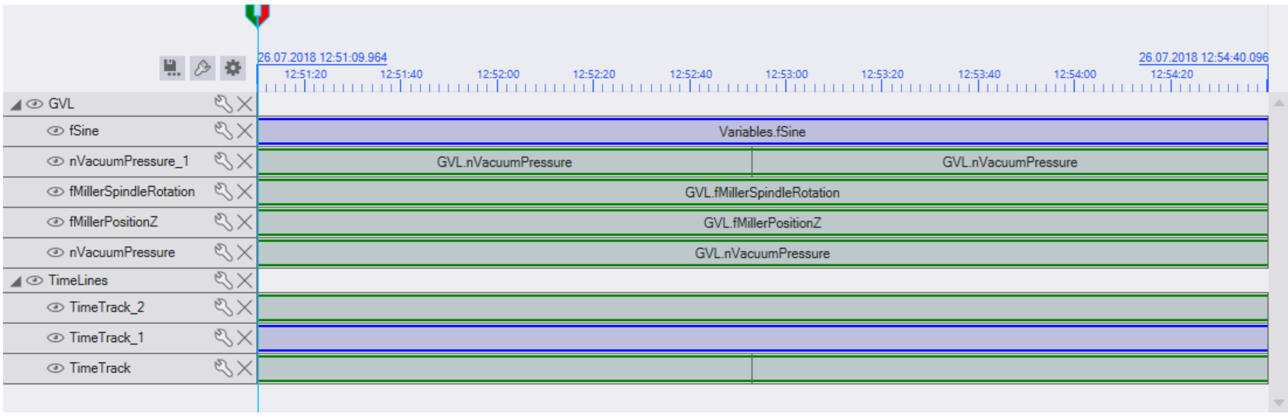
Start time: 26.07.2018 12:51:09.964

Original Times: Save source times in hidden track

The end time is calculated automatically, using the track with the fewest samples.

Time Tracks

In order not to lose the original times of the sources, Time Tracks can be generated. How these are generated can be set in the Timeline Options. There are three variants for generating Time Tracks. The Time Tracks can be generated as visible tracks, and all of them are combined in one container. The second variant is to generate the Time Tracks in the same way, but they and the container they are in are not displayed in the editor. The last variant is not to create any Time Tracks and to discard the original times when exporting.



The Time Track generation works via the data elements of the tracks. For each combination of data items there is a Time Track. Only the time range and the source are considered, but not the symbols of the data elements.

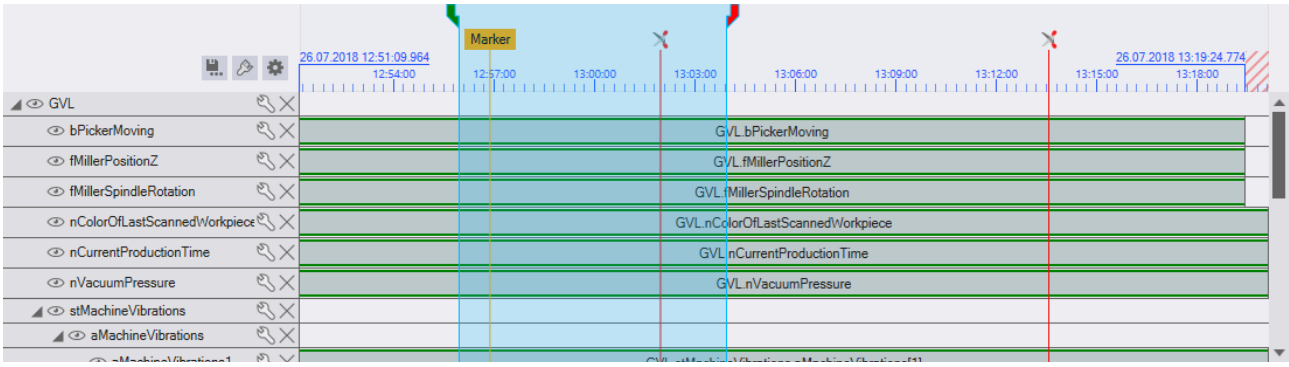
In the example display, the Time Tracks were generated visibly. Three were automatically created because three combinations were found in the tracks above. On the one hand the track with one blue data element, on the other hand a track with two green data elements and three more tracks with one green data element.

Time Tracks are structured like Single Tracks. They can contain data items, but these data items have no symbols, only a time range and a source.

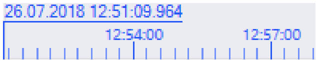
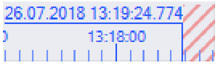
6.2.1.1 Tracks

Tracks are elements of the Data Tracks Editor, which contain information about the contained data and the structure of the symbolism.

Presentation



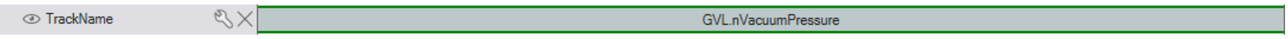
The area provided for tracks in the Data Track Editor has the following elements:

💾	This button opens a new Track Editor Export window.
🔧	Clicking on it opens a popup where settings for the display of the tracks can be made.
⚙	Clicking on it opens a popup where settings for the timeline and Time Tracks can be made.
	The blue area lies above the tracks and represents which data is displayed in the chart. Also, the selected area is used for some toolbar functions. The red and green areas can be moved to increase or decrease the selection. If clicked between the red and green area, the selected area can be moved. On the left and right edges, the selection can snap. This happens near marker and track ends. To ignore the snap function, the Alt key can be pressed while moving. The selection cannot be smaller than 10 values.
	Markers are displayed in the track as well as in the chart. In the tracks, markers can be renamed by clicking on the text and markers can be removed by right-clicking. The markers that have no text but scissors are special markers that can be used for elimination.
	The timeline has the start time marked at the left end. Markers then follow at even intervals, indicating the time present there.
	At the right end of the timeline is a marker for the end time. The end time will be equal to the start time plus the number of samples from the track with the fewest samples times the cycle time. If the number of samples in the tracks differs, a red dashed area is displayed behind the end time marker. This area is not exported, because there would not be data for all tracks.
Tracks Container	In the lower part the tracks are displayed, where the display of the data from the tracks starts at the start time and before that the new symbolism is displayed.

Data Tracks

Currently there are two types of tracks that can contain data. The first is the Single Track. This represents a single symbol. The second is the Multi Track. This can group multiple symbols together, keeping the presentation simple. However, in Multi Tracks e.g. the data types or names of the contained symbols cannot be customized.

Tracks that contain data are displayed in a similar way.



On the left side are the following functions:

👁 ⊘	Here you can define whether the track should be visible as a graph in the chart or not.
Track Name	The name of the track is used in the chart for the graph and in the export as the symbol name. The name must be unique below a container and cannot contain spaces.
🔧	Clicking on it will open a popup. This shows the settings of the track (see the Settings section for more details).
✕	This button removes the track and all the tracks under it

On the right side the data elements of the track are displayed. Data Tracks can contain multiple data elements that match the type of track (e.g. a Single Track cannot contain an array).

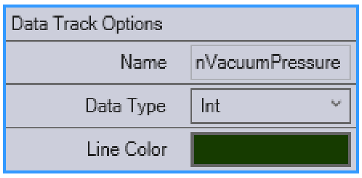
Container Tracks

Container Tracks are structured similarly to Data Tracks. However, no data elements can be in Container Tracks, but they can contain tracks. Included tracks are displayed below the Container Track and can be expanded or collapsed using the ◢ ▷ buttons. When a Container Track is made visible or invisible, all sub tracks become visible or invisible as well.

Settings

For all types of tracks there are different settings in the popup of the 🔧 button.

For Single Tracks there are these:

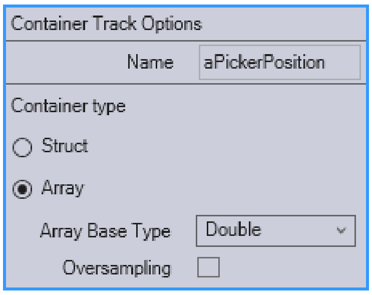


Data Track Options

Name	nVacuumPressure
Data Type	Int
Line Color	🟩

Data type	Here you can define the data type of the Single Track. The displayed as well as the exported data will be converted to the types specified here. For string data types, another text field is displayed, which determines the length of the string. If the data type could lead to problems with the conversion, because one or more data elements of the track have a larger data type, a warning will be issued, but the values will still be converted.
Lines color	Sets the color of the graph in the chart. When you click on it, a color picker is displayed in which you can set the color.

For Container Track there are these settings:



Container Track Options

Name: aPickerPosition

Container type

○ Struct

● Array

Array Base Type: Double

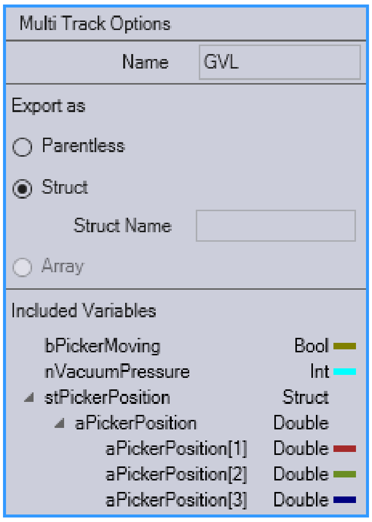
Oversampling: □

Container type	A Container Track can be a structure or an array. To be created as an array, however, all sub tracks must be compatible to form an array. If only Single Tracks are available, a Container Track can be made into an array with a primitive data type. If there are only Container Tracks that have the same structure, the Container Track can be made into an array with primitive array or structure as the base data type.

Each container type has its own properties:

Structure	Structure Name	Specifies the desired name for the data type in the export.
Array	Base data type	Here you can set the data type for the array elements. Setting the data type here changes the array base data type of Sub Container Track and the data type of Sub Single Tracks. For string data types, the length of the string can be set.
	Oversampling	This option is only available if the Container Track has only Single Tracks as sub tracks. When active, the sub tracks are not displayed as individual graphs in the chart, but the Container Track is displayed as one graph. For this purpose, the values of the sub tracks are loaded and all values at the same time are placed one after the other.

For Multi Tracks there are these settings:



Multi Track Options

Name: GVL

Export as

○ Parentless

● Struct

Struct Name:

○ Array

Included Variables

bPickerMoving	Bool	🟩
nVacuumPressure	Int	🟦
◢ stPickerPosition	Struct	
◢ aPickerPosition	Double	
aPickerPosition[1]	Double	🟥
aPickerPosition[2]	Double	🟩
aPickerPosition[3]	Double	🟦

Export as	Similar to the container type of the Container Track, it can be set how the Multi Track should be handled in the export. For Structure, the Multi Track is exported as a structure data type with the contained variables as elements. The desired name for the structure can be specified. The option to export as array is available only if all contained variables have the same data type. The data type can then also not be changed. The Multi Track becomes an array during export with the contained variables as array elements. With Parentless, the Multi Track is not taken into account in the export and it is as if the contained variables were on the Multi Track level.
Variables included	The symbolism present in the Multi Track is displayed here. All data elements of the Multi Track must have the same symbolism.

Data elements

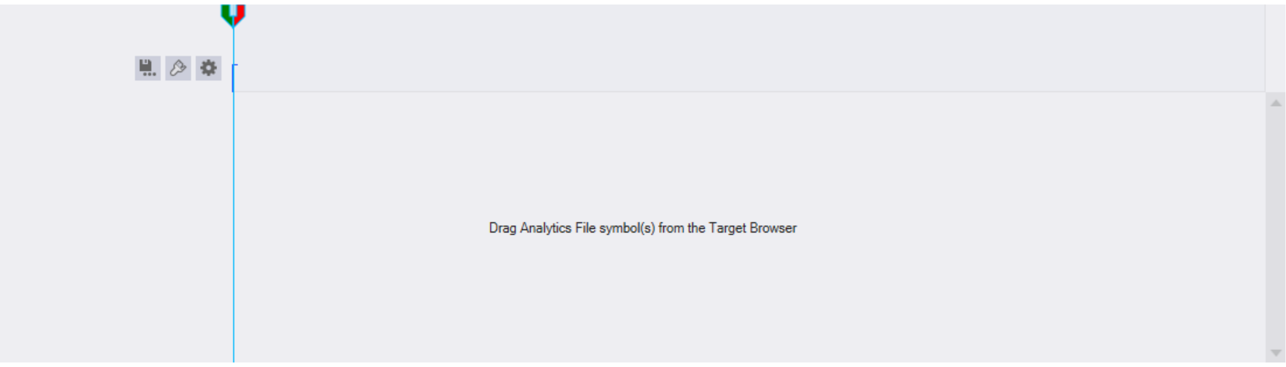
Data elements can be located in Tracks. These contain one or more variables from a source, and a time range from the source. Using the time range and the cycle time of the source, it is calculated how many samples are in the data element. The length of a data item is determined by looking at which track contains the most samples and how much space is available in total. The length then depends on the number of samples and how many pixels correspond to one sample.

A data item is represented as a block. The color of the data item depends on the source it comes from, so all data items from the same source have the same color. There is a dividing line at the end of the data element. The original name of the symbol from the source is displayed in the data element. If there are several symbols, only the number of symbols is displayed.

By right-clicking on a data element, the context menu can be opened. The data element can also be removed completely without using the functions from the toolbar.

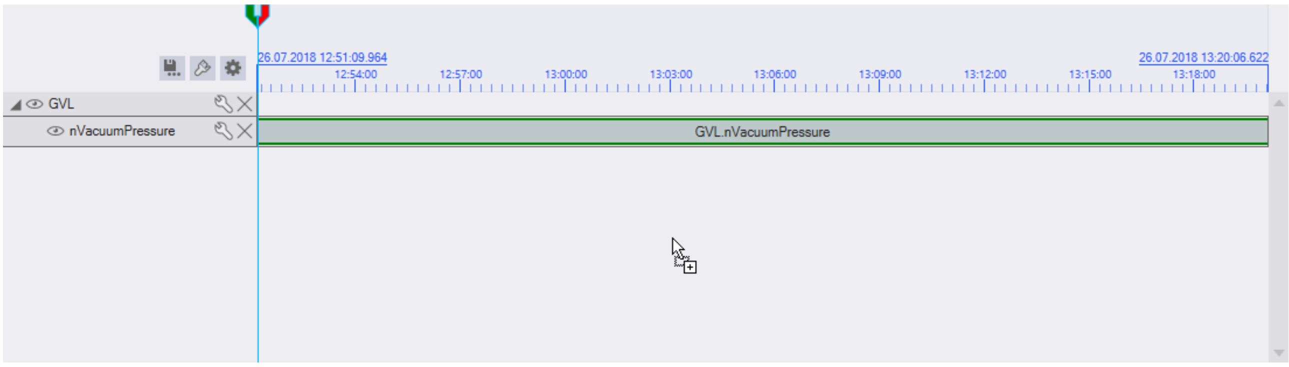
Add via drag and drop

The tracks of a Data Track Editor are displayed in a separate area. This is empty at the beginning and can be filled using the Target Browser.

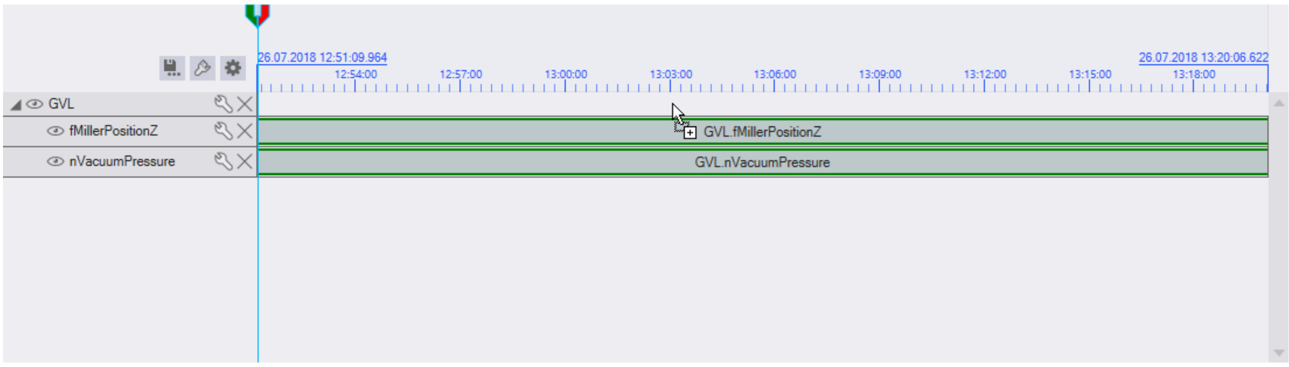


Drag Analytics File symbol(s) from the Target Browser

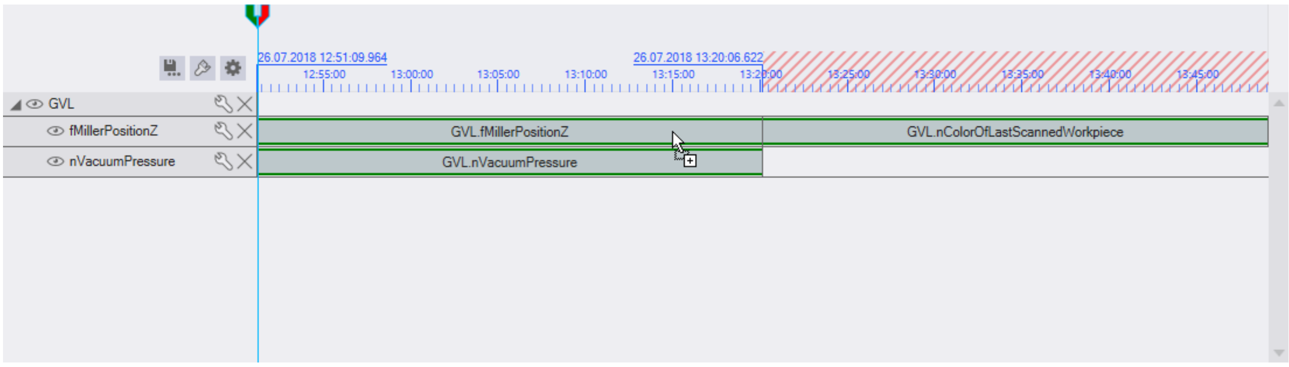
When one or more symbols are selected from the Target Browser and dragged into the empty area of the Track Editor, a new Container Track is created. To the new Container Track all selected symbols will be added as tracks.



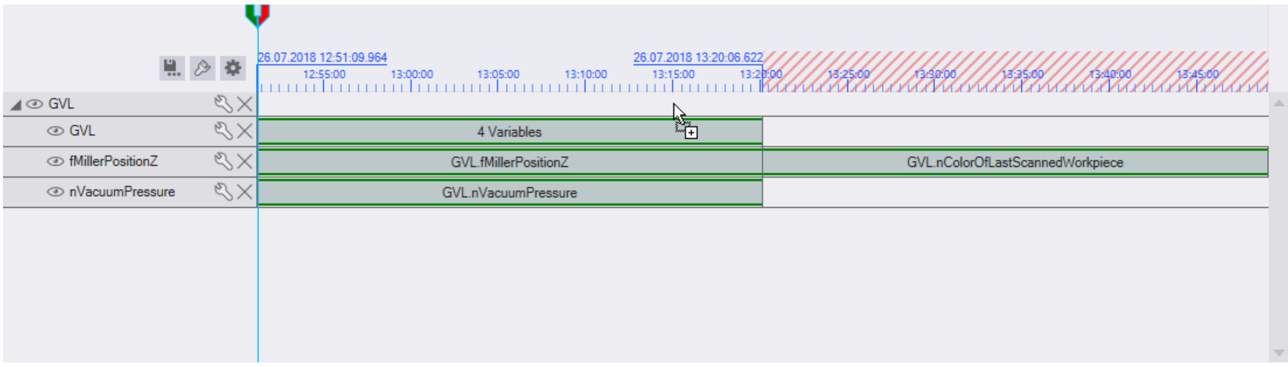
If a symbol is dragged onto the data area of a Container Track, a new track is created below this container.



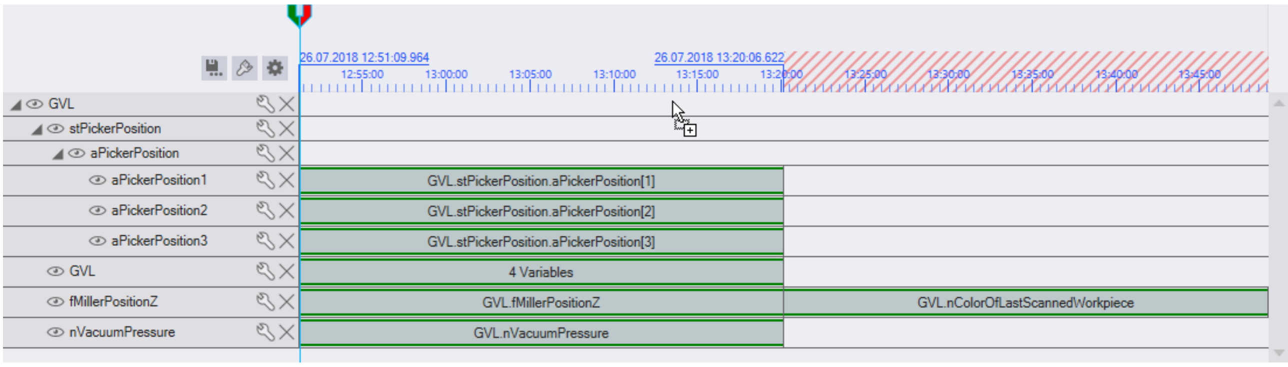
When dragging to a Data Track, a new data element can be added to that track. However, this is only possible if the symbol to be added is compatible with the track. No structure can be drawn on a track with primitive data types.



To create new Multi Tracks you can drag to a Container Track or to the empty area while the [Ctrl] key is pressed.

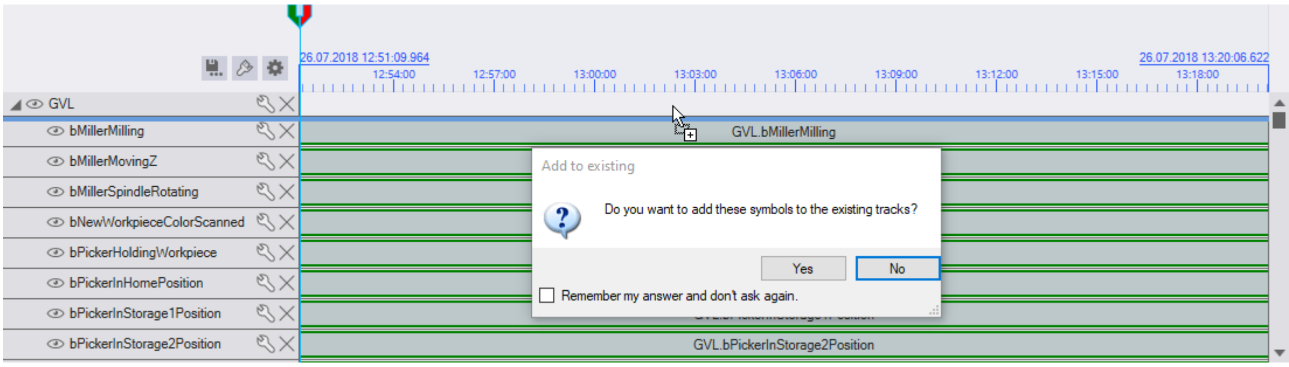


If structures or arrays are to be added, this can happen as with symbols with primitive data type. However, Container Tracks are created for the upper elements of structures and arrays.



Automatic merging

To more easily merge symbols with the same structure, such as in an Analytics File with multiple records, when dragging multiple symbols onto a container or the empty area of the tracks, the system checks if the symbolism is the same. As soon as this is detected, a message appears that it is possible to add the symbols to the already existing tracks.

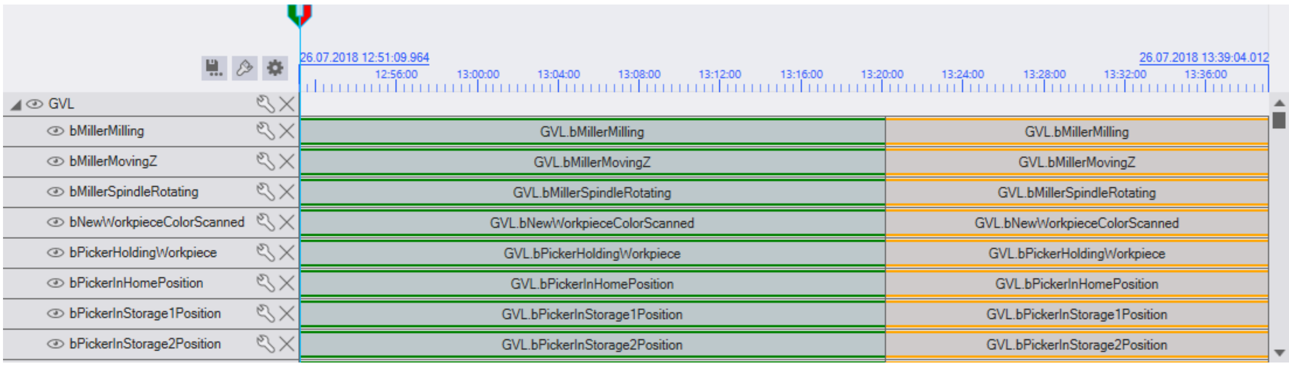

Add to existing

Do you want to add these symbols to the existing tracks?

Yes | No

□ Remember my answer and don't ask again.

If this query is answered with Yes, the added symbols will be appended as a data element to the matching existing track. If the query is answered with No, new tracks are added.


26.07.2018 12:51:09.964
26.07.2018 13:39:04.012
GVL
bMillerMilling
bMillerMovingZ
bMillerSpindleRotating
bNewWorkpieceColorScanned
bPickerHoldingWorkpiece
bPickerInHomePosition
bPickerInStorage1Position
bPickerInStorage2Position
GVL.bMillerMilling
GVL.bMillerMovingZ
GVL.bMillerSpindleRotating
GVL.bNewWorkpieceColorScanned
GVL.bPickerHoldingWorkpiece
GVL.bPickerInHomePosition
GVL.bPickerInStorage1Position
GVL.bPickerInStorage2Position

The same result can be achieved by dragging each symbol individually onto the already existing track.

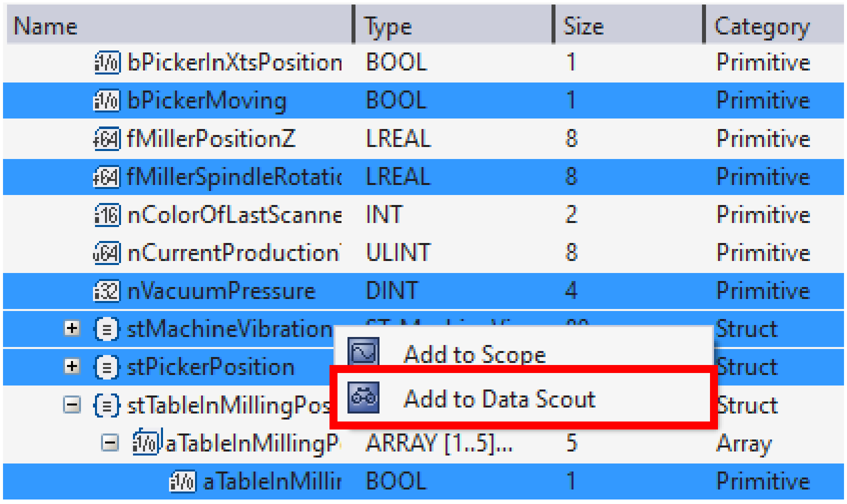
Change order

If the order of existing tracks or data elements is to be changed, this can be done with drag and drop. Data elements can be moved on the whole surface and tracks on the left border . To do this, the area must be clicked with the left mouse button and moved with the mouse button pressed.

Data elements can be rearranged in the same track or moved to another track. When data elements are dragged onto container tracks or into the track editor, a new track is created. Also, the data element must be compatible with the track it is moved to e.g. a data element with multiple symbols from a multi track cannot be moved to a single track.

Tracks can be rearranged below the container they are in or moved to another container. It is not possible to move tracks into containers that are inside the track to be moved or to move tracks that cannot be changed by their configuration or the configuration of the container above them, e.g. structure arrays.

Add via Target Browser Command



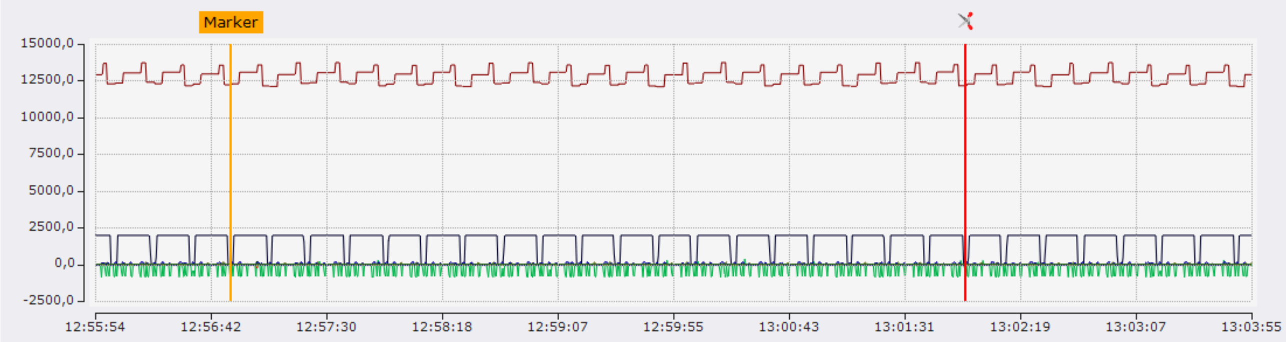
Name	Type	Size	Category
bPickerInXtsPosition	BOOL	1	Primitive
bPickerMoving	BOOL	1	Primitive
fMillerPositionZ	LREAL	8	Primitive
fMillerSpindleRotatio	LREAL	8	Primitive
nColorOfLastScanne	INT	2	Primitive
nCurrentProduction	ULINT	8	Primitive
nVacuumPressure	DINT	4	Primitive
⊞ stMachineVibration	[illegible]	[illegible]	Struct
⊞ stPickerPosition			Struct
⊟ stTableInMillingPos			Struct
⊟ aTableInMillingP	ARRAY [1..5]...	5	Array
aTableInMillir	BOOL	1	Primitive

It is possible to add symbols from the Target Browser to a Track Editor by calling the "Add to Data Scout" command in the context menu. The command is displayed when symbols compatible with the Track Editor are selected in the Target Browser and right-clicked. Tracks are then created for each symbol and the sub-symbols of arrays and structures.

If no Data Scout Project exists, an attempt is made to create a new one. If there is already a Data Scout Project in the current Solution, the symbols will be added there. If a Track Editor is selected in the Solution Explorer, the symbols are inserted into it, otherwise a new one is created.

6.2.1.2 Chart

The Data Track Editor uses a chart similar to the YT chart of the Scope View to visualize the data of the tracks.



The chart has a Y axis on the left and an X axis at the bottom, on which the times are displayed. On these axes, each visible track containing data is represented as a graph.

The time values are reassigned for the chart and the time values from the original sources are not used to represent multiple sources in one chart. The start time as well as the cycle time can be set in the Track Editor Timeline options.

A blue area is displayed in the Track Editor above the timeline and track data elements. This area indicates which area is displayed in the chart. If you zoom in the chart using the toolbar functions or the mouse wheel, the selected area in the tracks also changes. It works the same the other way around, so in the tracks the selected range can be adjusted and the range displayed in the chart changes.

Loading data

As soon as a track with data is in the Data Track Editor, this data is loaded. Loading happens automatically in the background. When loading, the tracks whose composition of data elements is the same, i.e. belong to the same Time Track, are loaded simultaneously. Each time the selected area changes or a new track is inserted, the data for the selected area is updated.

If too much data would be displayed in the chart with a high display width, the data will only be partially displayed.

Load Analytics Files

Analytics Files must be prepared for fast display. In the Target Browser in the Analytics File Extension, a green arrow is displayed at the folder for Analytics Files that are not prepared. Clicking on this arrow opens a new window and the Analytics File is prepared for quick display.

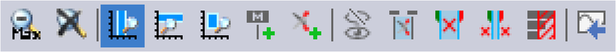
Marker

Markers can be displayed in the chart. These remain at a certain position and can be subsequently moved by clicking and dragging on the X axis.

There are two types of markers. The standard marker has a text above the marker line. The text is highlighted with the same color as the marker line. The second type of marker is intended for cutting out data. There can be only two of these markers in a Data Track editor, if a third is inserted the elimination marker closest to the new one will be removed. Cutout markers have no text above their line, but scissors and the line is always red.

When a marker is clicked with the left mouse button, a tooltip opens showing the time and values at the marker's position. If a marker is clicked with the right mouse button, a context menu is opened. In this there is an option to remove the marker from the Data Track Editor.

6.2.1.3 Toolbar



The Data Track Editor toolbar provides various functions to manipulate the display in the chart and edit the data in the tracks.

These functions are used to reset the X and/or Y axis.

Max	Sets the display width of the chart to the maximum possible width and resets the scaling of the Y axis.
✕	Resets the scaling of the Y axis.

Of these functions, only one is active at a time. They influence how a left click into the chart behaves.

	When this function is activated, it is possible to click in the chart to select a range on the X axis. For this purpose it is necessary to click into the chart, keep the mouse button pressed and then move the mouse to the left or right, which will lead to a preview of the selection. When the mouse button is released, the display width is reduced to this area.
	When this function is activated, it is possible to click in the chart to select a range on the Y axis. When you press the mouse button, a preview of the selected area is displayed. When the mouse button is released, the scaling on the Y axis is reduced to this range.
	When this function is activated, it is possible to click in the chart to reduce the displayed area on the X and Y axes. When you press the mouse button on the chart, a preview of the area that will be displayed is shown. When the mouse button is released, the scaling of the Y axis and the display width is reduced to this area.
	When this function is activated, it is possible to click in the chart to insert a new marker. The marker is also displayed in the tracks and can be renamed there.
	When this function is enabled, it is possible to click in the chart to insert a special marker to cut out an area.

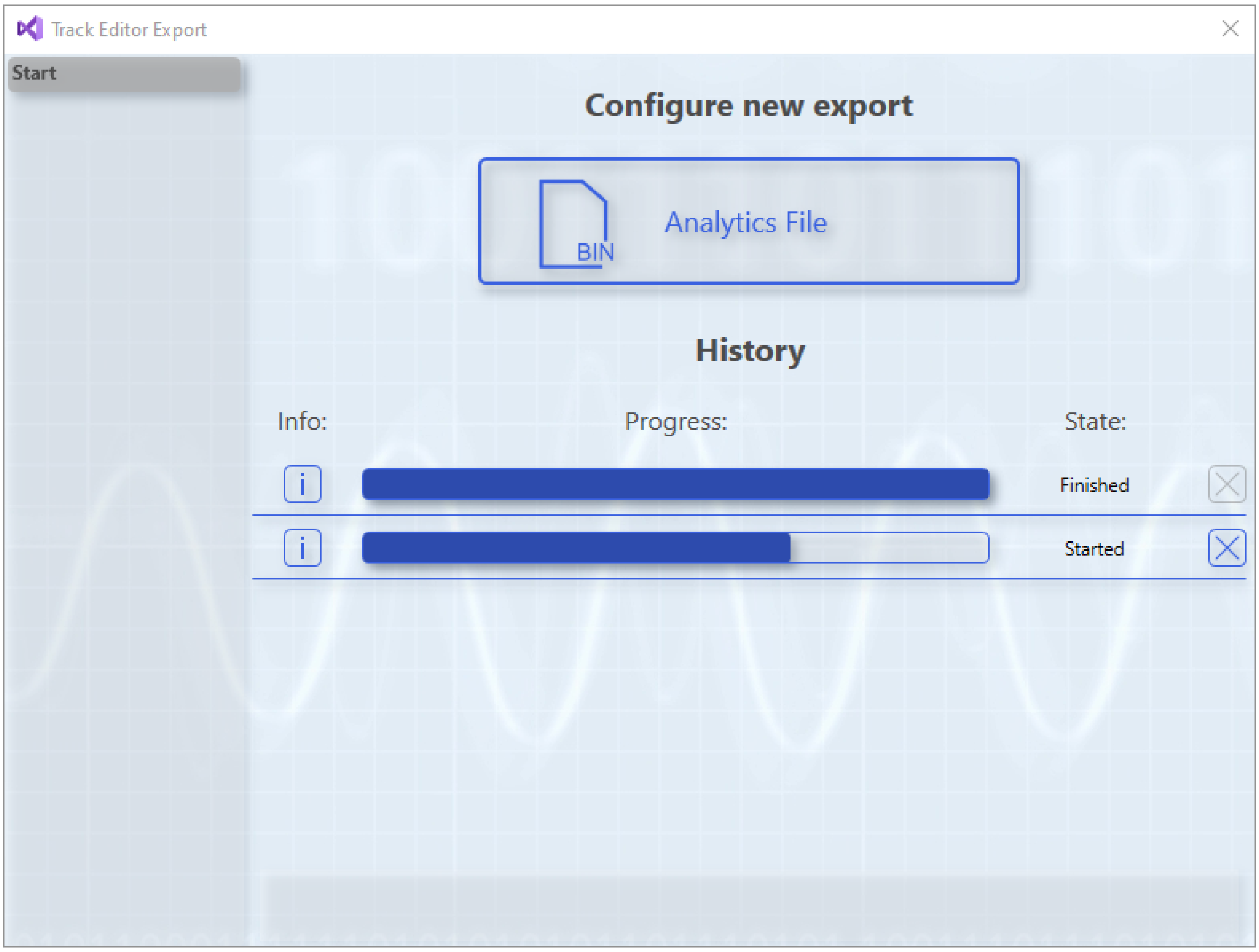
These functions can be used to edit the data of the tracks.

	Determines whether only visible tracks or also invisible tracks are processed by the following functions that can remove data.
	This function can only be used if two markers have been inserted for cutting. These can be inserted with or by right-clicking on a graph. When running, the area between the two markers is removed.
	This function can be used to remove the area selected in the tracks. The selected area will then be reduced in size as the area previously displayed on the chart has been removed.
	This function removes everything except the selected area.
	This function can only be used if not all tracks have the same amount of data. This difference is also shown dashed red in the tracks timeline. When running, the tracks that contain more data will have as much data removed at the end so that the number matches the track that contains the least data.
	Here you can open the Curve Creator in a new window. This contains the data of the selected area, from the visible tracks. After closing the Curve Creator, the graphs are saved in an Analytics file and replaced in the tracks. If the function to change the start or endpoint has been used in the Curve Creator, the range from which the data originates will be replaced, otherwise the previously selected range. This function is only available if the selected range is smaller than 100,000 values.

6.2.1.4 Export

A Data Track Editor can be exported using the Track Editor Export Wizard. To call the wizard you should click this button , which is next to the tracks timeline.

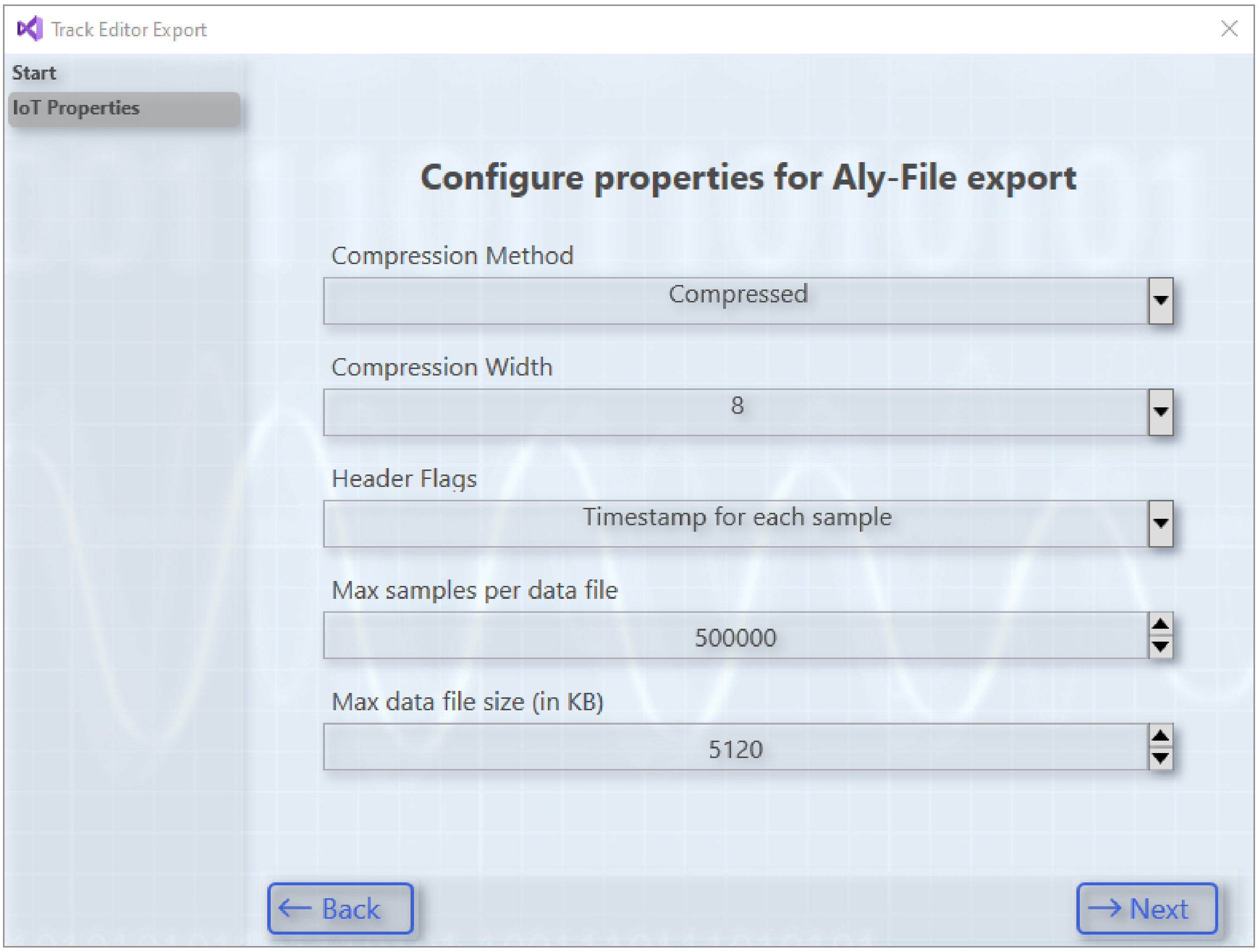
Start page



On the start page of the wizard, new exports can be started in the upper part. All available export formats are displayed as a separate button. In the lower part is a history of exports that were started while the project was open. Each entry in the history has a button on the left side with info about the export. The info contains the start time of the export, important properties depending on the format and if errors occur, the error message. To the right of this is a progress bar showing how many of the samples to be exported have already been exported. After that, the status of the export is output. The button on the far right can be used to cancel running exports.

When clicking one of the buttons to start a new export, one or more new pages will appear in the wizard where the settings for the format can be made.

Analytics File Configuration Page



Configure properties for Aly-File export

Compression Method: Compressed

Compression Width: 8

Header Flags: Timestamp for each sample

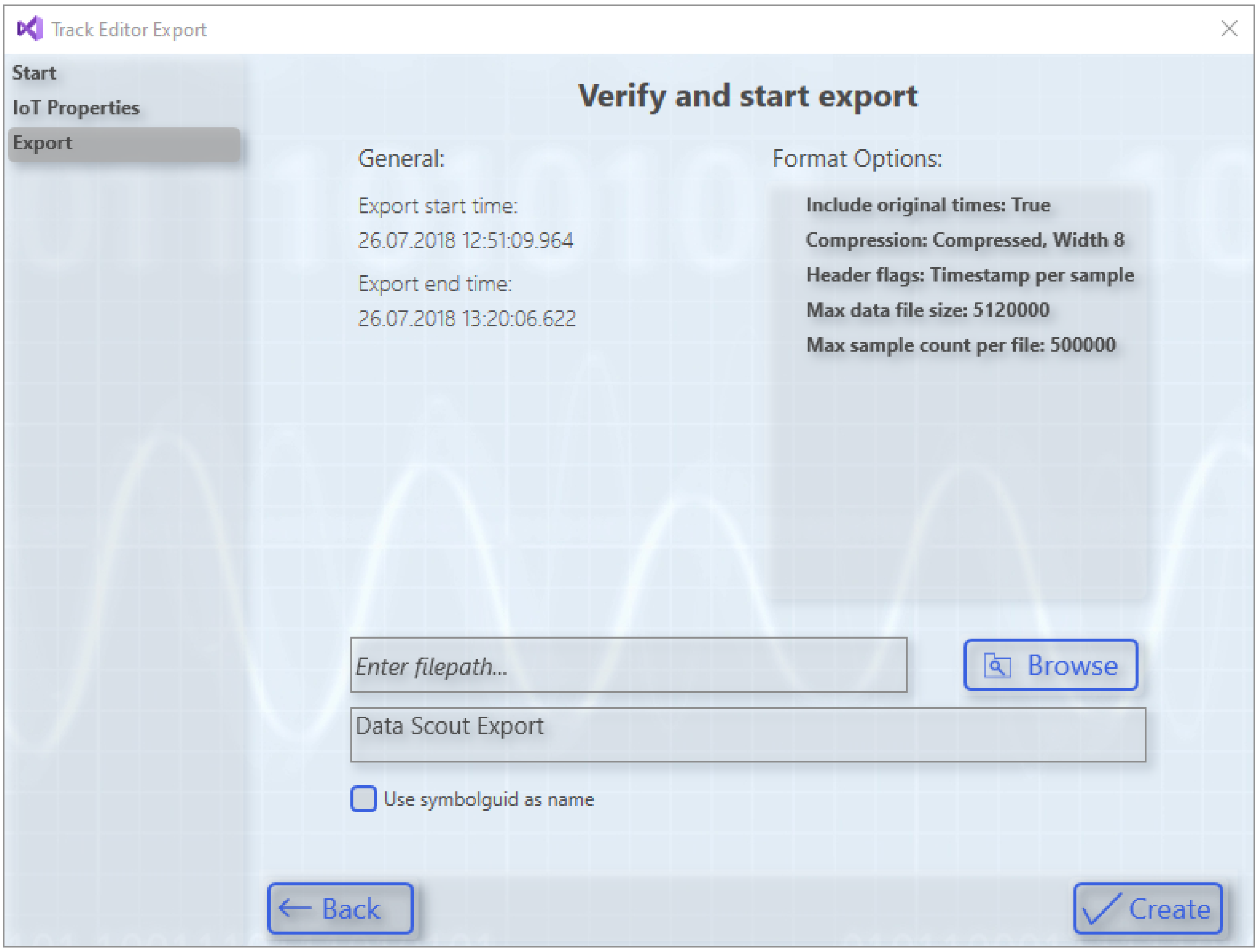
Max samples per data file: 500000

Max data file size (in KB): 5120

This Analytics File Format configuration page specifies the following settings:

Compression method	Analytics File can be saved compressed or uncompressed. (Data Compression)
Compression width	This option is only available if the Analytics File is to be saved in compressed form. It gives the maximum width to compare for compression.
Header flags	In the flags you can specify whether each .tay file should have only one time value or whether each sample in the .tay file should have its own time value.
Maximum samples and file size	With these two properties you can set how big the .tay files of the Analytics File can become. On the one hand you can set the maximum number of samples to be stored in a file and on the other hand you can set the size of a file in kilobytes in the file system. If one of these values is exceeded, a new .tay file is created.

Export page



Verify and start export

General:

Export start time:
26.07.2018 12:51:09.964

Export end time:
26.07.2018 13:20:06.622

Format Options:

Include original times: True
Compression: Compressed, Width 8
Header flags: Timestamp per sample
Max data file size: 5120000
Max sample count per file: 500000

Enter filepath...

Data Scout Export

□ Use symbolguid as name

The Export page is displayed after going through the configuration pages for the format. Here all settings are listed again and a storage location can be specified if necessary for the format. The upper text field indicates the folder and the lower one the file name.

For the Analytics File Format, a checkbox can also be checked instead of specifying a file name, then the ID of the symbolism will be used as the folder name.

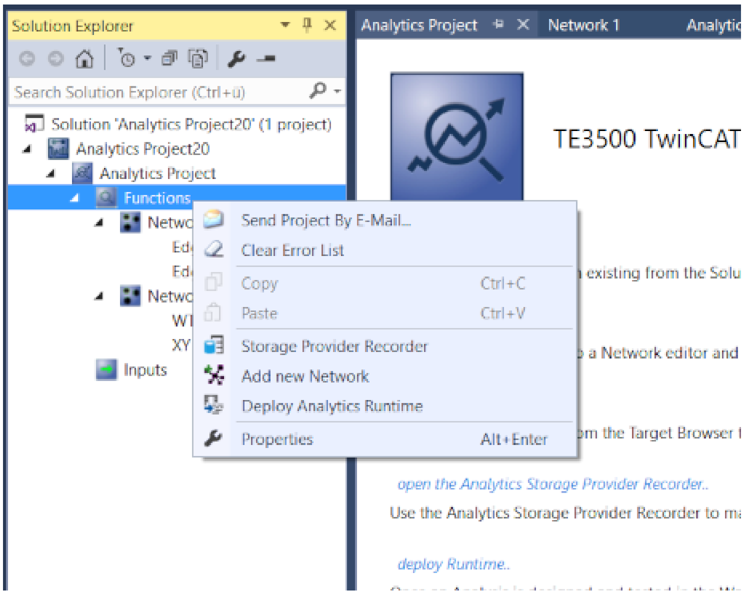
When the Create button is clicked, the export starts in the background. Then this page closes and the wizard jumps back to the start page. The new export is then also listed in the history and the progress can be tracked.

6.3 Editor Basics

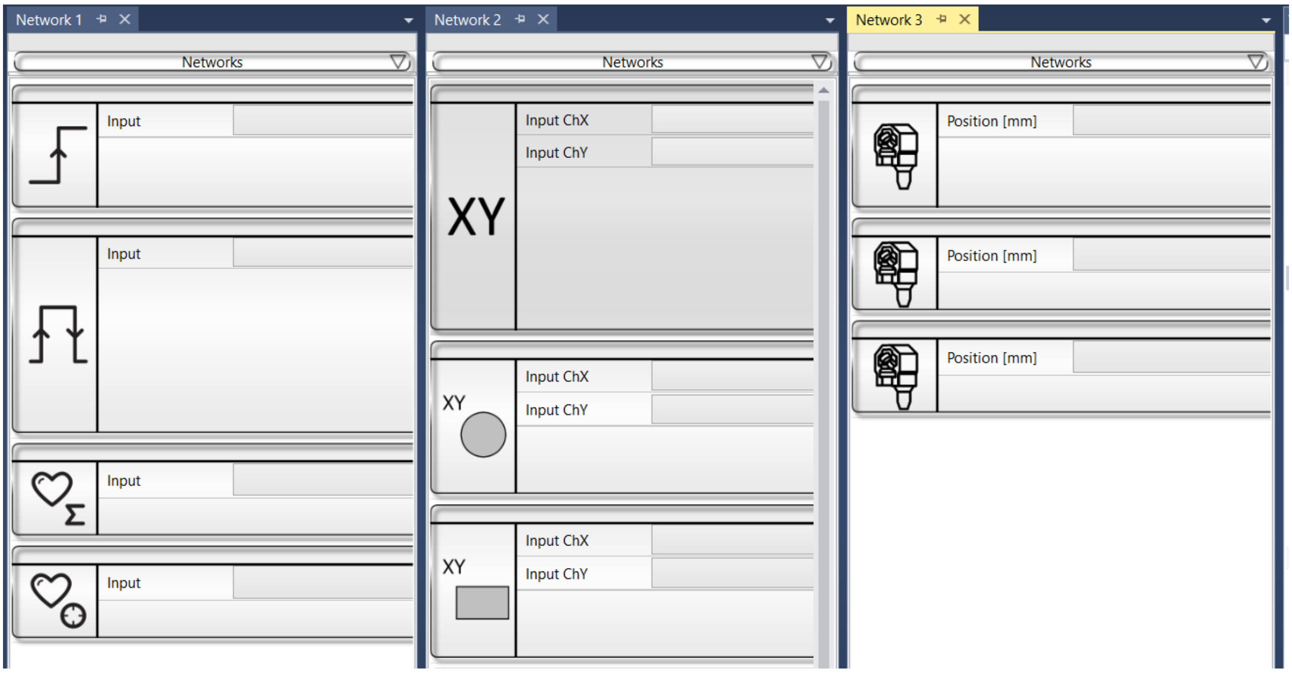
6.3.1 Networks

The networks are suitable for organizing and structuring an analysis. This significantly increases the clarity. Furthermore, they can also serve as containers for algorithms, which you can save as templates.

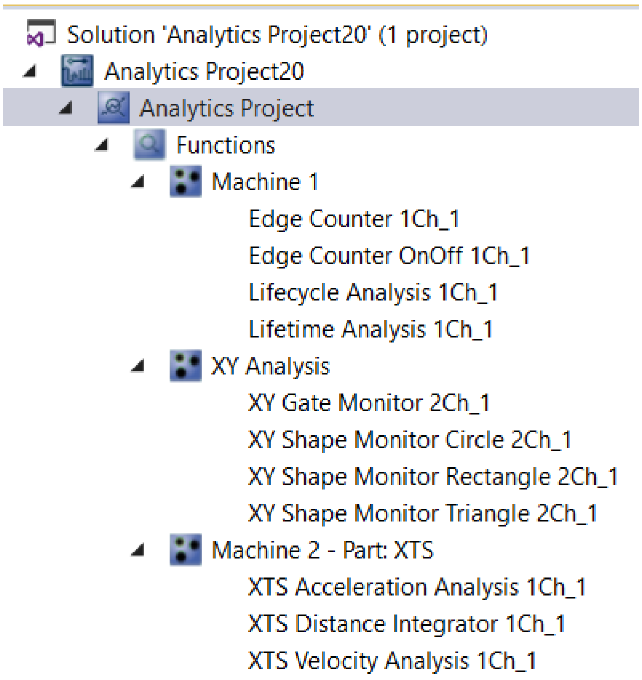
A network can be added directly from the Analytics Project home page or from the context menu.



Each network is displayed in an individual tab. In this way, the networks can be displayed separately, i.e. side by side or one above the other.



Furthermore, you have the option to rename the networks (F2 on the selected network element in the Solution Explorer) to create networks for different machines, machine parts or other content-related connections, for example.

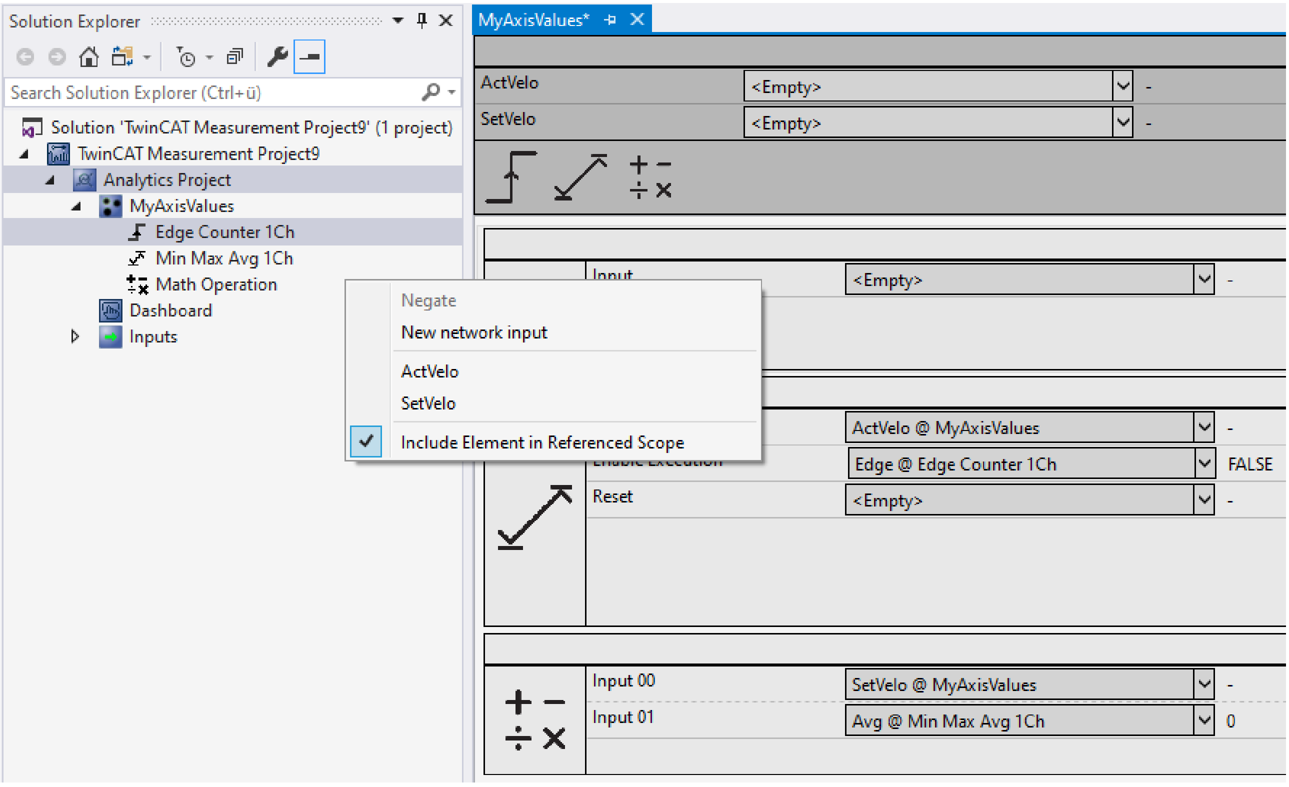


Solution 'Analytics Project20' (1 project)
- Analytics Project20
  - Analytics Project
    - Functions
      - Machine 1
        - Edge Counter 1Ch_1
        - Edge Counter OnOff 1Ch_1
        - Lifecycle Analysis 1Ch_1
        - Lifetime Analysis 1Ch_1
      - XY Analysis
        - XY Gate Monitor 2Ch_1
        - XY Shape Monitor Circle 2Ch_1
        - XY Shape Monitor Rectangle 2Ch_1
        - XY Shape Monitor Triangle 2Ch_1
      - Machine 2 - Part: XTS
        - XTS Acceleration Analysis 1Ch_1
        - XTS Distance Integrator 1Ch_1
        - XTS Velocity Analysis 1Ch_1

6.3.1.1 Networks as template

Inputs, parameters or even outputs of algorithms within a network can be pinned directly to the outside of the network. Thus, the network itself has inputs, parameters and outputs. This makes it possible to save recurring analyses as templates and to instantiate them several times.

Once you have created an analysis, you must first select which of the available inputs should be visible to the outside of the network. To do this, select the desired input and switch to the context menu by right-clicking. There you can link the input to an existing network input or alternatively define a new network input.



Solution Explorer
Search Solution Explorer (Ctrl+ü)
MyAxisValues*

Solution 'TwinCAT Measurement Project9' (1 project)
- TwinCAT Measurement Project9
  - Analytics Project
    - MyAxisValues
      - Edge Counter 1Ch
      - Min Max Avg 1Ch
      - Math Operation
    - Dashboard
    - Inputs

ActVelo	<Empty>	-
SetVelo	<Empty>	-

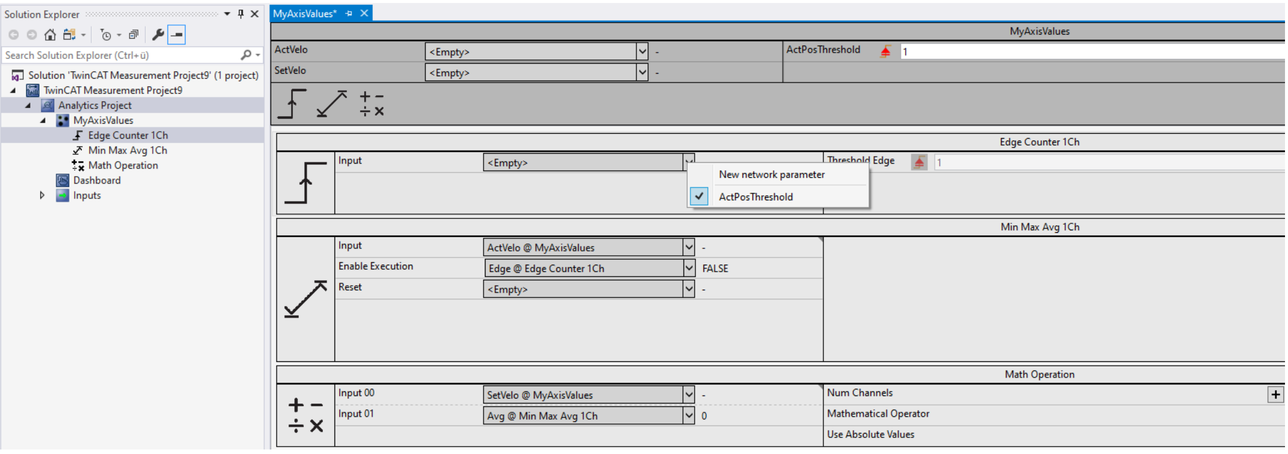
Input	<Empty>	-

Negate
New network input
ActVelo
SetVelo
✓ Include Element in Referenced Scope

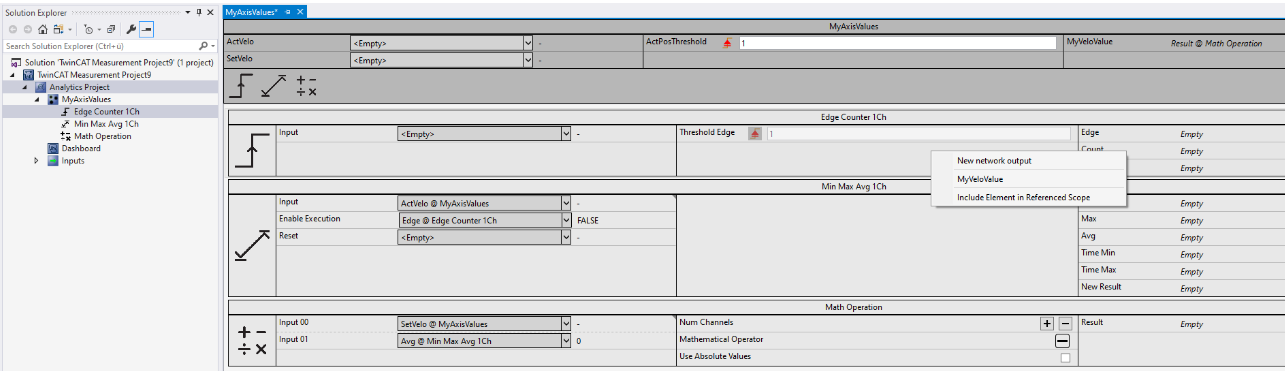
	ActVelo @ MyAxisValues	-
Enable Execution	Edge @ Edge Counter 1Ch	FALSE
Reset	<Empty>	-

Input 00	SetVelo @ MyAxisValues	-
Input 01	Avg @ Min Max Avg 1Ch	0

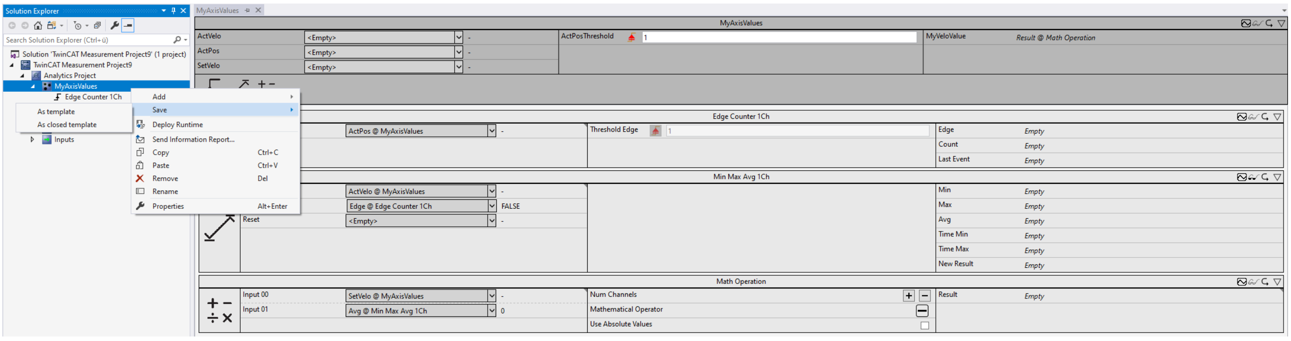
The same procedure is available for parameters and outputs.



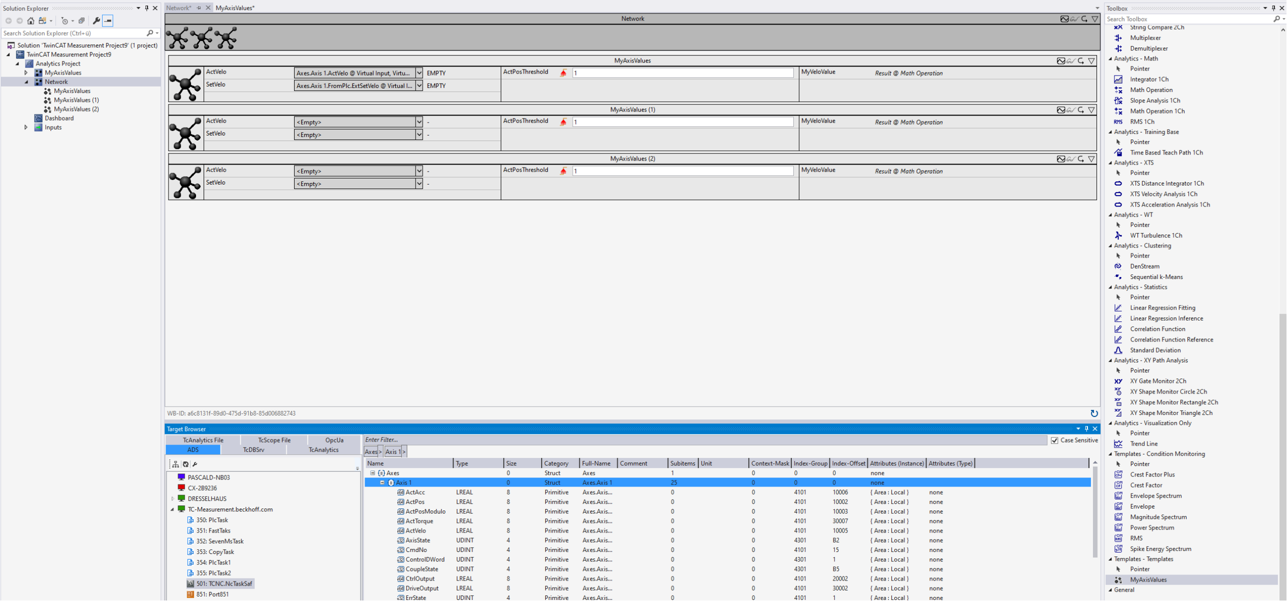
In addition to these inputs and outputs, it is also possible to specify dynamic inputs and outputs. These are automatically offered to you when one of the selected algorithms supports this function. The background to this is the option of increasing the number of inputs or outputs on an algorithm via a parameter, e.g. the inputs of the Math Operation algorithm.



Once the definition of the network to be used as a template is complete, you can save it accordingly. To do this, go to Solution Explorer and right-click on the network. Use the Save option in the context menu. You can choose between Template and Closed Template. In a simple template you can look inside after instantiation and see the interconnection of the basic algorithms and also change them. This option is not available for a closed network. However, this does not offer know-how protection! The internal logic is also visible in a possible PLC code generation by the Analytics Workbench.



After saving, the template is selectable in the toolbox and can be used for recurring analyses.



The Target Browser offers a special function.
If the names have been chosen for the network in such a way that they correspond exactly to variables of a structure, you can drag a complete structure such as an axes structure directly onto one of the x-inputs of a network. All matching names of all network inputs and structure variables are mapped automatically.

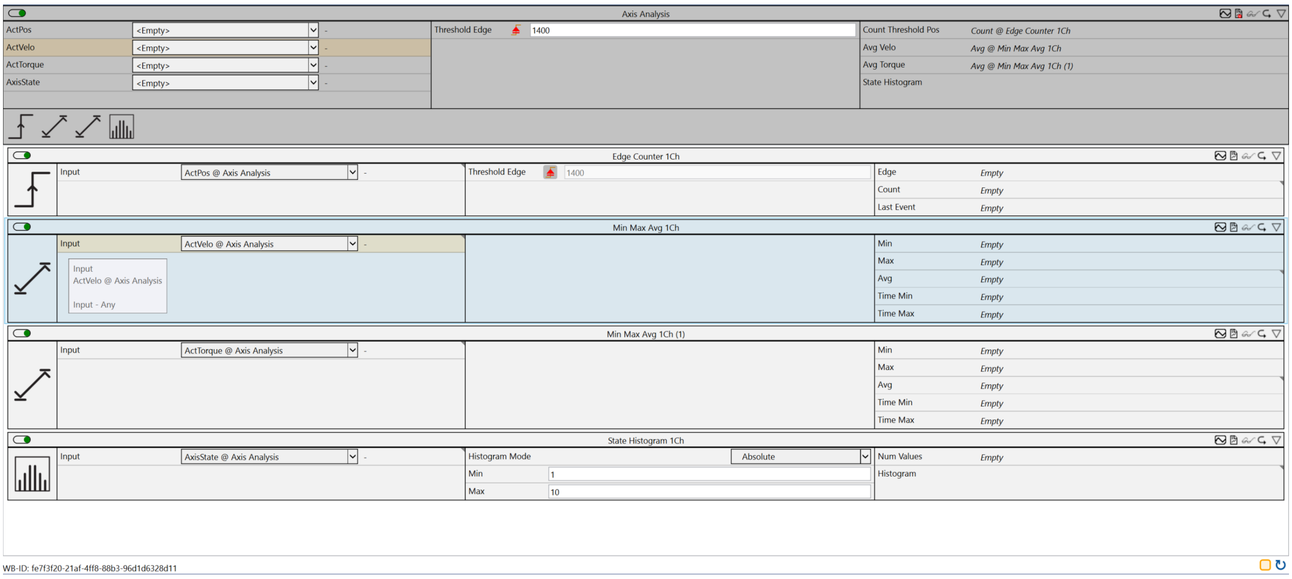
Scope configuration stored in network template

A created Scope configuration can be saved together with the associated network in a network template [▸ 348], in order to automatically obtain the same Scope configuration when a network is used again.

6.3.2 Smart Link

The Smart Link functions in TwinCAT Analytics make engineering easier. Modules can be linked to variables and other values in groups very quickly using a few key/mouse combinations.

A first variant of the Smart Link function is the linking of structure members from the Target Browser. This is particularly useful if you use Analytics network templates. This requires that you name your network inputs exactly the same as the members that you want to link from a structure.


Axis Analysis

ActPos	<Empty>	Threshold Edge	1400
ActVelo	<Empty>		
ActTorque	<Empty>		
AxisState	<Empty>		

Count Threshold Pos	Count @ Edge Counter 1Ch
Avg Velo	Avg @ Min Max Avg 1Ch
Avg Torque	Avg @ Min Max Avg 1Ch (1)
State Histogram	

Edge Counter 1Ch

Input: ActPos @ Axis Analysis; Threshold Edge: 1400; Edge: Empty; Count: Empty; Last Event: Empty

Min Max Avg 1Ch

Input: ActVelo @ Axis Analysis; Min: Empty; Max: Empty; Avg: Empty; Time Min: Empty; Time Max: Empty

Input
ActVelo @ Axis Analysis
Input - Any

Min Max Avg 1Ch (1)

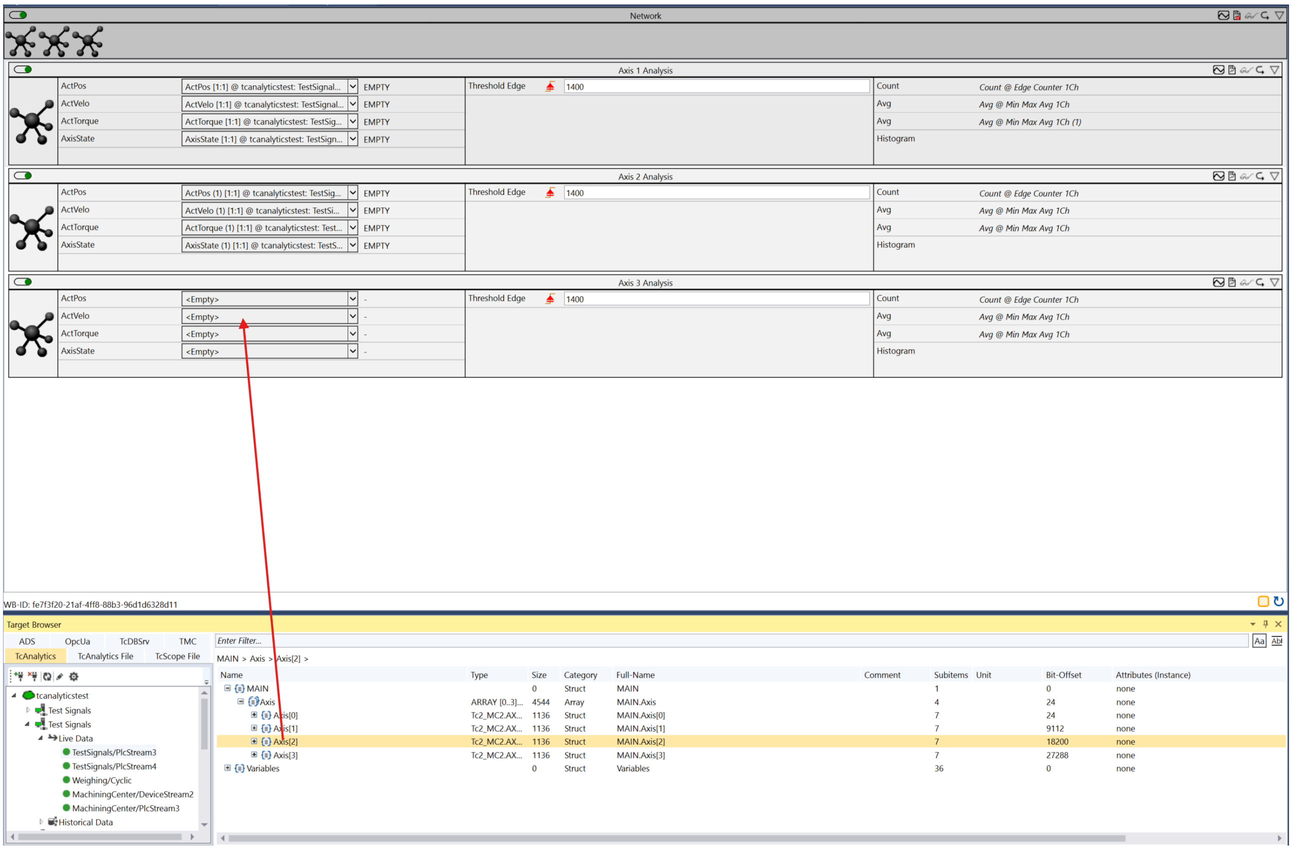
Input: ActTorque @ Axis Analysis; Min: Empty; Max: Empty; Avg: Empty; Time Min: Empty; Time Max: Empty

State Histogram 1Ch

Input: AxisState @ Axis Analysis; Histogram Mode: Absolute; Min: 1; Max: 10; Num Values: Empty; Histogram

WB-ID: fe7f3f20-21af-4ff8-88b3-96d1d6328d11

In the figure shown, these are the names of the known NC structure NcToPlc. The inputs are given exactly the same names as the members in the structure.

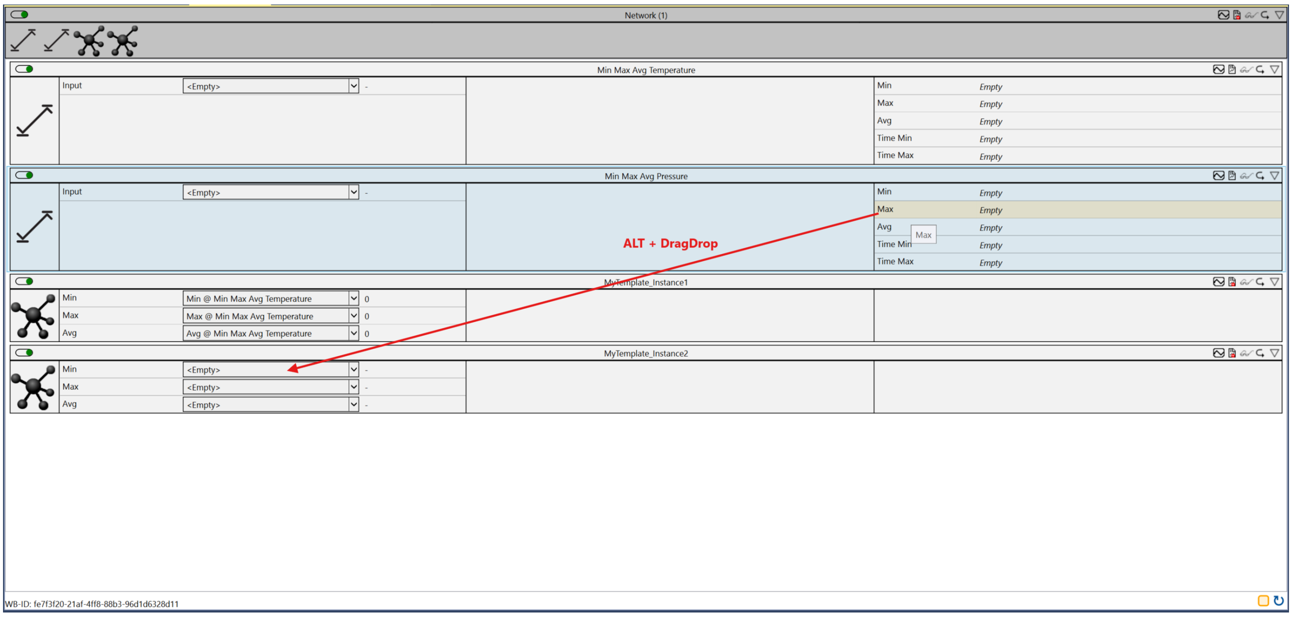


Network
Axis 1 Analysis
ActPos
ActVelo
ActTorque
AxisState
ActPos [1:1] @ tcanalyticstest: TestSignal...
ActVelo [1:1] @ tcanalyticstest: TestSignal...
ActTorque [1:1] @ tcanalyticstest: TestSig...
AxisState [1:1] @ tcanalyticstest: TestSign...
EMPTY
Threshold Edge
1400
Count
Count @ Edge Counter 1Ch
Avg
Avg @ Min Max Avg 1Ch
Avg @ Min Max Avg 1Ch (1)
Histogram
Axis 2 Analysis
ActPos (1) [1:1] @ tcanalyticstest: TestSig...
ActVelo (1) [1:1] @ tcanalyticstest: TestSi...
ActTorque (1) [1:1] @ tcanalyticstest: Test...
AxisState (1) [1:1] @ tcanalyticstest: TestS...
Axis 3 Analysis
<Empty>
WB-ID: fe7f3f20-21af-4ff8-88b3-96d1d6328d11
Target Browser
ADS
OpcUa
TcDBSrv
TMC
TcAnalytics
TcAnalytics File
TcScope File
Enter Filter...
MAIN > Axis > Axis[2] >
tcanalyticstest
Test Signals
Test Signals
Live Data
TestSignals/PlcStream3
TestSignals/PlcStream4
Weighing/Cyclic
MachiningCenter/DeviceStream2
MachiningCenter/PlcStream3
Historical Data

Name	Type	Size	Category	Full-Name	Comment	Subitems	Unit	Bit-Offset	Attributes (Instance)
MAIN		0	Struct	MAIN		1		0	none
Axis	ARRAY [0..3]...	4544	Array	MAIN.Axis		4		24	none
Axis[0]	Tc2_MC2.AX...	1136	Struct	MAIN.Axis[0]		7		24	none
Axis[1]	Tc2_MC2.AX...	1136	Struct	MAIN.Axis[1]		7		9112	none
Axis[2]	Tc2_MC2.AX...	1136	Struct	MAIN.Axis[2]		7		18200	none
Axis[3]	Tc2_MC2.AX...	1136	Struct	MAIN.Axis[3]		7		27288	none
Variables		0	Struct	Variables		36		0	none

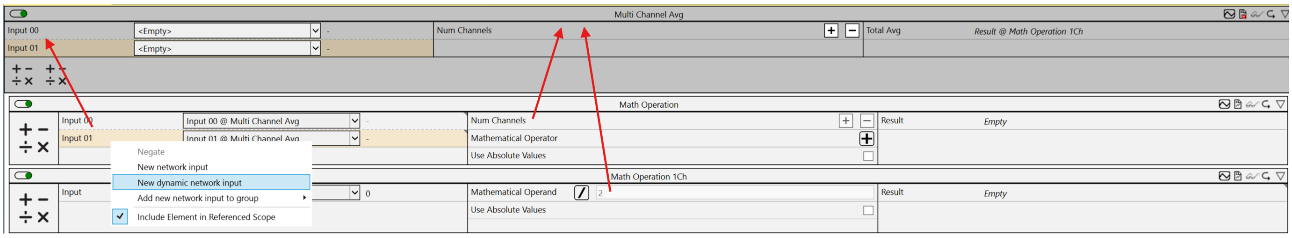
The drive axis structures can be easily linked in the following step. It is sufficient to link the desired structure to one of the inputs of the network template using drag and drop. All members are automatically linked to the inputs.

Another Smart Link function makes it easy to link outputs to inputs while taking their names into account. For example, build a network template and name the inputs in the same way as the outputs of the modules to be linked.



This means that, by holding down the Alt key while dragging and dropping an output onto an input of the template, all other outputs can automatically link to their corresponding inputs.

The next Smart Link function is combined with a search function in the modules of the current network. For example, it creates a network template with dynamic inputs, which allows the number of inputs to be set via the parameters during subsequent use. The Num-Channel parameter, which can be seen as an example in the screenshot, is also linked as a network parameter. The same applies to the operand, which grows with the number of channels.



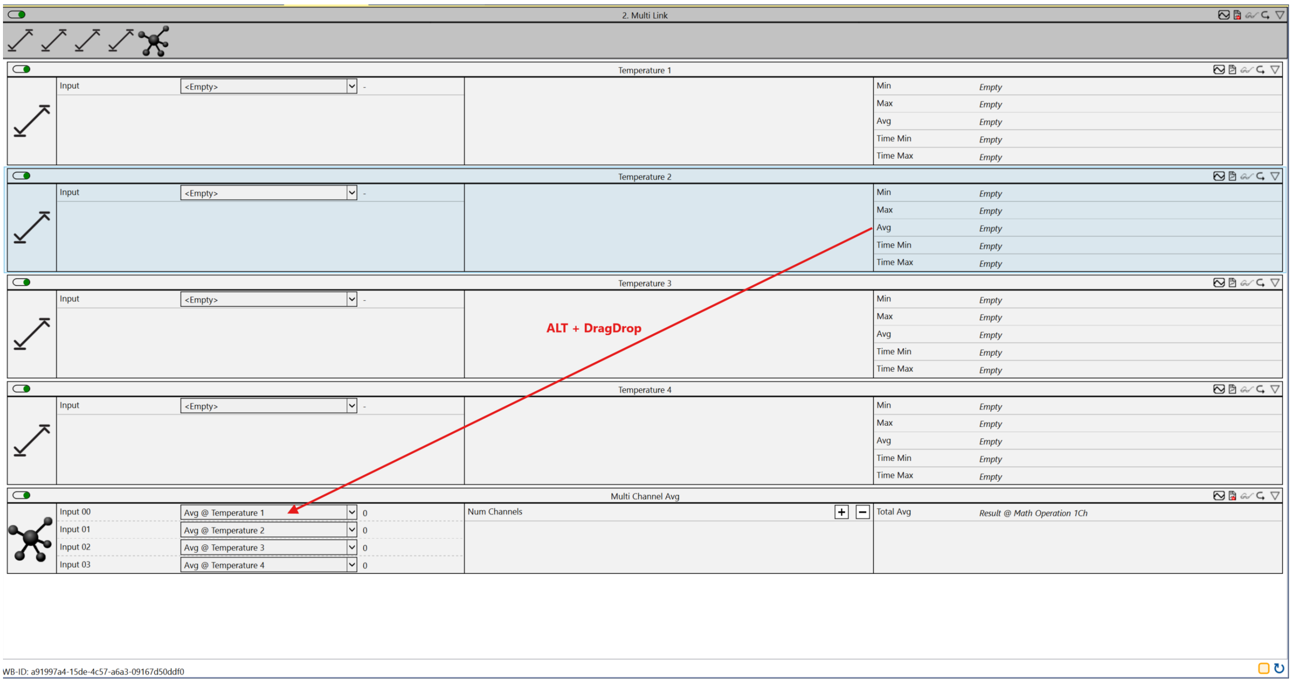
Multi Channel Avg
Input 00 <Empty> -
Input 01 <Empty> -
Num Channels
Total Avg Result @ Math Operation 1Ch

Math Operation
Input 00 Input 00 @ Multi Channel Avg -
Input 01 Input 01 @ Multi Channel Avg -
Num Channels
Mathematical Operator
Use Absolute Values □
Result Empty

Negate
New network input
New dynamic network input
Add new network input to group
Include Element in Referenced Scope

Math Operation 1Ch
Input 0
Mathematical Operand 2
Use Absolute Values □
Result Empty

The example should simply create the sum of the X inputs and calculate the average.



2. Multi Link

Temperature 1
Input <Empty> -
Min Empty
Max Empty
Avg Empty
Time Min Empty
Time Max Empty

Temperature 2
Input <Empty> -
Min Empty
Max Empty
Avg Empty
Time Min Empty
Time Max Empty

Temperature 3
Input <Empty> -
Min Empty
Max Empty
Avg Empty
Time Min Empty
Time Max Empty

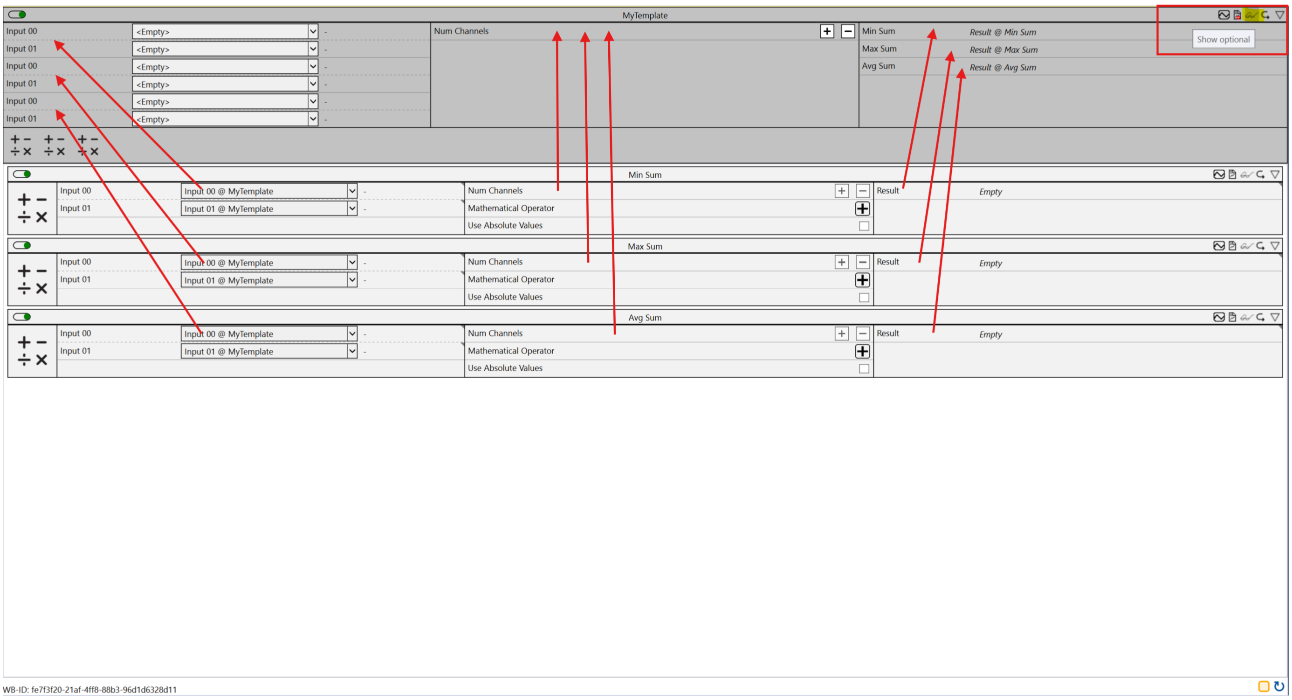
Temperature 4
Input <Empty> -
Min Empty
Max Empty
Avg Empty
Time Min Empty
Time Max Empty

Multi Channel Avg
Input 00 Avg @ Temperature 1 0
Input 01 Avg @ Temperature 2 0
Input 02 Avg @ Temperature 3 0
Input 03 Avg @ Temperature 4 0
Num Channels
Total Avg Result @ Math Operation 1Ch

WB-ID: a91997a4-15de-4c57-a6a3-09167d50ddf0

During use, the number of inputs is entered in the template. Then, while holding down the ALT key, drag an output with a corresponding name in the local network to one of the template inputs. This feature searches the local network for outputs with the same name and automatically links them to the template's inputs.

The Smart Link function shown can also be extended to group linking. To do this, first define dynamic inputs, parameters and corresponding outputs in your template. In the example, the sum of all minimum, maximum, and average values of a network is to be calculated.


MyTemplate

Input 00 <Empty> -
Input 01 <Empty> -
Input 00 <Empty> -
Input 01 <Empty> -
Input 00 <Empty> -
Input 01 <Empty> -

Num Channels

Min Sum *Result @ Min Sum*
Max Sum *Result @ Max Sum*
Avg Sum *Result @ Avg Sum*

Show optional

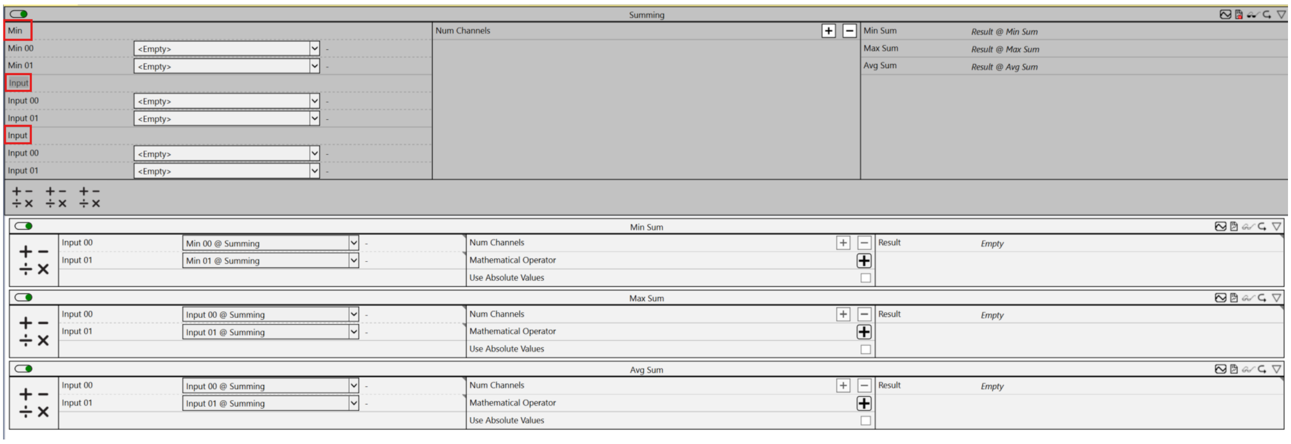
Min Sum
Input 00 Input 00 @ MyTemplate -
Input 01 Input 01 @ MyTemplate -
Num Channels
Mathematical Operator
Use Absolute Values □
Result *Empty*

Max Sum
Input 00 Input 00 @ MyTemplate -
Input 01 Input 01 @ MyTemplate -
Num Channels
Mathematical Operator
Use Absolute Values □
Result *Empty*

Avg Sum
Input 00 Input 00 @ MyTemplate -
Input 01 Input 01 @ MyTemplate -
Num Channels
Mathematical Operator
Use Absolute Values □
Result *Empty*

WB-ID: fe7f3f20-21af-4ff8-88b3-96d1d6328d11

To make the group link work, click on the glasses icon in the top right corner at the network level to reveal more details.


Summing

Min
Min 00 <Empty> -
Min 01 <Empty> -
Input
Input 00 <Empty> -
Input 01 <Empty> -
Input
Input 00 <Empty> -
Input 01 <Empty> -

Num Channels

Min Sum *Result @ Min Sum*
Max Sum *Result @ Max Sum*
Avg Sum *Result @ Avg Sum*

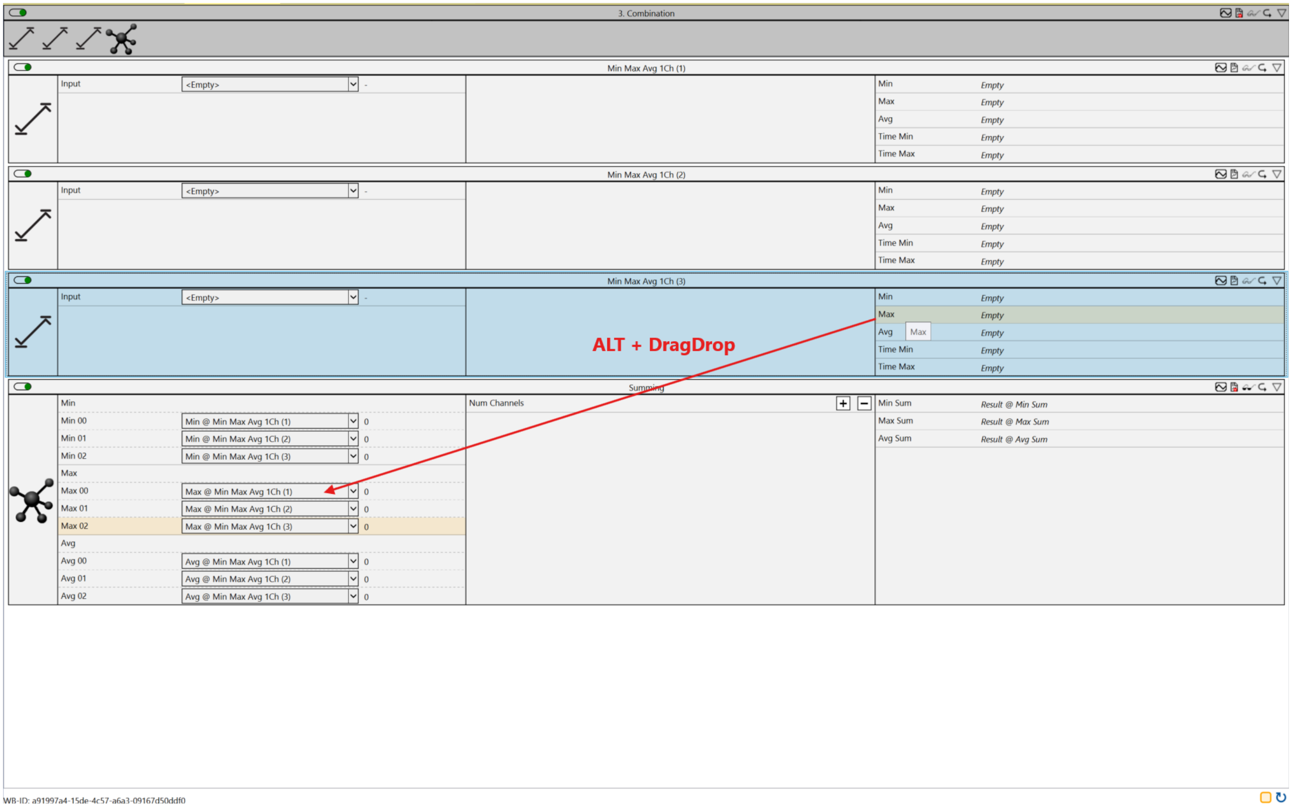
Min Sum
Input 00 Min 00 @ Summing -
Input 01 Min 01 @ Summing -
Num Channels
Mathematical Operator
Use Absolute Values □
Result *Empty*

Max Sum
Input 00 Input 00 @ Summing -
Input 01 Input 01 @ Summing -
Num Channels
Mathematical Operator
Use Absolute Values □
Result *Empty*

Avg Sum
Input 00 Input 00 @ Summing -
Input 01 Input 01 @ Summing -
Num Channels
Mathematical Operator
Use Absolute Values □
Result *Empty*

You will now see group names at the network inputs. You assign these as the exact names of the subsequent outputs. In our example, you form the groups Min, Max and Avg.

When using the template, it is again a single operation with the ALT key pressed and a drag and drop from one of the outputs to an input of the template. This searches all algorithms in the network and automatically links all outputs with the same name as the group names.



3. Combination

Min Max Avg 1Ch (1)

Input <Empty> -

Min	Empty
Max	Empty
Avg	Empty
Time Min	Empty
Time Max	Empty

Min Max Avg 1Ch (2)

Input <Empty> -

Min	Empty
Max	Empty
Avg	Empty
Time Min	Empty
Time Max	Empty

Min Max Avg 1Ch (3)

Input <Empty> -

Min	Empty
Max	Empty
Avg	Empty
Time Min	Empty
Time Max	Empty

ALT + DragDrop

Summing

Min		
Min 00	Min @ Min Max Avg 1Ch (1)	0
Min 01	Min @ Min Max Avg 1Ch (2)	0
Min 02	Min @ Min Max Avg 1Ch (3)	0
Max		
Max 00	Max @ Min Max Avg 1Ch (1)	0
Max 01	Max @ Min Max Avg 1Ch (2)	0
Max 02	Max @ Min Max Avg 1Ch (3)	0
Avg		
Avg 00	Avg @ Min Max Avg 1Ch (1)	0
Avg 01	Avg @ Min Max Avg 1Ch (2)	0
Avg 02	Avg @ Min Max Avg 1Ch (3)	0

Num Channels

Min Sum	Result @ Min Sum
Max Sum	Result @ Max Sum
Avg Sum	Result @ Avg Sum

WB-ID: a91997a4-15de-4c57-a6a3-09167d50ddf0

6.3.3 Enable Disable

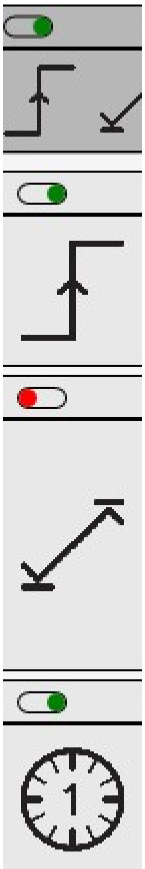
The TwinCAT Analytics Workbench offers the functionality to switch individual algorithms or even complex networks on or off in a data analysis. This document explains the use and advantages of this function.

The function allows users to specifically optimize or even deactivate individual parts of an analysis in order to influence the behavior of the analysis as a whole.

Advantages and application examples for the deactivation of modules in an analysis:

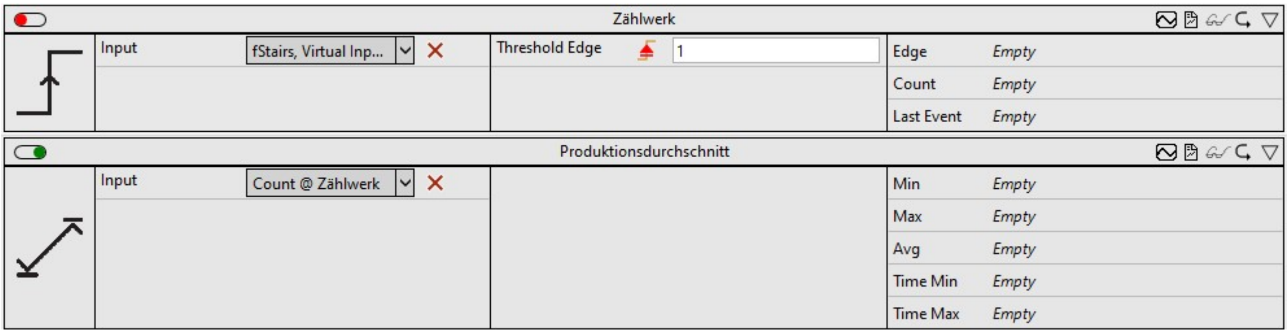
- Improved precision:
  The ability to turn algorithms and networks on or off in the data analysis allows users to improve the accuracy of the analysis. If an algorithm is not working optimally or a certain network area is not relevant, it can be disabled to improve the overall precision of the analysis.
- Optimizing resource use:
  Turning algorithms or networks on or off can also help optimize resource use. For example, if an algorithm requires high computational power, it can be turned off as needed to focus resources on other areas of analysis.
- Flexibility:
  The ability to turn algorithms and networks on or off in data analysis provides users with flexibility and control over analysis. Users can customize the analysis to meet the specific needs of their data.
- Debugging:
  The function also allows users to debug problems in the analysis. If a particular part of the analysis is faulty, the affected algorithm or network can be disabled to isolate and fix the problem.

Switching modules on or off is controlled by the control elements in the upper left corner of the respective module.



Alignment	Color	Status
Right	Green	Enabled
Left	Red	Disabled

For linked modules, the chain of links is passed through and inputs linked to a disabled module are indicated by a red X at the input.
Disabled Link between two analysis algorithms:



Zählwerk

Input: fStairs, Virtual Inp... ✕

Threshold Edge: 1

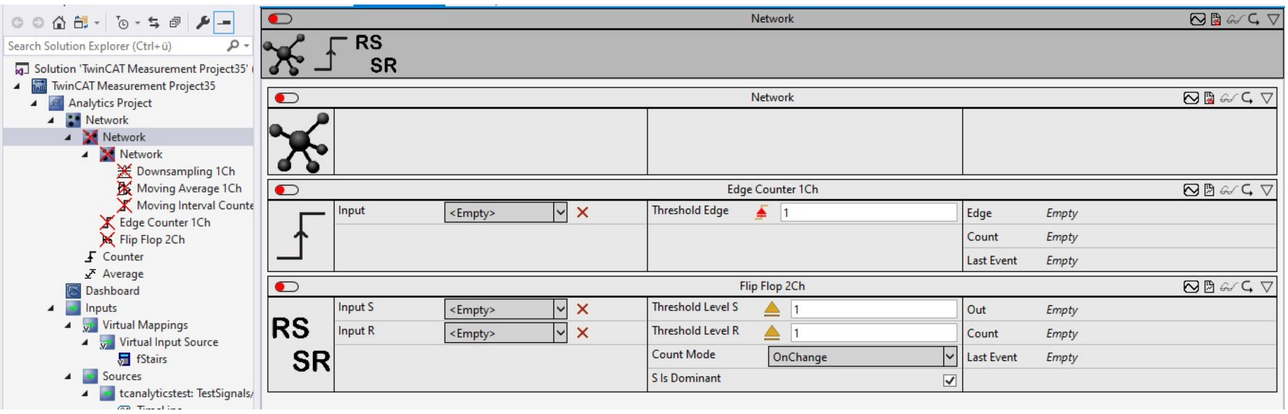
Edge	*Empty*
Count	*Empty*
Last Event	*Empty*

Produktionsdurchschnitt

Input: Count @ Zählwerk ✕

Min	*Empty*
Max	*Empty*
Avg	*Empty*
Time Min	*Empty*
Time Max	*Empty*

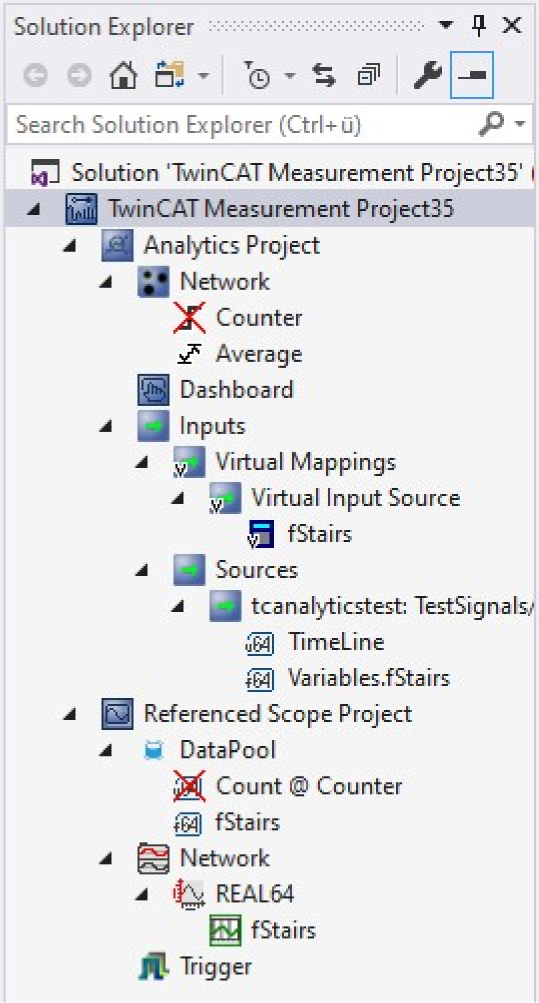
If a network is deactivated, the processing of all algorithms below the network and its subnetworks (recursive) is also deactivated.
Deactivated network elements with their subelements:



Input <Empty>
Threshold Edge 1
Edge Empty
Count Empty
Last Event Empty

Input S <Empty>
Input R <Empty>
Threshold Level S 1
Threshold Level R 1
Count Mode OnChange
S Is Dominant ☑
Out Empty
Count Empty
Last Event Empty

If you use a Referenced Scope (see Interaction with Scope [▶ 344]), linked elements of the Scope are also affected by the deactivation.
Deactivating the algorithm also deactivates the associated Scope acquisition:

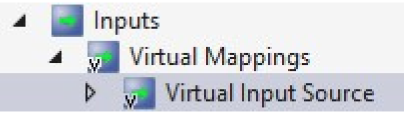


6.3.4 Static Values

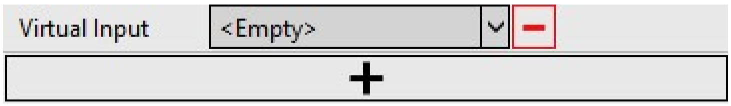
Static values are often used in analysis algorithms to store constants that should remain unchanged during the execution of the algorithm. These static values can serve, for example, as thresholds, scaling factors, or other mathematical constants used in the algorithm.

The use of static values in analysis algorithms offers several advantages. First, the performance of the algorithm can be improved by avoiding computations that otherwise, would have to be performed during each execution of the algorithm. Instead, the algorithm can access the static value set during the initialization of the algorithm. On the other hand, the readability and comprehensibility of the algorithm can be improved by structuring the input value more clearly and providing constants with meaningful designations.

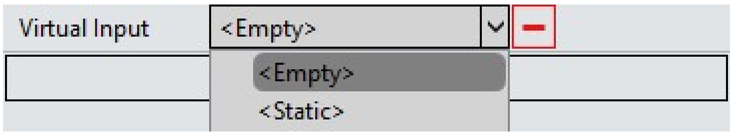
Static values are sorted in the TwinCAT Analytics Workbench under the Virtual Inputs in order to generate them at a central location and manage them later.

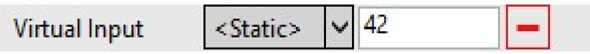


In the editor of the Virtual Input Source you can create a new Virtual Input or edit an existing one by using **+**.

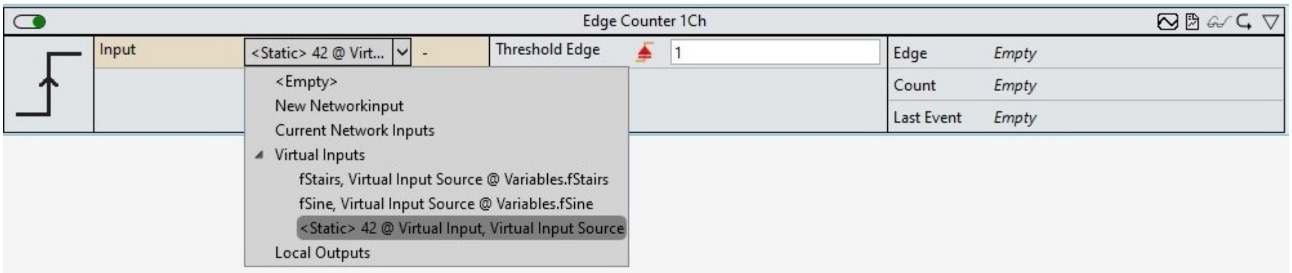

Virtual Input <Empty>

Afterwards, the virtual input can be set to a static value (here 42) via the drop-down menu.


Virtual Input <Empty>


Virtual Input <Static> 42

This static value is now selectable throughout the configuration via the Input module.


Edge Counter 1Ch

Input	<Static> 42 @ Virt...	Threshold Edge	1	Edge Empty
				Count Empty
				Last Event Empty

<Empty>
New Networkinput
Current Network Inputs
Virtual Inputs
fStairs, Virtual Input Source @ Variables.fStairs
fSine, Virtual Input Source @ Variables.fSine
<Static> 42 @ Virtual Input, Virtual Input Source
Local Outputs

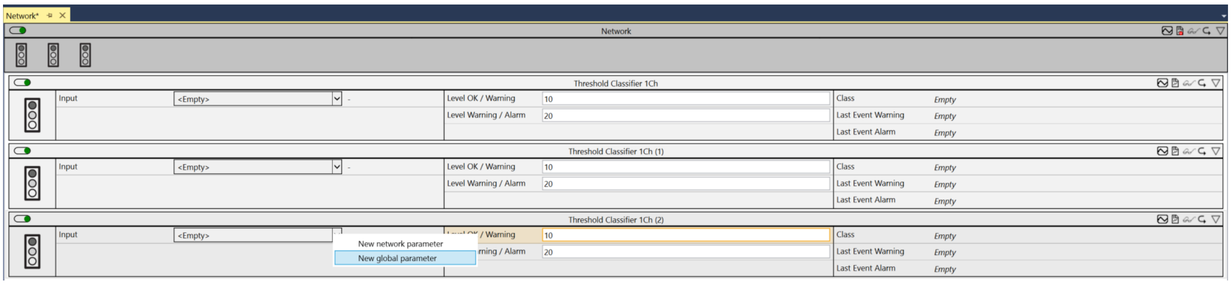
Static values can also be used on network inputs. Here the linking with the dependent inputs of the individual modules would run exactly the same as the behavior with incoming data inputs.

Static values in analysis algorithms should be used cautiously. If the data or parameters to which the algorithm is applied change, it may be necessary to adjust the static values to ensure optimal performance of the algorithm. Therefore, static values should be checked regularly and adjusted if necessary to ensure that the algorithm is working correctly and effectively.

6.3.5 Global parameters

With the option of global parameters, fixed values such as a cost factor for energy management can be defined, which can occur multiple times in different algorithms.

1. To create a new global parameter, right-click on an existing parameter and select the **New global parameter** option.


Network* — Network

Threshold Classifier 1Ch

Input	<Empty>	Level OK / Warning	10	Class	Empty
		Level Warning / Alarm	20	Last Event Warning	Empty
				Last Event Alarm	Empty

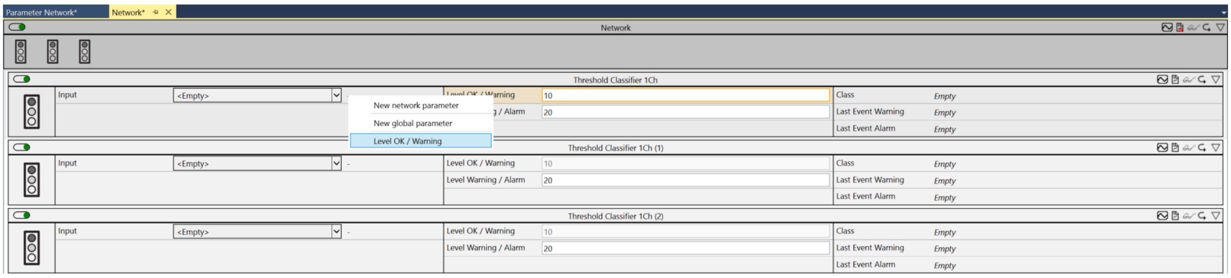
Threshold Classifier 1Ch (1)

Input	<Empty>	Level OK / Warning	10	Class	Empty
		Level Warning / Alarm	20	Last Event Warning	Empty
				Last Event Alarm	Empty

Threshold Classifier 1Ch (2)

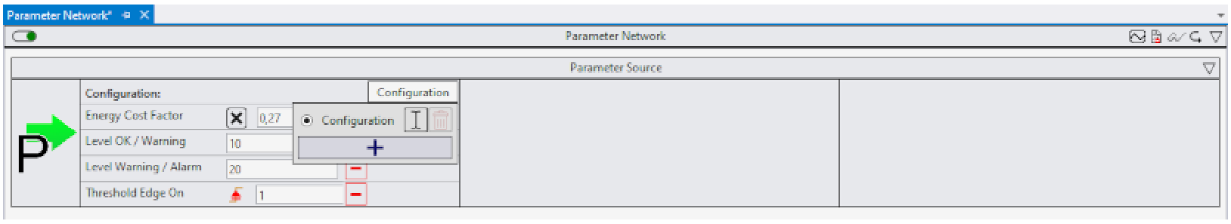
Input	<Empty>	Level OK / Warning	10	Class	Empty
		Level Warning / Alarm	20	Last Event Warning	Empty
				Last Event Alarm	Empty

2. Existing global parameters can then be selected via the same context menu on other algorithms.

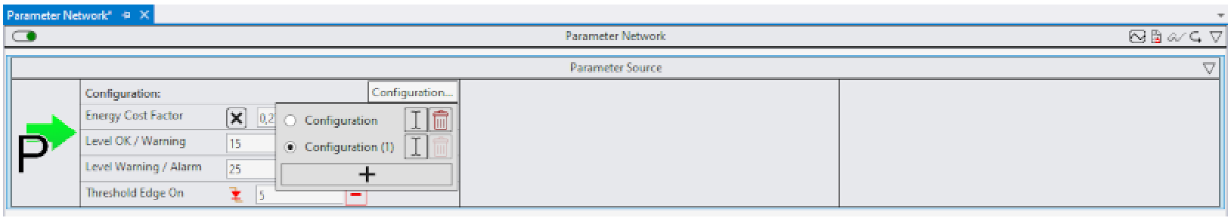


Level OK / Warning	10	Class	Empty
Level Warning / Alarm	20	Last Event Warning	Empty
		Last Event Alarm	Empty

3. As soon as the first global parameter has been created, a new **Parameter Network** is created. This allows you to set parameters in one central location. It is also possible to define several configurations of parameters that can be switched.

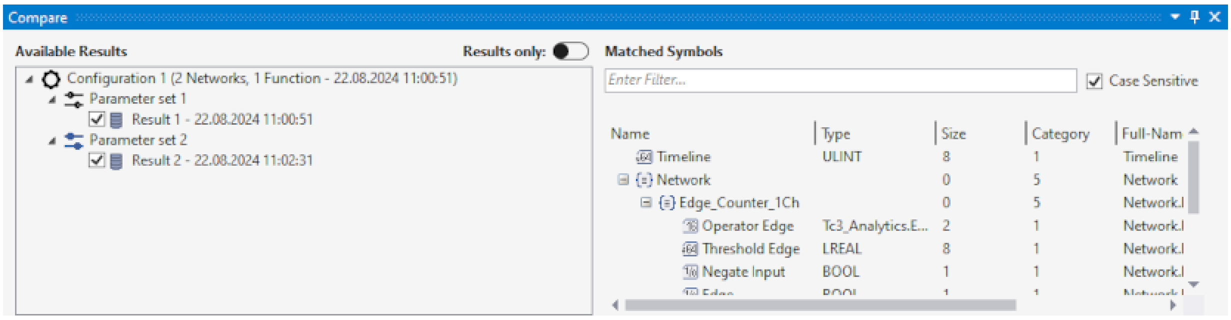


Configuration:	
Energy Cost Factor	0,27
Level OK / Warning	10
Level Warning / Alarm	20
Threshold Edge On	1



Configuration:	
Energy Cost Factor	0,2
Level OK / Warning	15
Level Warning / Alarm	25
Threshold Edge On	5

4. If the project is executed successively with different parameter configurations, a new **Parameter Set** with the corresponding result data set is created in the Analytics Diary [▸ 524] under the existing project configuration.



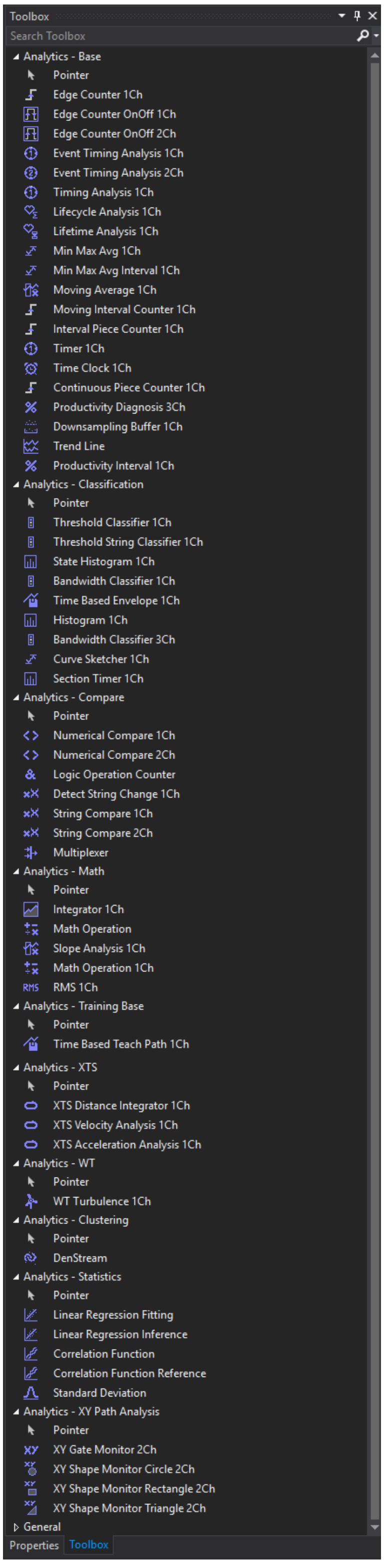
- Configuration 1 (2 Networks, 1 Function - 22.08.2024 11:00:51)
  - Parameter set 1
    - ☑ Result 1 - 22.08.2024 11:00:51
  - Parameter set 2
    - ☑ Result 2 - 22.08.2024 11:02:31

Name	Type	Size	Category	Full-Nam
Timeline	ULINT	8	1	Timeline
Network		0	5	Network
Edge_Counter_1Ch		0	5	Network.l
Operator Edge	Tc3_Analytics.E...	2	1	Network.l
Threshold Edge	LREAL	8	1	Network.l
Negate Input	BOOL	1	1	Network.l

⇨ No further changes may be made to the project, otherwise a new project configuration will also be created.

6.4 Algorithms

The TwinCAT Analytics Workbench configurator and the TwinCAT Analytics Service Tool include various analysis algorithms that can be found in the toolbox. If the more than 50 algorithms do not meet the requirements, user-specific algorithms can be developed using the Analytics Lambda functions. If the toolbox is empty, select the Analytics project to see the algorithms.



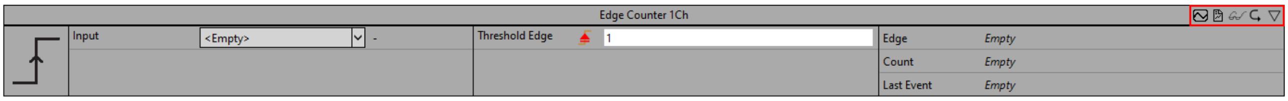
Toolbox
Search Toolbox

- Analytics - Base
  - Pointer
  - Edge Counter 1Ch
  - Edge Counter OnOff 1Ch
  - Edge Counter OnOff 2Ch
  - Event Timing Analysis 1Ch
  - Event Timing Analysis 2Ch
  - Timing Analysis 1Ch
  - Lifecycle Analysis 1Ch
  - Lifetime Analysis 1Ch
  - Min Max Avg 1Ch
  - Min Max Avg Interval 1Ch
  - Moving Average 1Ch
  - Moving Interval Counter 1Ch
  - Interval Piece Counter 1Ch
  - Timer 1Ch
  - Time Clock 1Ch
  - Continuous Piece Counter 1Ch
  - Productivity Diagnosis 3Ch
  - Downsampling Buffer 1Ch
  - Trend Line
  - Productivity Interval 1Ch
- Analytics - Classification
  - Pointer
  - Threshold Classifier 1Ch
  - Threshold String Classifier 1Ch
  - State Histogram 1Ch
  - Bandwidth Classifier 1Ch
  - Time Based Envelope 1Ch
  - Histogram 1Ch
  - Bandwidth Classifier 3Ch
  - Curve Sketcher 1Ch
  - Section Timer 1Ch
- Analytics - Compare
  - Pointer
  - Numerical Compare 1Ch
  - Numerical Compare 2Ch
  - Logic Operation Counter
  - Detect String Change 1Ch
  - String Compare 1Ch
  - String Compare 2Ch
  - Multiplexer
- Analytics - Math
  - Pointer
  - Integrator 1Ch
  - Math Operation
  - Slope Analysis 1Ch
  - Math Operation 1Ch
  - RMS 1Ch
- Analytics - Training Base
  - Pointer
  - Time Based Teach Path 1Ch
- Analytics - XTS
  - Pointer
  - XTS Distance Integrator 1Ch
  - XTS Velocity Analysis 1Ch
  - XTS Acceleration Analysis 1Ch
- Analytics - WT
  - Pointer
  - WT Turbulence 1Ch
- Analytics - Clustering
  - Pointer
  - DenStream
- Analytics - Statistics
  - Pointer
  - Linear Regression Fitting
  - Linear Regression Inference
  - Correlation Function
  - Correlation Function Reference
  - Standard Deviation
- Analytics - XY Path Analysis
  - Pointer
  - XY Gate Monitor 2Ch
  - XY Shape Monitor Circle 2Ch
  - XY Shape Monitor Rectangle 2Ch
  - XY Shape Monitor Triangle 2Ch
- General

Properties Toolbox

Currently, there are eleven different groups of algorithms: Analytics-Base, Analytics-Classification, Analytics-Compare, Analytics-Math, Analytics-Training Base, Analytics-XTS, Analytics-WT, Analytics-XY Path Analysis, Analytics-Clustering, Analytics-Statistics and Analytics-C++ Lambda Functions. User-specific algorithms can be developed using the C++ lambda functions (see C++ lambda functions [▶ 270]). In addition, it is possible to use algorithms from other libraries that are implemented in the Analytics Toolbox (see Algorithms from other TwinCAT libraries [▶ 300]). The Analytics Custom Toolbox Cleaner [▶ 299] can be used to clean up the toolbox with regard to user-specific elements, such as the templates or the C++ lambda functions.

Each algorithm has the same five icons in the upper right corner:



Edge Counter 1Ch

Input <Empty> -

Threshold Edge 1

Edge Empty

Count Empty

Last Event Empty

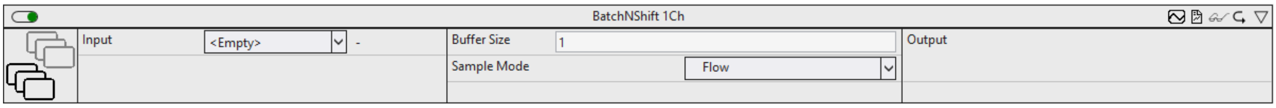
- **Include in Referenced Scope:** if you click on the *Included in Referenced Scope* icon, the algorithm outputs are added to a referenced Scope project.
- **Include in Report:** If you click on the *Include in Report* icon, the algorithm with the configured visualization will be integrated into a manual report.
- **Eyeglasses:** If you click on the eyeglasses icon you can see the optional parameter *Enable Execution*. You can select a Boolean signal for this parameter, so that the algorithm is just active, if the value of the selected signal is *TRUE*.
- **Reset arrow:** If you click on the arrow the output values of the specific algorithm will be reset.
- **Minimize arrow:** If you click on the minimize arrow on the right, the algorithm will be folded.

The different algorithm groups are described below.

6.4.1 Analytics - Base

The algorithms of the category *Analytics-Base* provide base functionalities for analyzing process and application data. For example threshold detection, timing analysis or calculation of minimum, maximum and average values.

6.4.1.1 BatchNShift 1Ch



BatchNShift 1Ch

Input <Empty> -

Buffer Size 1

Sample Mode Flow

Output

The *BatchNShift 1Ch* buffers the values of the input signal according to the buffer size and the sample mode. The number of output channels in which the buffered input values are stored corresponds to the buffer size. With the help of the sample mode it is possible to distinguish between two different operating modes of the algorithm. If the sample mode *Flow* is selected, a ring buffer or shift register is realized (Shift). The values are written to the buffer one after the other and shifted by one position of the buffer in each cycle. If the buffer is full, the last value falls out. In the *Wait* mode, the buffer is instead completely emptied and filled with new values whenever it is completely full, so that the values are processed in the form of batches (batch). At the beginning of an analysis, the system also waits until the buffer is completely filled before writing the values to the buffer. Therefore, the function block supplies valid values only from the cycle (*BufferSize + 1)*.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Buffer Size:** specifies the size of the buffer and thus the number of values that are stored. The number of output channels equals the buffer size.
- **Sample Mode:** the values from the buffer can be passed to the output channels in two different modes:
  - *Flow*: the buffer is filled like a ring buffer. At the start of the analysis all output values are set to zero. Each change to the ring buffer is transferred to the output channels immediately. The New Result flag is set to TRUE, once all output channels got assigned a value and is always true, when a new value is saved in the buffer.
  - *Wait*: at the start of the analysis or after reset all output channels are set to zero. Only when the internal buffer is full, these values are transferred to the output channels and the New Result flag is set to TRUE. These values stay as output values until all the values in the internal buffer are renewed. Only then they are transferred to the output channels.

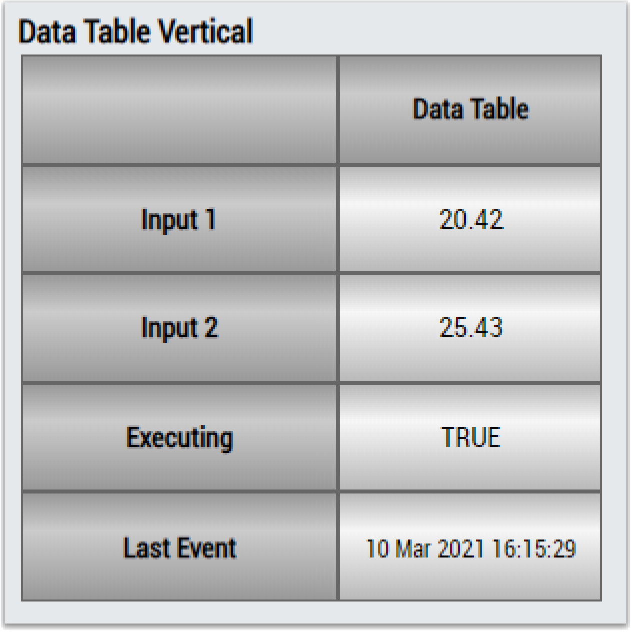
Output values

- **Output Value 00..n**: results of the BatchNShift buffer according to the Sample Mode. Each output channel represents a buffer storage space.

Standard HMI Controls

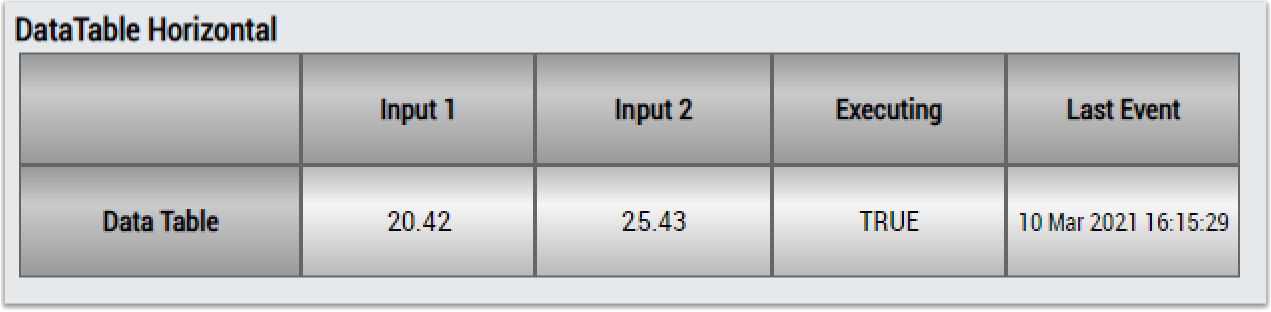
For the *BatchNShift 1Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes the output values *Output Value 00..n*



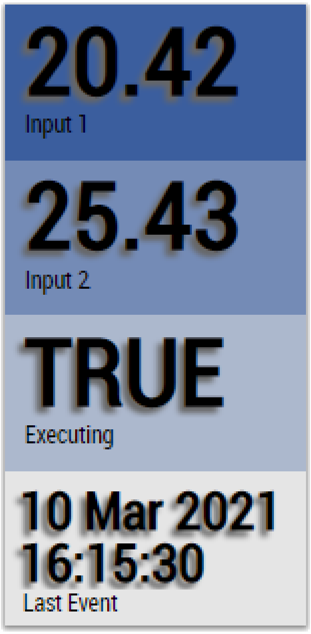
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

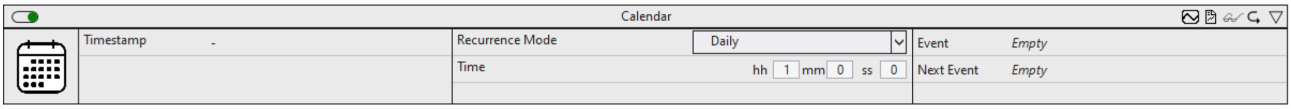
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *BatchNShift 1Ch* algorithm using the Mapping Wizard [▶ 432].

6.4.1.2 Calendar



Calendar

Timestamp -

Recurrence Mode: Daily

Time: hh 1 mm 0 ss 0

Event: *Empty*

Next Event: *Empty*

The *Calendar* creates a recurring event at certain times. The repeat mode can be set individually. The modes of daily, weekly, monthly or annual repetition are available.
A timestamp is required as a reference value, as the algorithm needs a time context in which to work. This reference timestamp is set automatically, if there is another algorithm in the configuration. Therefore, it is not possible to use the *Calendar* individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Recurrence Mode:**
  Daily: Daily recurrence. The time of the event can be set.
  Weekly: Weekly recurrence. The time of the event and the days of the week on which the event is to be generated can be set.
  Monthly: Monthly recurrence. The time of the event and the recurrence type can be set.
  Yearly: Annual recurrence. The time of the event, the recurrence type and the months in which the event is to be created can be set.
- **Time:** Time of the event
- **Day Of Week Mask:** Days of the week on which an event is to be created.
- **Recurrence Specification:**
  Specific Day: The event should be created on a specific day. This can be set using the *Day* parameter.
  First: The event should be created on the first specified day of the month. The specification is made via the *Day Of Week* parameter.
  Last: The event should be created on the last specified day of the month. Specification takes place via the *Day Of Week* parameter
- **Day:** Day on which an event is to be created.
- **Day Of Week:** Day of the week of the event
  Unspecified: The event is created on the first or last day of the month, depending on the *Recurrence Specification* parameter.
  Monday..Sunday: The event is generated on the first or last set weekday of the month, depending on the *Recurrence Specification* parameter.
- **Month Of Year Mask:** Months in which an event is to be created.

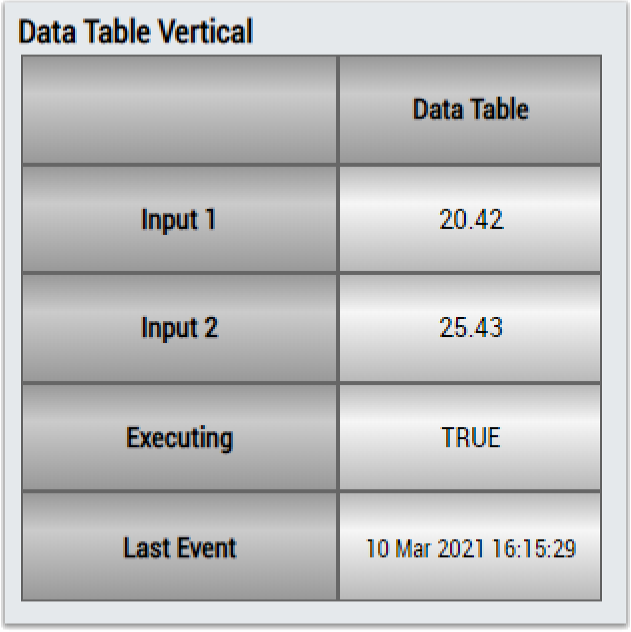
Output values

- **Event:** Indicates *TRUE* at the time of the event, otherwise *FALSE*.
- **Next Event:** Displays the time remaining until the next event.

Standard HMI Controls

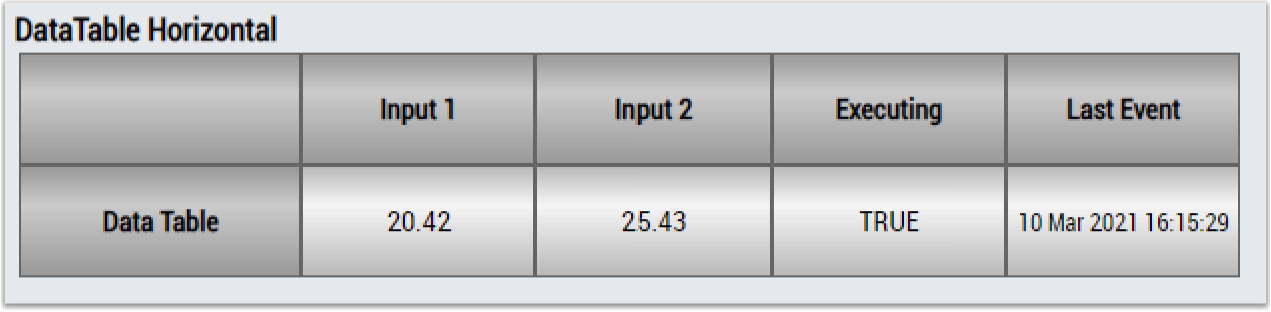
For the Calendar algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values: Event, Next Event.



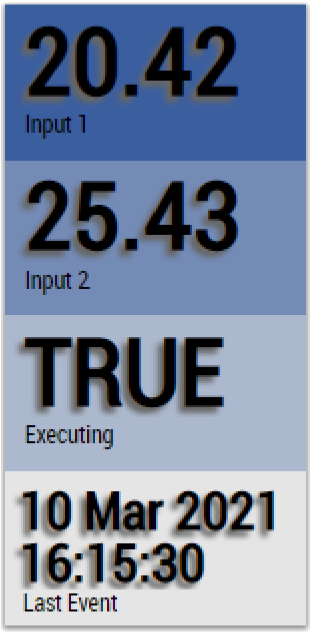
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

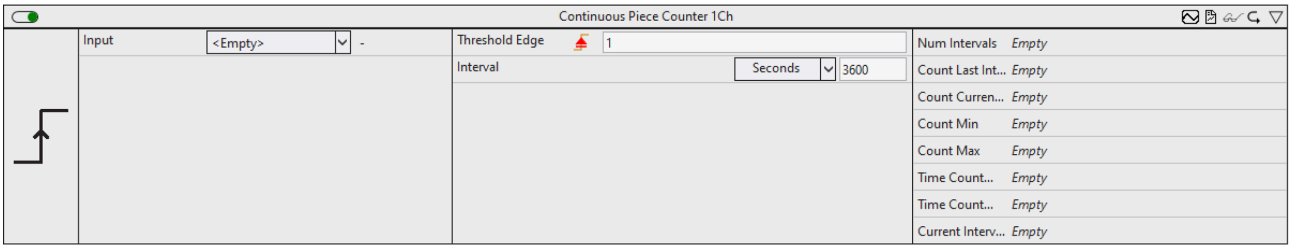
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Calendar* algorithm using the Mapping Wizard [▸ 432].

6.4.1.3 Continuous Piece Counter 1Ch



Continuous Piece Counter 1Ch

Input: <Empty> -

Threshold Edge: 1

Interval: Seconds 3600

Num Intervals	*Empty*
Count Last Int...	*Empty*
Count Curren...	*Empty*
Count Min	*Empty*
Count Max	*Empty*
Time Count...	*Empty*
Time Count...	*Empty*
Current Interv...	*Empty*

The *Continuous Piece Counter 1Ch* counts the number of pieces within the configured interval. The counter is increased when the signal of the input channel passes the configured edge at a specific threshold. The calculation restarts when the time of the interval has elapsed. The algorithm provides the amount of pieces, the minimal and the maximal number of pieces as well as the time values of minimum and maximum.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The counter increments when the signal passes this threshold.
- **Interval:** Time interval in which the values are to be calculated.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

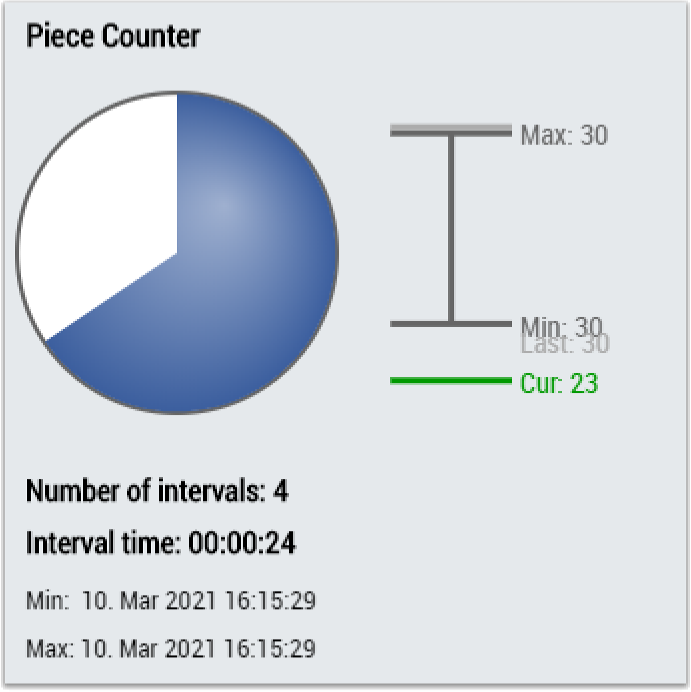
Output Values

- **Num Interval:** Shows the number of intervals.
- **Count Last Interval:** Shows the amount of pieces in the last interval.
- **Count Current Interval:** Shows the amount of pieces in the current interval.
- **Count Min:** Shows the minimal number of pieces in an interval.
- **Count Max:** Shows the maximal number of pieces in an interval.
- **Time Count Min:** Shows the time value of the minimum.
- **Time Count Max:** Shows the time value of the maximum.
- **Current Interval Time:** Shows the time of the current interval.

Standard HMI Controls

For the Continuous Piece Counter 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The PieceCounter control visualizes the output values Num Intervals, Count Last Interval, Count Current Interval, Count Min, Count Max, Time Count Min, Time Count Max and Time Current Interval.



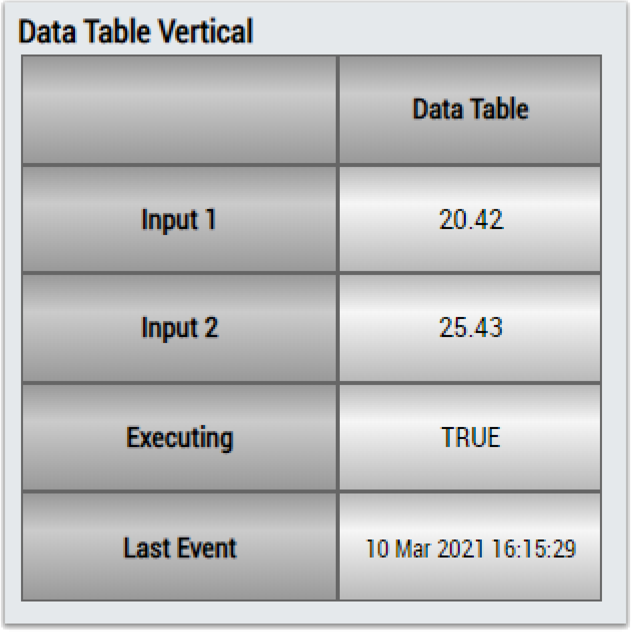
Number of intervals: 4

Interval time: 00:00:24

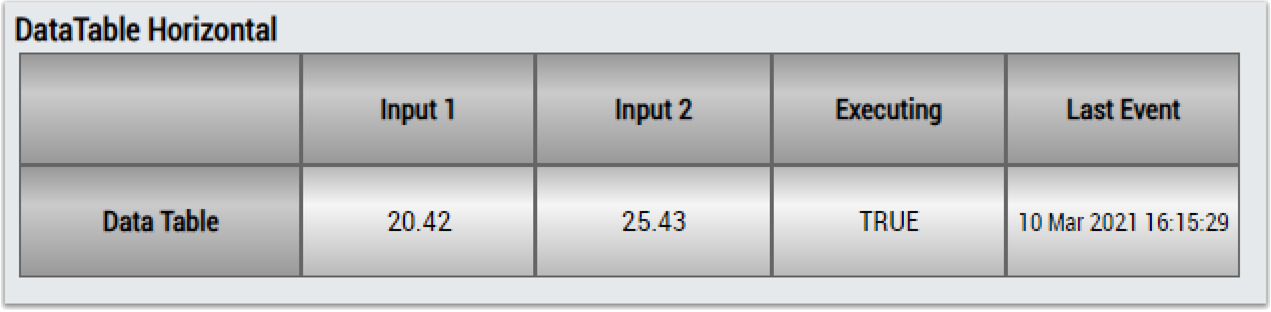
Min: 10. Mar 2021 16:15:29

Max: 10. Mar 2021 16:15:29

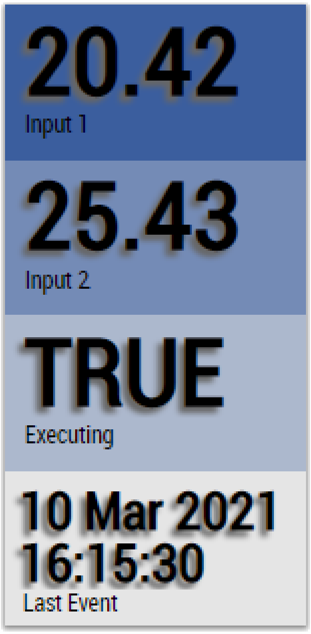
2. The Table Control or Multivalue Control visualizes all output values: Num Intervals, Count Last Interval, Count Current Interval, Count Min, Count Max, Time Count Min, Time Count Max, Time Current Interval.


Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29


DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

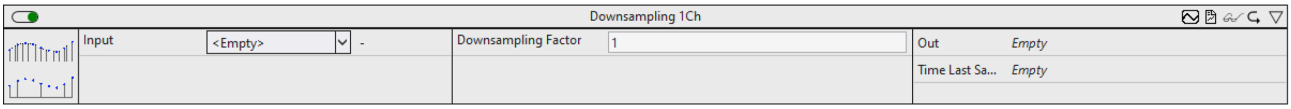
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Continuous Piece Counter 1Ch algorithm using the Mapping Wizard [▸ 432].

6.4.1.4 Downsampling 1Ch



Input <Empty> -

Downsampling Factor 1

Out *Empty*

Time Last Sa... *Empty*

Downsampling 1Ch processes the values of the input channel with a configurable downsampling factor. This achieves downsampling so that the output signal is a representation of the input signal at a lower sampling rate. This can be useful, for example, to better identify trends or to perform subsequent compression of highly sampled signals if only lower sampling rates are required within the analysis. This is a simple way to increase the performance of the analysis.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Downsampling Factor:** the factor used for downsampling. For example, if the downsampling factor is 100, only every 100th value is saved. The sample time is thus 100 times the original cycle time, which lies between two sampled data points. If the downsampling factor is set to one, all values are buffered.

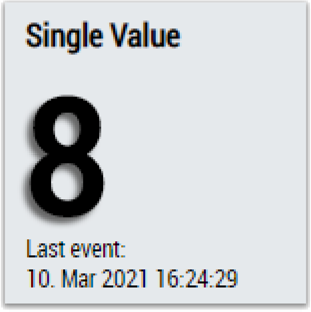
Output values

- **Output Value**: output signal with the sampling rate lower by the downsampling factor.

Standard HMI Controls

For the *Downsampling 1Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue Control visualizes the output value *Output Value*.



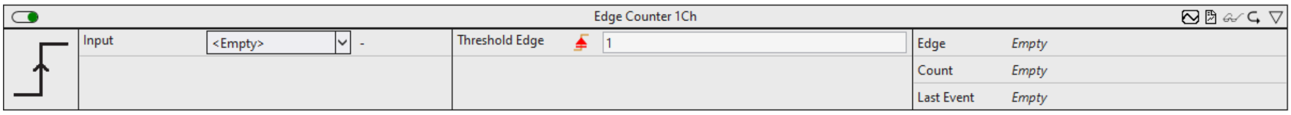
Single Value

8

Last event:
10. Mar 2021 16:24:29

Alternatively, customer-specific HMI controls can be mapped in the *Downsampling 1Ch* algorithm using the Mapping Wizard [▶ 432].

6.4.1.5 Edge Counter 1Ch



Edge Counter 1Ch

Input: <Empty> -

Threshold Edge: 1

Edge	*Empty*
Count	*Empty*
Last Event	*Empty*

The *Edge Counter 1Ch* counts the amount of raised events. An event is raised when the signal of the input channel passes the configured edge at a specific threshold.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

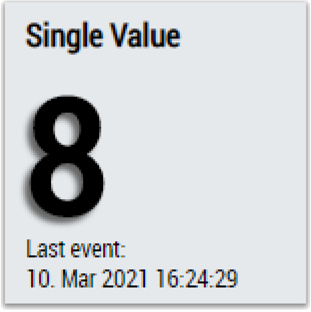
Output values

- **Edge:** Indicates *TRUE* at the time the event is triggered, otherwise *FALSE*.
- **Count:** Counts the number of triggered events.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Edge Counter 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue control visualizes the output values Count and Last Event.

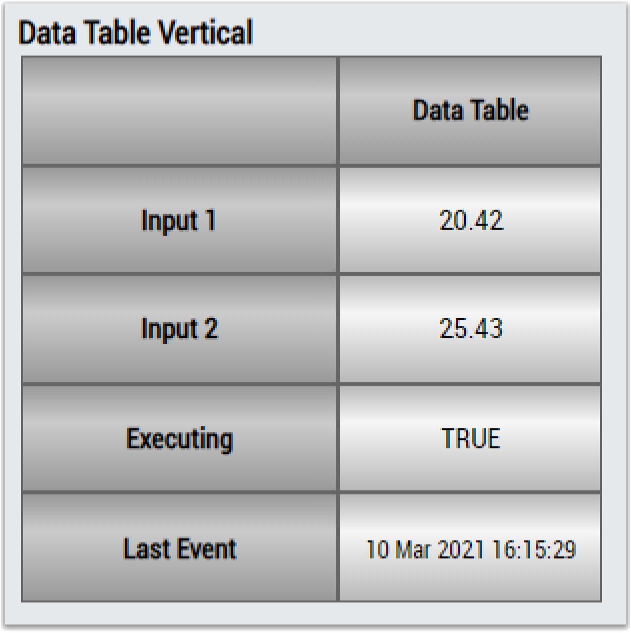


Single Value

8

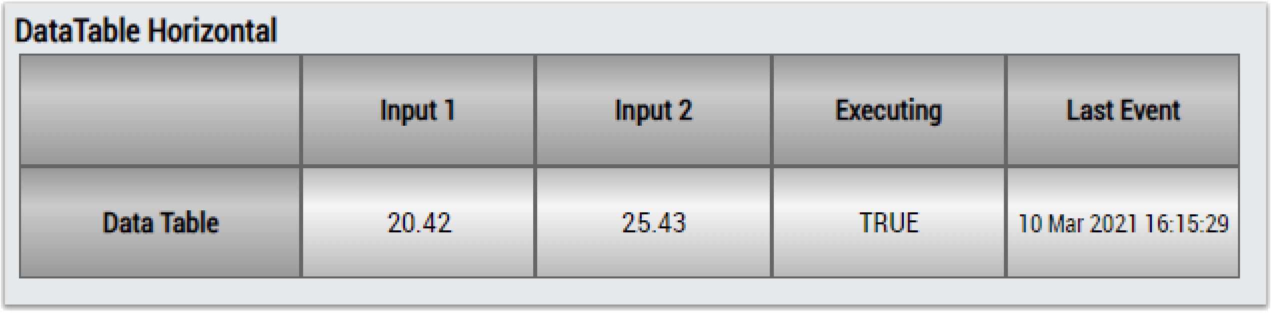
Last event:
10. Mar 2021 16:24:29

2. The Table Control or Multivalue Control visualizes all output values: Edge, Count, Last Event.



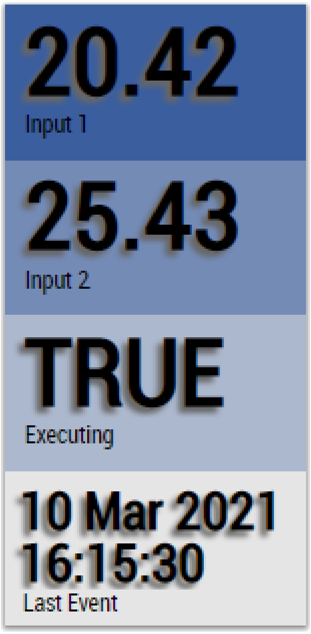
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

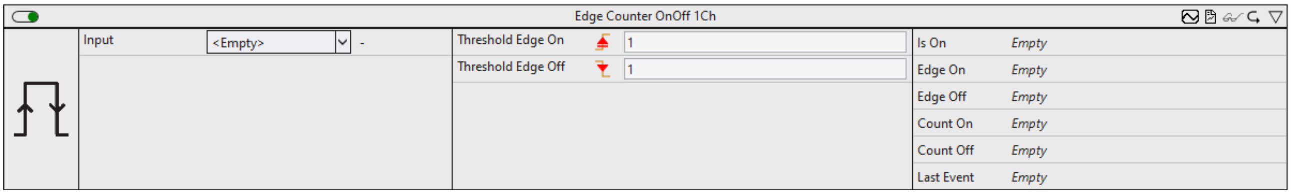
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, custom HMI controls can be mapped in the Edge Counter 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.6 Edge Counter On Off 1Ch



Edge Counter OnOff 1Ch

Input	<Empty>	Threshold Edge On	1	Is On	Empty
		Threshold Edge Off	1	Edge On	Empty
				Edge Off	Empty
				Count On	Empty
				Count Off	Empty
				Last Event	Empty

The *Edge Counter On Off 1Ch* counts the amount of raised on- and off-events. An on-event is raised when the signal of the input channel passes the configured edge at a specific on-threshold and an off-event is raised when the off-threshold is passed by the same signal.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold On:** Threshold of the signal at the respective edge. The On event is triggered when the signal passes this threshold.
- **Threshold Off:** Threshold of the signal at the respective edge. The Off event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

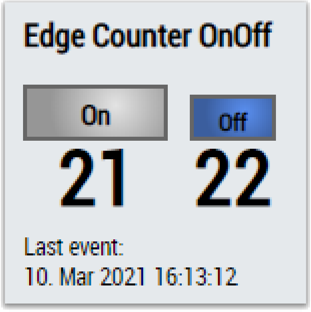
Output values

- **Is On:** Indicates *TRUE* within the timespan between the On event and the Off result, otherwise *FALSE*.
- **Edge On:** Indicates TRUE on a rising edge.
- **Edge Off:** Indicates TRUE on a falling edge.
- **Count On:** Counts the number of triggered On events.
- **Count Off:** Counts the number of triggered Off events.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Edge Counter On Off 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The EdgeCounterOnOff control visualizes the output values Is On, Count On, Count Off and Last Event.



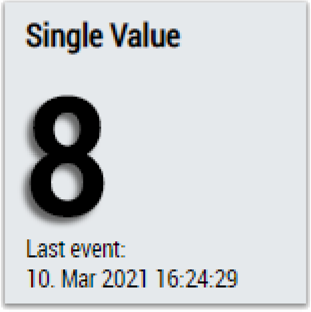
Edge Counter OnOff

On 21

Off 22

Last event:
10. Mar 2021 16:13:12

2. The SingleValue control visualizes the output values Count On and Last Event.

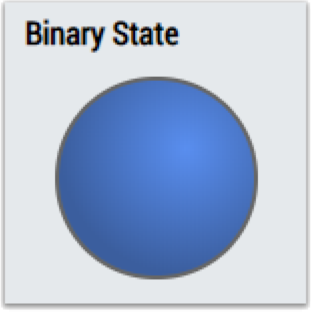


Single Value

8

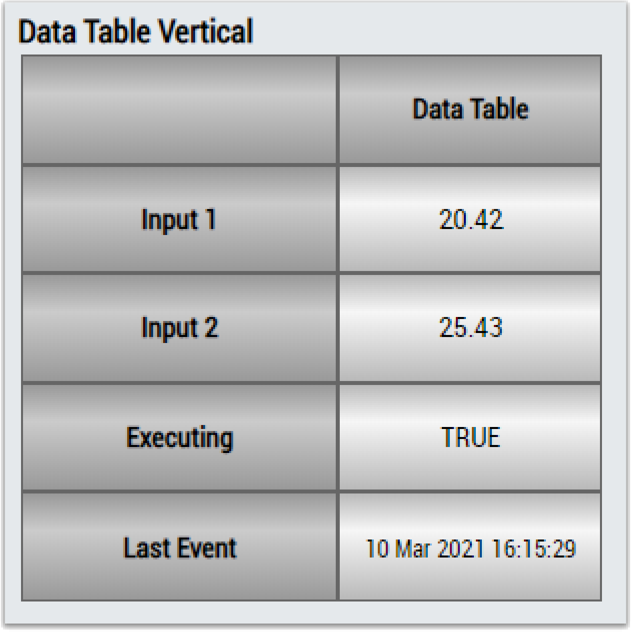
Last event:
10. Mar 2021 16:24:29

3. The BinaryState control visualizes the output value Is On.



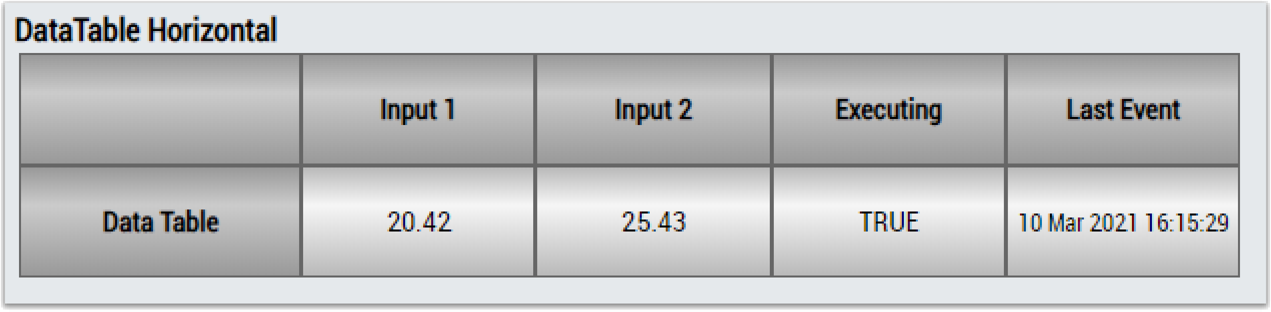
Binary State

4. The Table Control or Multivalue Control visualizes all output values: Flanks (Is On, Edge On, Edge Off), Count On, Count Off, Last Event.



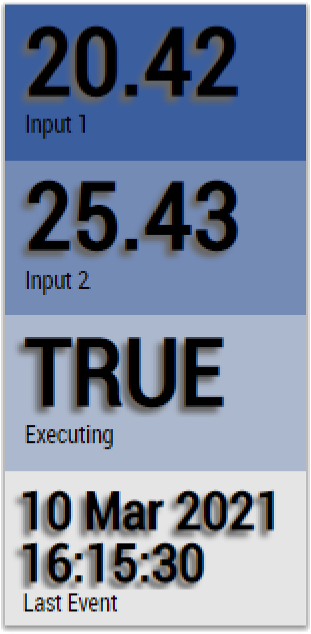
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

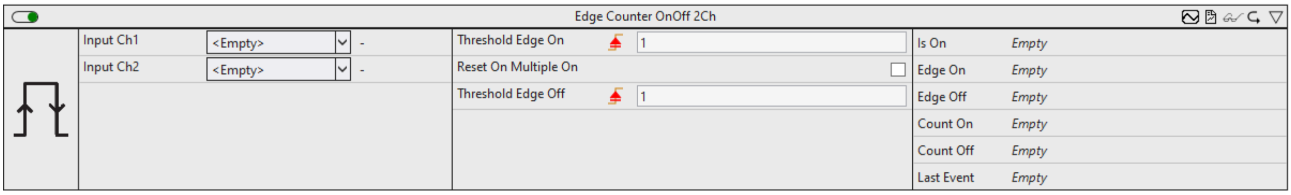
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, custom HMI controls can be mapped in the Edge Counter On Off 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.7 Edge Counter On Off 2Ch



Edge Counter OnOff 2Ch

Input Ch1	<Empty> -	Threshold Edge On	1	Is On	Empty
Input Ch2	<Empty> -	Reset On Multiple On	□	Edge On	Empty
		Threshold Edge Off	1	Edge Off	Empty
				Count On	Empty
				Count Off	Empty
				Last Event	Empty

The *Edge Counter On Off 2Ch* counts the amount of raised on- and off-events. An on-event is raised when the signal of the first input channel passes the configured edge at a specific on-threshold and an off-event is raised when the off-threshold is passed by the signal of the second channel.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold On:** Threshold of the signal at the respective edge. The On event is triggered when the signal passes this threshold.
- **Reset On Multiple On:** If the checkbox is checked, the "Count On" counter increments with every On event. Otherwise, the On events are only counted after a counter reset (Off event).
- **Threshold Off:** Threshold of the signal at the respective edge. The Off event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

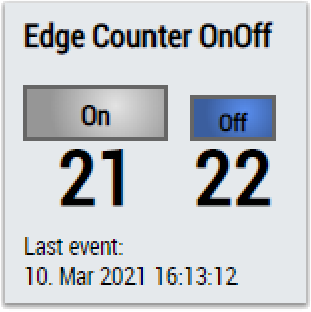
Output values

- **Is On:** Indicates *TRUE* within the timespan between the On event and the Off result, otherwise *FALSE*.
- **Edge On:** Indicates TRUE on a rising edge.
- **Edge Off:** Indicates TRUE on a falling edge.
- **Count On:** Counts the number of triggered On events.
- **Count Off:** Counts the number of triggered Off events.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Edge Counter On Off 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The EdgeCounterOnOff control visualizes the output values Is On, Count On, Count Off and Last Event.



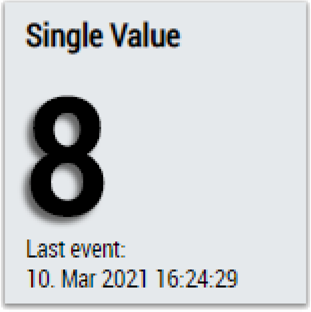
Edge Counter OnOff

On 21

Off 22

Last event:
10. Mar 2021 16:13:12

2. The SingleValue control visualizes the output values Count On and Last Event.

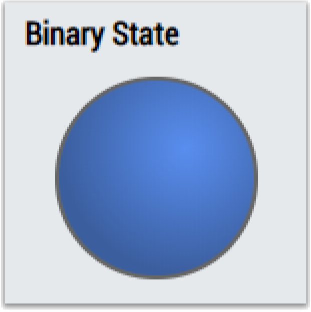


Single Value

8

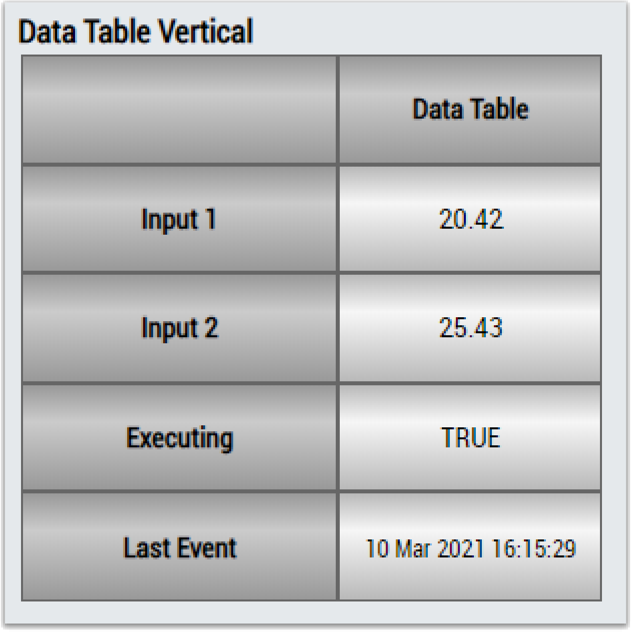
Last event:
10. Mar 2021 16:24:29

3. The BinaryState control visualizes the output value Is On.



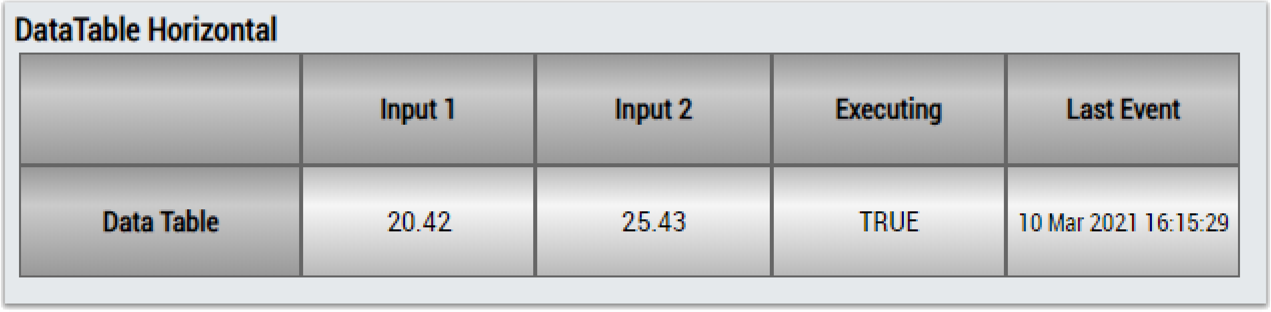
Binary State

4. The Table Control or Multivalue Control visualizes all output values: Flanks (Is On, Edge On, Edge Off), Count On, Count Off, Last Event.



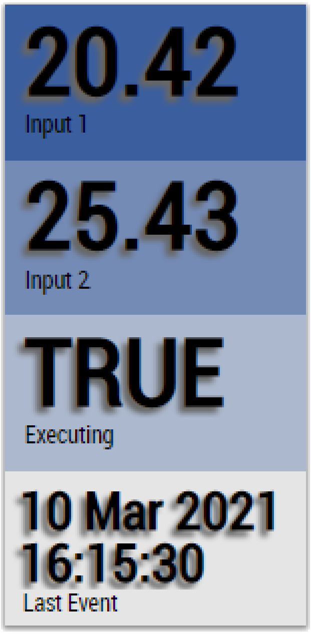
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

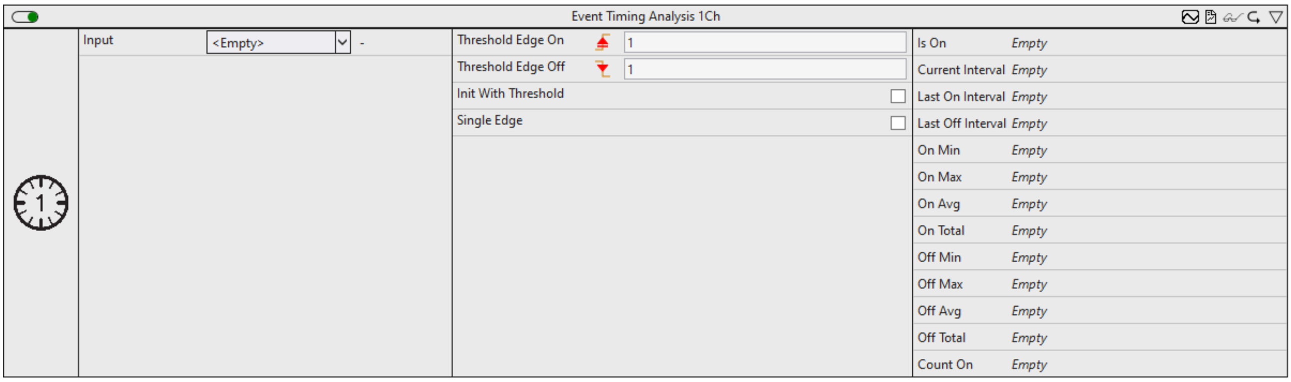
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, custom HMI controls can be mapped in the Edge Counter On Off 2Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.8 Event Timing Analysis 1Ch



Event Timing Analysis 1Ch

Input <Empty> -

Threshold Edge On	1
Threshold Edge Off	1
Init With Threshold	□
Single Edge	□

Is On	Empty
Current Interval	Empty
Last On Interval	Empty
Last Off Interval	Empty
On Min	Empty
On Max	Empty
On Avg	Empty
On Total	Empty
Off Min	Empty
Off Max	Empty
Off Avg	Empty
Off Total	Empty
Count On	Empty

The *Event Timing Analysis 1Ch* measures time differences between on- and off-event and counts the amount of raised events. An on-event is raised when the signal of the input channel passes the configured edge at a specific on-threshold and an off-event is raised when the off-threshold is passed by the same signal.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** indicates whether the edge counter should react to a rising or a falling edge.
- **Threshold On:** threshold of the signal at the respective edge. The On event is triggered when the signal passes this threshold.
- **Threshold Off:** threshold of the signal at the respective edge. The Off event is triggered when the signal passes this threshold.
- **Init With Threshold:** if the value is TRUE, the algorithm uses a threshold to initialize the internal state instead of waiting for an edge.
- **Single Edge:** if the value is TRUE, the algorithm is executed only on the basis of the edges of the On event.
- **Tolerance (optional):** tolerance value for the Equal / NotEqual comparisons.

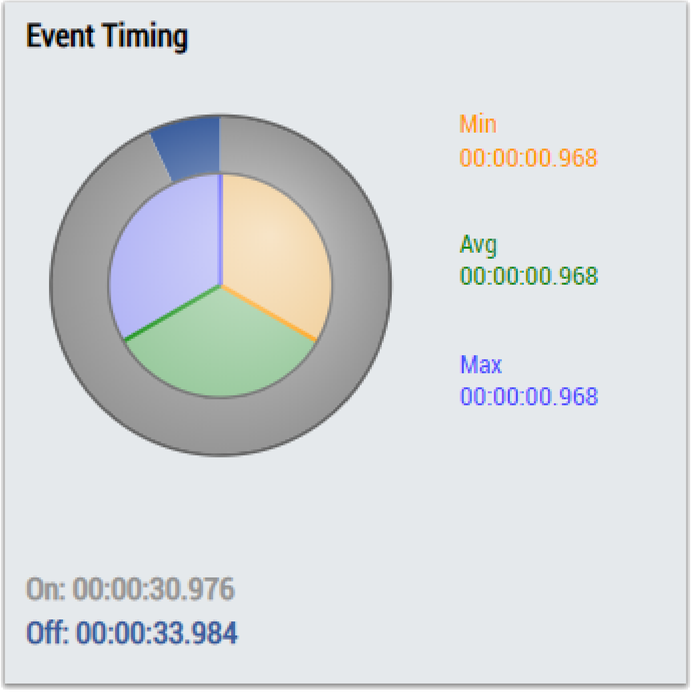
Output values

- **Is On:** Indicates *TRUE* within the timespan between the On event and the Off result, otherwise *FALSE*.
- **Current Interval:** Indicates the time of the current interval.
- **Last On Interval:** Timespan of the last completed interval between On event and Off event.
- **Last Off Interval:** Timespan of the last completed interval between Off event and the On event.
- **On Min:** Minimum time between On event and Off event.
- **On Max:** Maximum time between On event and Off event.
- **On Avg:** Average time between On event and Off event.
- **On Total:** Total time between On events and Off events.
- **Off Min:** Minimum time between Off event and On event.
- **Off Max:** Maximum time between Off event and On event.
- **Off Avg:** Average time between Off event and On event.
- **Off Total:** Total time between Off events and On events.
- **Count On:** Counts the number of triggered On events.

Standard HMI Controls

For the Event Timing Analysis 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The EventTiming control visualizes the output values Is On, Count On, Current Interval, On Min, On Max, On Avg, On Total, Off Min, Off Max, Off Avg and Off Total.



Event Timing

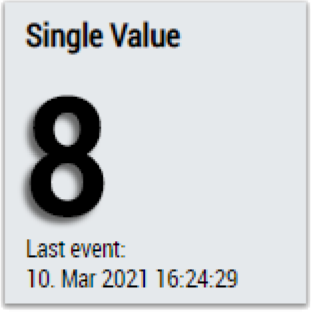
Min
00:00:00.968

Avg
00:00:00.968

Max
00:00:00.968

On: 00:00:30.976
Off: 00:00:33.984

2. The SingleValue control visualizes the output value Count On.

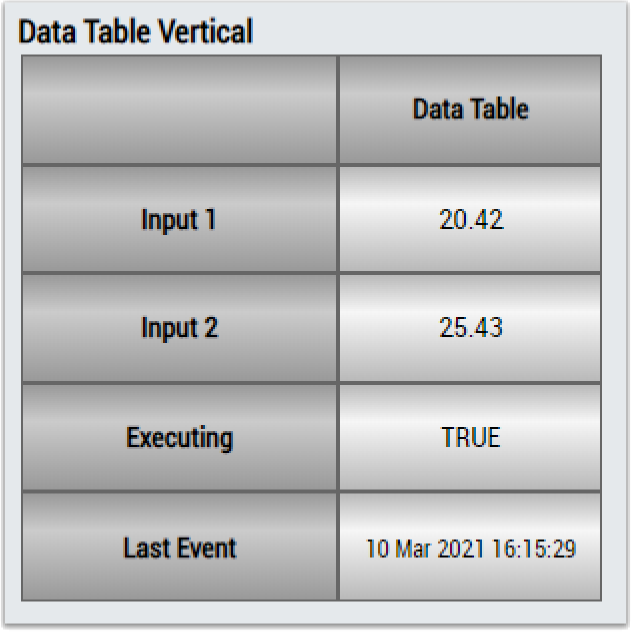


Single Value

8

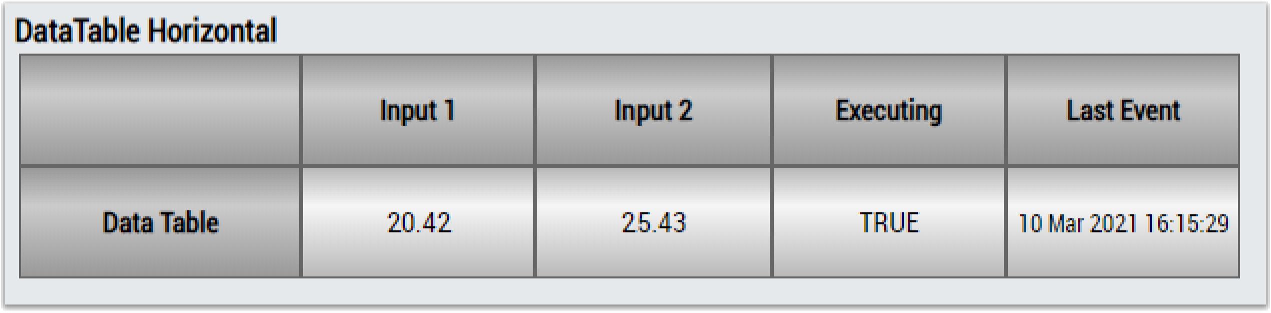
Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Is On, Count On, Current Interval, On Min, On Max, On Avg, On Total, Off Min, Off Max, Off Avg, Off Total.



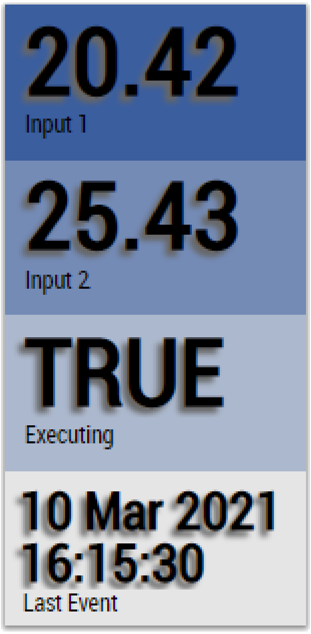
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

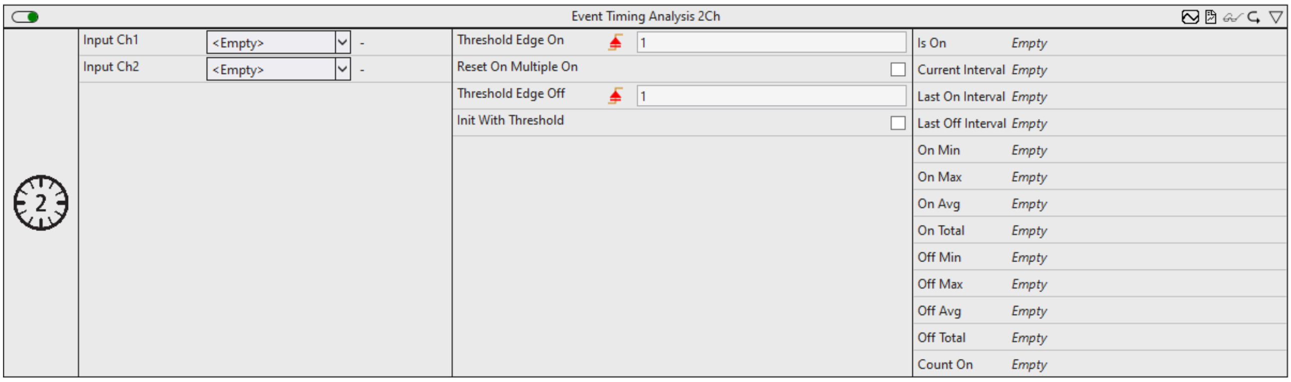
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Event Timing Analysis 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.9 Event Timing Analysis 2Ch



Event Timing Analysis 2Ch

Input Ch1	<Empty>	-
Input Ch2	<Empty>	-

Threshold Edge On	1
Reset On Multiple On	□
Threshold Edge Off	1
Init With Threshold	□

Is On	Empty
Current Interval	Empty
Last On Interval	Empty
Last Off Interval	Empty
On Min	Empty
On Max	Empty
On Avg	Empty
On Total	Empty
Off Min	Empty
Off Max	Empty
Off Avg	Empty
Off Total	Empty
Count On	Empty

The *Event Timing Analysis 2Ch* measures time differences between on- and off-event and counts the amount of raised events. An on-event is raised when the signal of the first input channel passes the configured edge at a specific on-threshold and an off-event is raised when the off-threshold is passed by the signal of the second channel.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold On:** Threshold of the signal at the respective edge. The On event is triggered when the signal passes this threshold.

- **Reset On Multiple On:** If the checkbox is checked, the "Count On" counter increments with every On event. Otherwise, the On events are only counted after a counter reset (Off event).
- **Threshold Off:** Threshold of the signal at the respective edge. The Off event is triggered when the signal passes this threshold.
- **Init With Threshold:** If the value is TRUE, the algorithm uses a threshold to initialize the internal state, instead of waiting for an edge.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

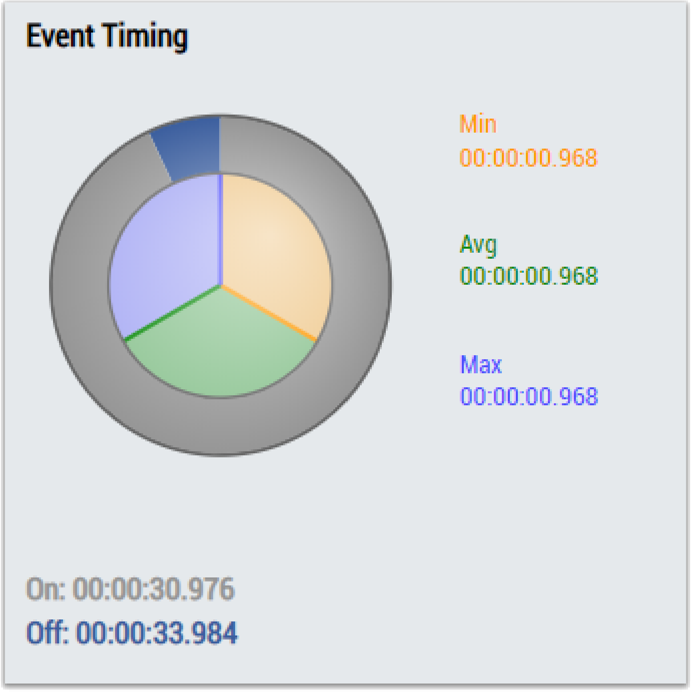
Output values

- **Is On:** Indicates *TRUE* within the timespan between the On event and the Off result, otherwise *FALSE*.
- **Current Interval:** Indicates the time of the current interval.
- **Last On Interval:** Timespan of the last completed interval between On event and Off event.
- **Last Off Interval:** Timespan of the last completed interval between Off event and the On event.
- **On Min:** Minimum time between On event and Off event.
- **On Max:** Maximum time between On event and Off event.
- **On Avg:** Average time between On event and Off event.
- **On Total:** Total time between On events and Off events.
- **Off Min:** Minimum time between Off event and On event.
- **Off Max:** Maximum time between Off event and On event.
- **Off Avg:** Average time between Off event and On event.
- **Off Total:** Total time between Off events and On events.
- **Count On:** Counts the number of triggered On events.

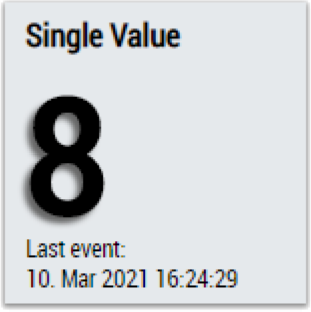
Standard HMI Controls

For the Event Timing Analysis 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The EventTiming control visualizes the output values Is On, Count On, Current Interval, On Min, On Max, On Avg, On Total, Off Min, Off Max, Off Avg and Off Total.



2. The SingleValue control visualizes the output value Count On.

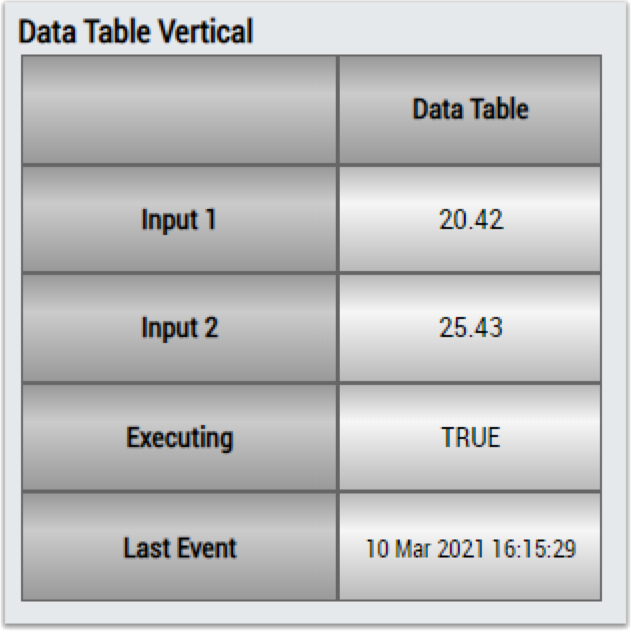


Single Value

8

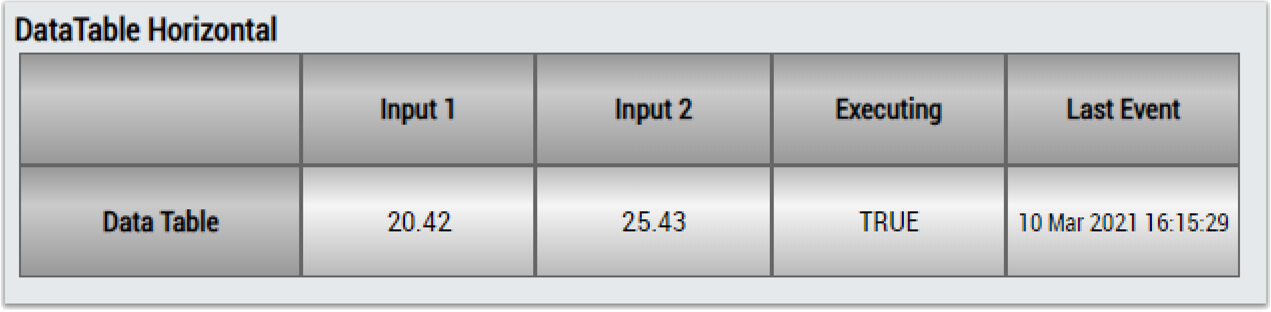
Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Is On, Count On, Current Interval, On Min, On Max, On Avg, On Total, Off Min, Off Max, Off Avg, Off Total.



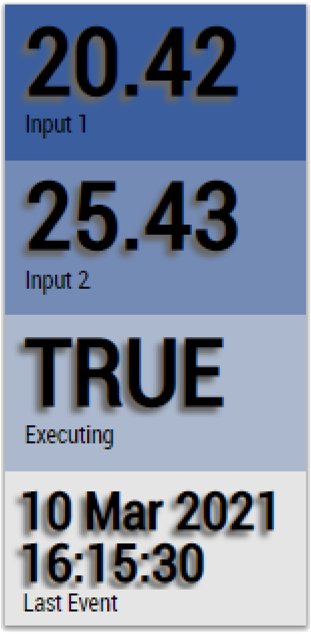
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

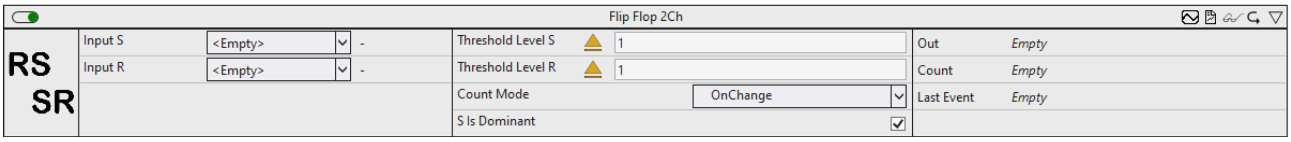
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Event Timing Analysis 2Ch algorithm using the Mapping Wizard [▸ 432].

6.4.1.10 Flip Flop 2Ch



Flip Flop 2Ch

RS SR

Input S	<Empty>	Threshold Level S	1	Out	*Empty*
Input R	<Empty>	Threshold Level R	1	Count	*Empty*
		Count Mode	OnChange	Last Event	*Empty*
		S Is Dominant	☑		

The *Flip Flop 2Ch* implements a bistable flip-flop. The dominance for setting (RS) or resetting (SR) the output value can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** specifies whether the input value should be greater than, greater than or equal to, less than or equal to, less than or not equal to the threshold.
- **Threshold Level S:** threshold of the signal for setting.
- **Threshold Level R:** threshold of the signal for resetting.
- **Count Mode:** mode of the result counter.
  OnChange: the counter counts every time the result changes to TRUE.
  Cyclic: the counter increments every cycle when the condition is *TRUE*.
- **S Is Dominant:** the dominance of setting (RS) or resetting (SR) can be configured.
- **Tolerance (optional):** tolerance value for the Equal / NotEqual comparisons.

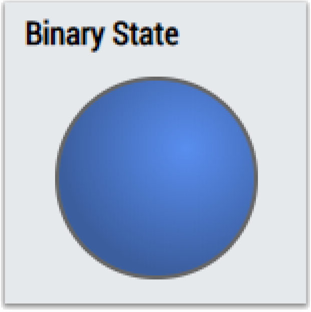
Output values

- **Out:** Result of the bistable flip-flop.
- **Count:** Incremented when the output value is TRUE. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

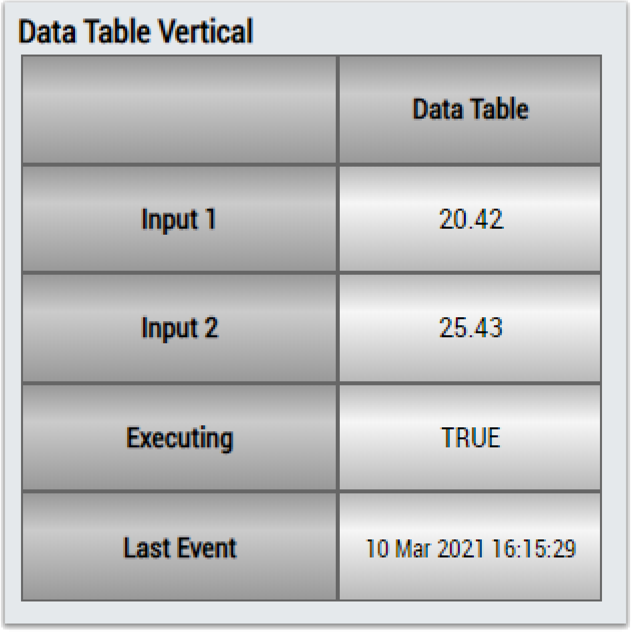
Standard HMI Controls

For the *Flip Flop 2Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

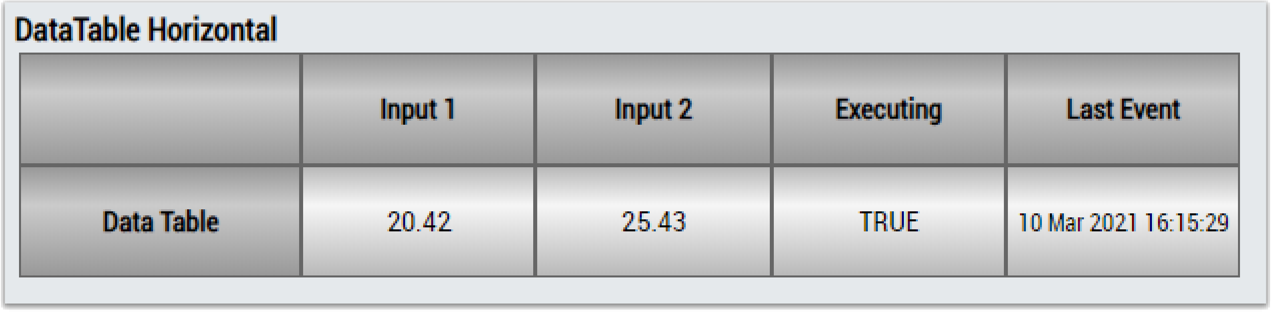
1. The BinaryState control visualizes the output value *Out*.



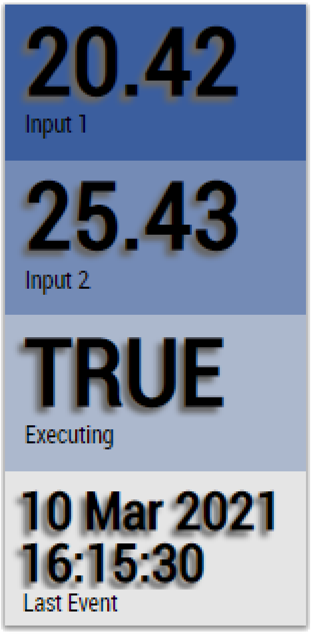
2. The Table Control and Multivalue control visualize all output values: *Out*, *Count*, *Last Event*.


Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29


DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

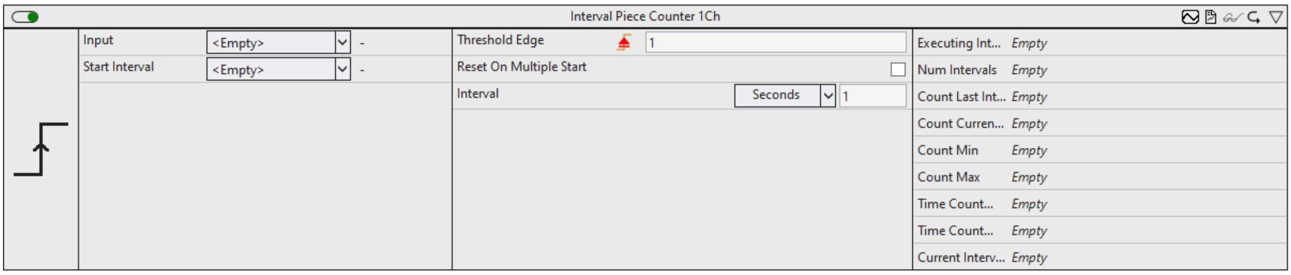
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Flip Flop 2Ch* algorithm using the Mapping Wizard [▸ 432].

6.4.1.11 Interval Piece Counter 1Ch



Interval Piece Counter 1Ch

Input <Empty> -
Start Interval <Empty> -

Threshold Edge 1
Reset On Multiple Start □
Interval Seconds 1

Executing Int... *Empty*
Num Intervals *Empty*
Count Last Int... *Empty*
Count Curren... *Empty*
Count Min *Empty*
Count Max *Empty*
Time Count... *Empty*
Time Count... *Empty*
Current Interv... *Empty*

The *Interval Piece Counter 1Ch* counts the amount of raised events within a configured interval, which starts when the value of the start interval flag is *TRUE.* An event is raised when the signal of the input channel passes the configured edge at a specific threshold. The calculation restarts when the time of the interval has elapsed and the value of the start interval flag is *True* again.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Reset On Multiple Start**: If the checkbox is checked, the interval restarts when the Start Interval flag becomes *TRUE* again. Otherwise, the interval restarts automatically when the time has elapsed.
- **Interval:** Time interval in which the values are to be calculated.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

Output values

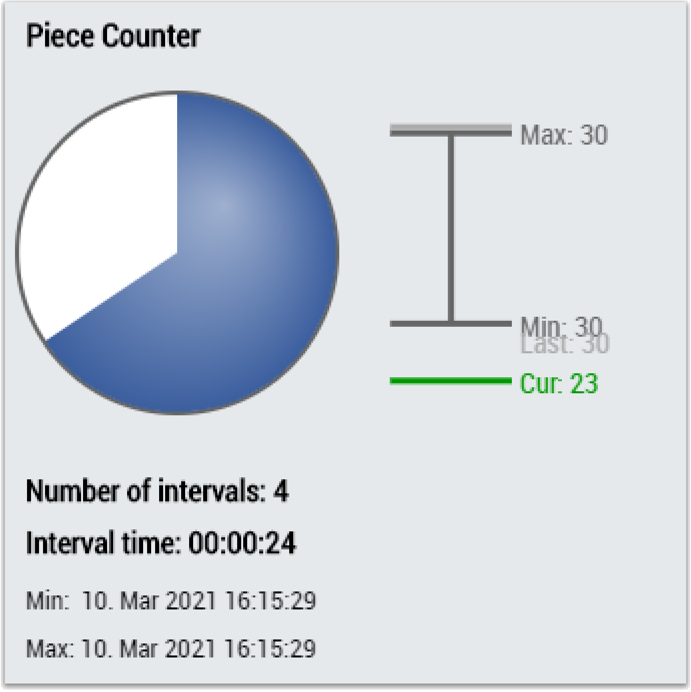
- **Executing Interval:** Indicates *True* if the calculation is active and the interval is running, otherwise *False*.
- **Num Intervals:** Indicates the number of intervals.
- **Count Last Interval:** Indicates the number of triggered events in the last interval.

- **Count Current Interval:** Indicates the number of triggered events in the current interval or, if the calculation is currently inactive, the number of triggered events in the last interval.
- **Count Min:** Indicates the minimum of triggered events in an interval.
- **Count Max:** Indicates the maximum of triggered events in an interval.
- **Time Count Min:** Indicates the time value of the minimum → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Time Count Max:** Indicates the time value of the maximum → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Current Interval Time:** Indicates the time of the current interval → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Interval Piece Counter 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The PieceCounter control visualizes the output values Num Intervals, Count Last Interval, Count Current Interval, Count Min, Count Max, Time Count Min, Time Count Max and Time Current Interval.



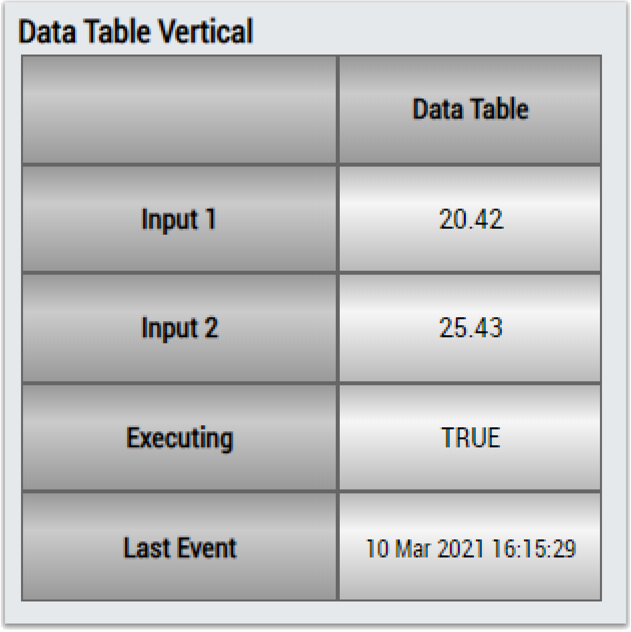
Number of intervals: 4

Interval time: 00:00:24

Min: 10. Mar 2021 16:15:29

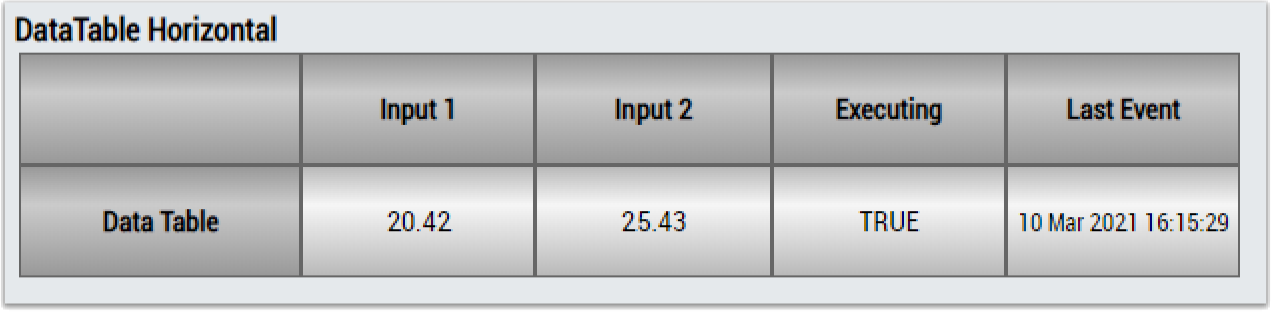
Max: 10. Mar 2021 16:15:29

2. The Table Control or Multivalue Control visualizes all output values: Num Intervals, Count Last Interval, Count Current Interval, Count Min, Count Max, Time Count Min, Time Count Max, Time Current Interval.



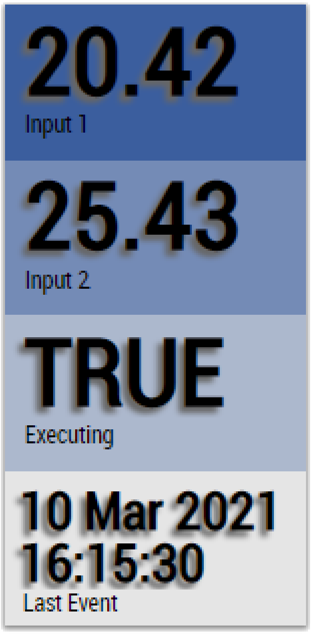
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

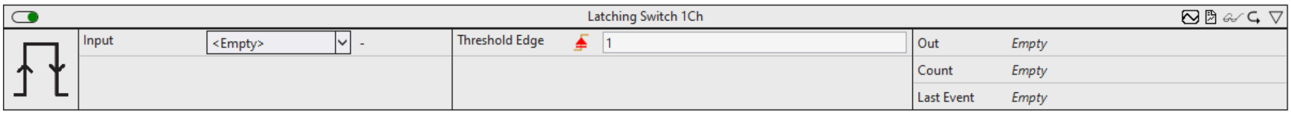
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Interval Piece Counter 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.12 Latching Switch 1Ch



Latching Switch 1Ch

Input	<Empty>	-	Threshold Edge	1	Out Empty
					Count Empty
					Last Event Empty

The *Latching Switch 1Ch* realizes a virtual impulse switch. The output alternates between TRUE and FALSE on each edge detected at the input.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether to react to a rising or falling edge at the input
- **Threshold Edge:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

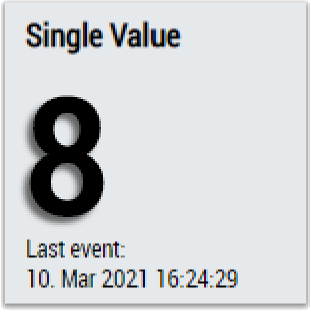
Output values

- **Out:** Result of the virtual impulse switch.
- **Count:** incremented when the output value changes.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the *Latching Switch 1Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue control visualizes the output values Count and Last Event.

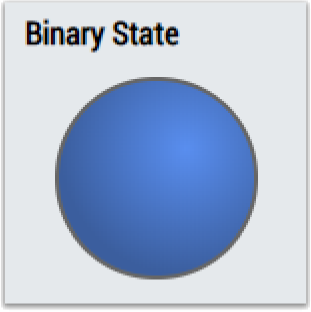


Single Value

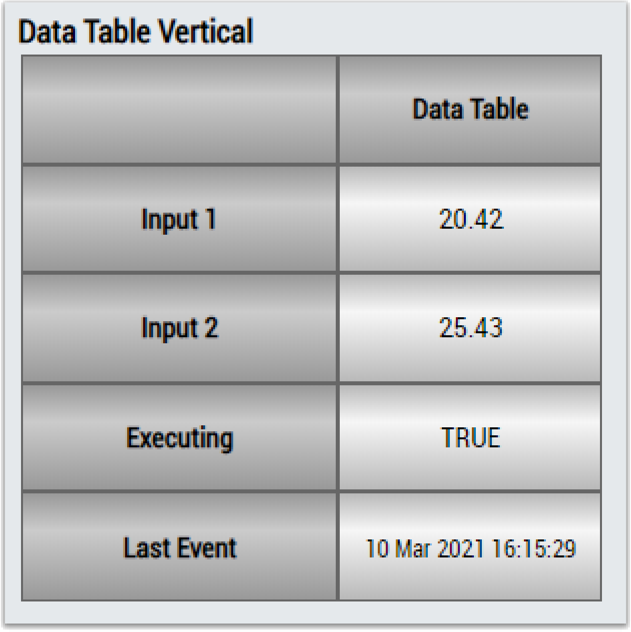
8

Last event:
10. Mar 2021 16:24:29

2. The BinaryState control visualizes the output value *Out*.

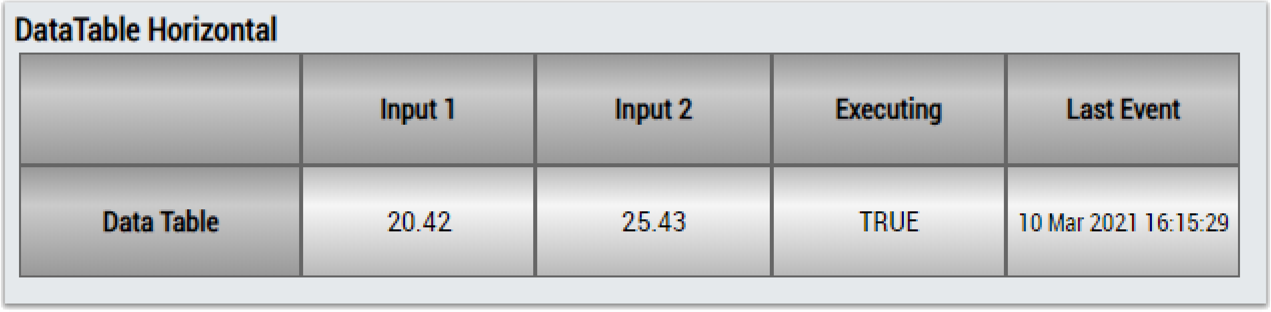


3. The Table Control and Multivalue Control visualize all output values.



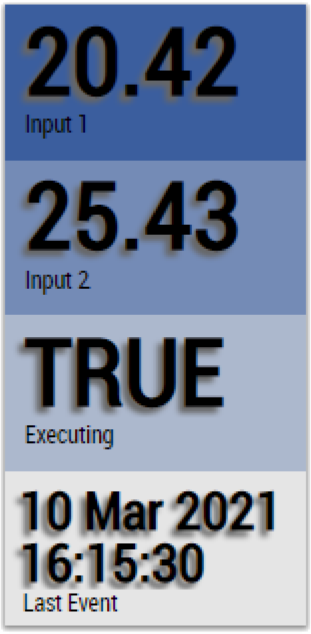
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

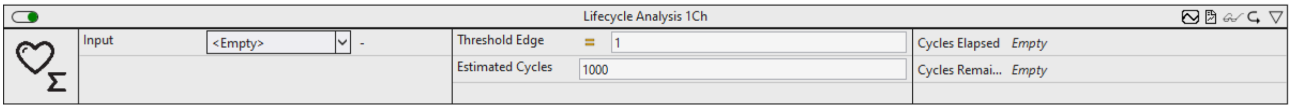
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped using the Mapping Wizard [▸ 432].

6.4.1.13 Lifecycle Analysis 1Ch



Lifecycle Analysis 1Ch

Input <Empty> -

Threshold Edge = 1

Estimated Cycles 1000

Cycles Elapsed *Empty*

Cycles Remai... *Empty*

The *Lifecycle Analysis 1Ch* calculates the elapsed and the estimated remaining cycles of a device. When the signal of the input channel passes the configured edge at a specific threshold, the elapsed cycles are increased and the remaining cycles are decreased.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. An event is triggered when the signal passes this threshold.
- **Estimated Cycles:** Estimated cycles over the lifetime of the respective device.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

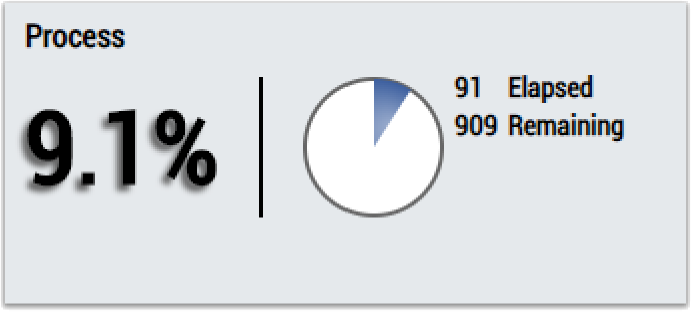
Output Values

- **Elapsed Cycles:** Counts the amount of cycles which are already elapsed.
- **Remaining Cycles:** Shows the remaining cycles of the device as the difference of estimated and elapsed cycles.

Standard HMI Controls

For the Lifecycle Analysis 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Process control visualizes the output values Elapsed Cycles and Remaining Cycles.

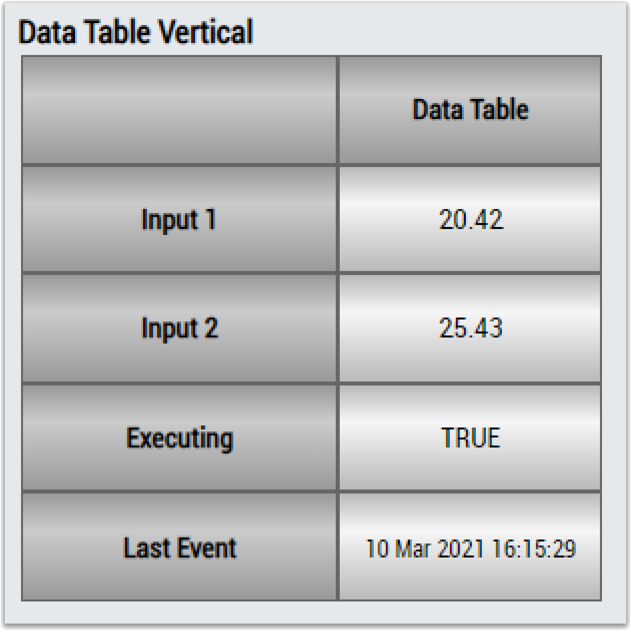


Process

9.1%

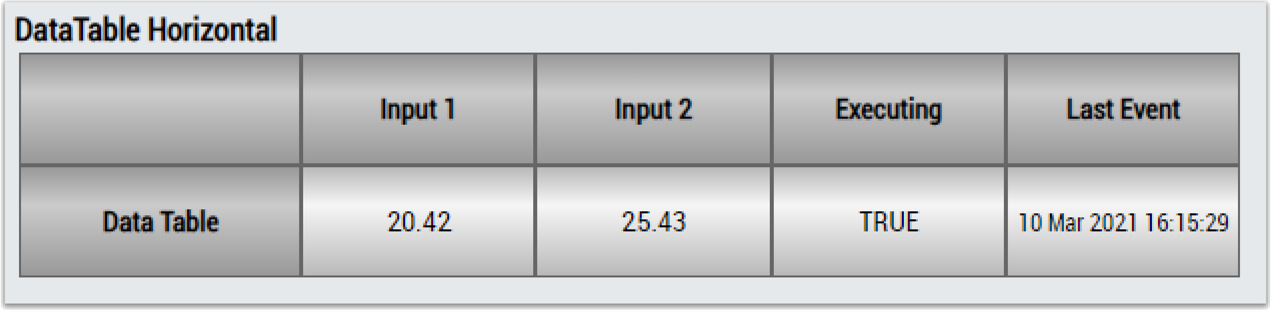
91 Elapsed
909 Remaining

2. The Table Control or Multivalue Control visualizes all output values: Elapsed Cycles, Remaining Cycles.



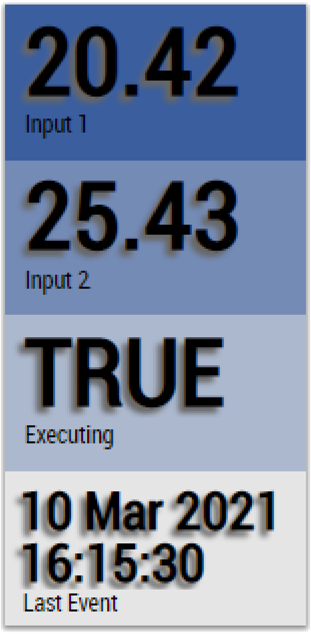
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

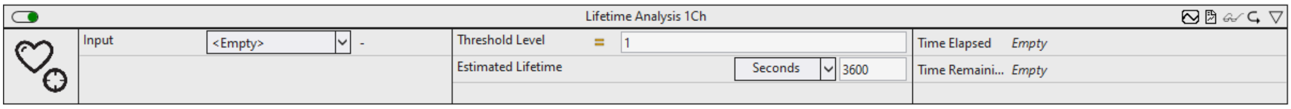
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Lifecycle Analysis 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.14 Lifetime Analysis 1Ch



Lifetime Analysis 1Ch

Input <Empty> -

Threshold Level = 1

Estimated Lifetime Seconds 3600

Time Elapsed *Empty*

Time Remaini... *Empty*

The *Lifetime Analysis 1Ch* calculates the elapsed and the estimated remaining lifetime of a device. If the input value met the configured condition the lifetime will be reduced.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** Specifies whether the input value should be greater than, greater than or equal to, less than or equal to, less than or not equal to the threshold.
- **Threshold:** Signal threshold.
- **Estimated Lifetime:** Estimated lifetime of the respective device.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

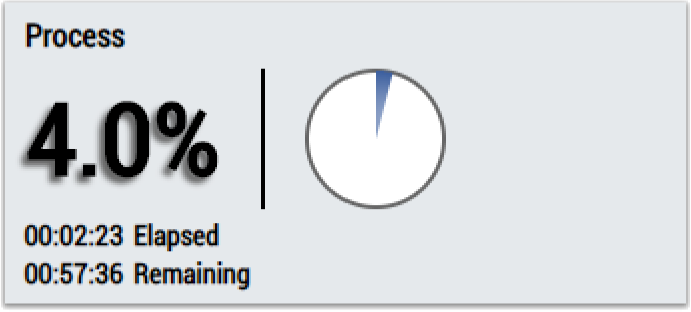
Output Values

- **Elapsed Lifetime:** Shows the lifetime which is already elapsed.
- **Remaining Lifetime:** Shows the remaining lifetime of the device as the difference of estimated and elapsed lifetime.

Standard HMI Controls

For the Lifetime Analysis 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Process control visualizes the output values Elapsed Lifetime and Remaining Lifetime.

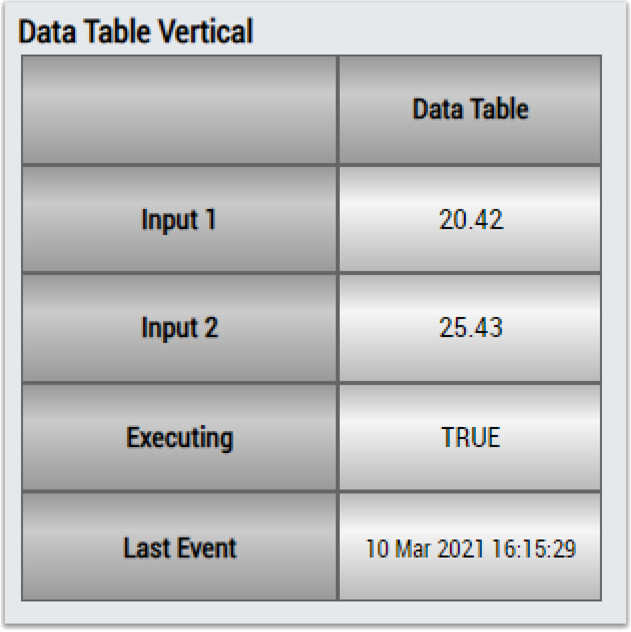


Process

4.0%

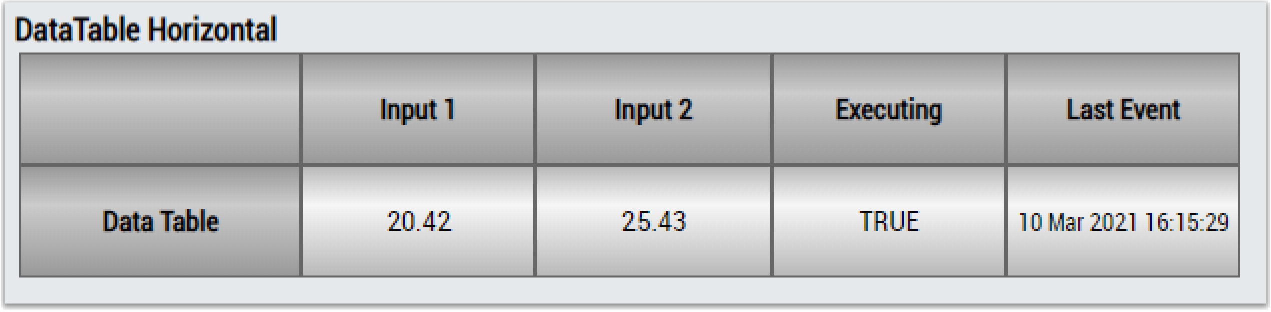
00:02:23 Elapsed
00:57:36 Remaining

2. The Table Control or Multivalue Control visualizes all output values: Elapsed Lifetime, Remaining Lifetime.



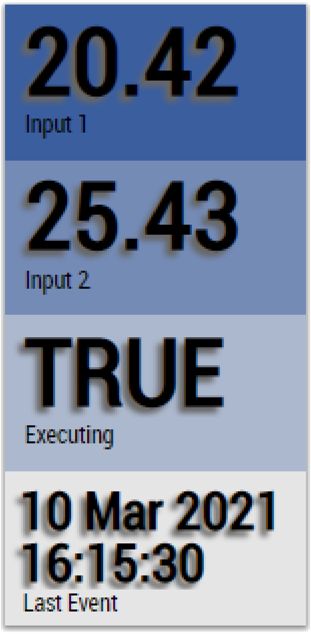
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

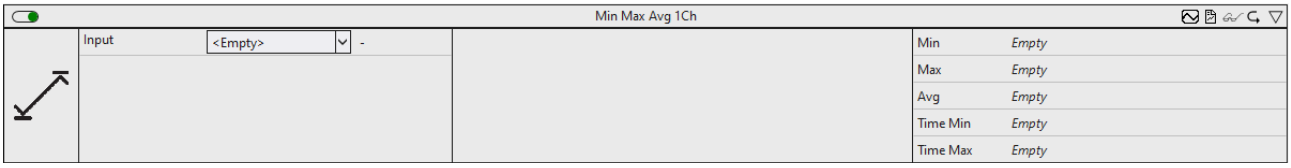
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Lifetime Analysis 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.15 Min Max Avg 1Ch



Min Max Avg 1Ch

Input <Empty> -

Min	Empty
Max	Empty
Avg	Empty
Time Min	Empty
Time Max	Empty

The *Min Max Avg 1Ch* calculates the minimum, maximum and the average of the input values from the beginning of the analysis up to the current moment. Furthermore, the time values of minimum and maximum are shown.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

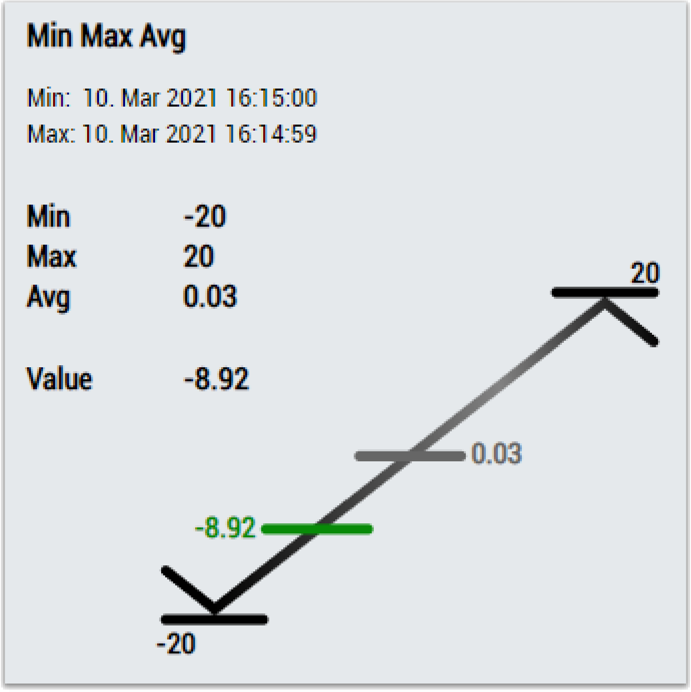
Output values

- **Min:** Indicates the minimum of the input values.
- **Max:** Indicates the maximum of the input values.
- **Avg:** Indicates the average of the input values.
- **Time Min:** Indicates the time value of the minimum → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Time Max:** Indicates the time value of the maximum → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

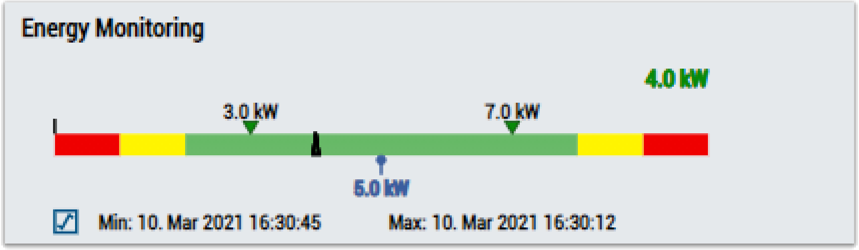
Standard HMI Controls

For the Min Max Avg 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

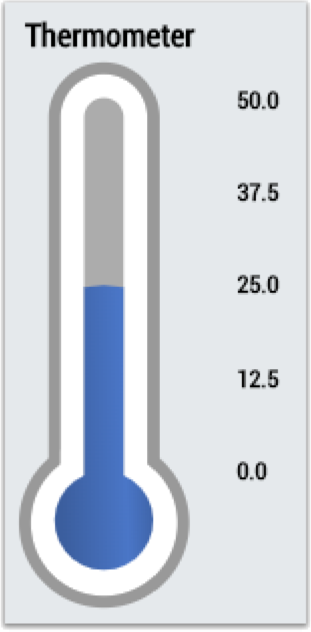
1. The MinMaxAvg control visualizes the output values Min, Max, Avg, Time Min and Time Max as well as the input value of the data.



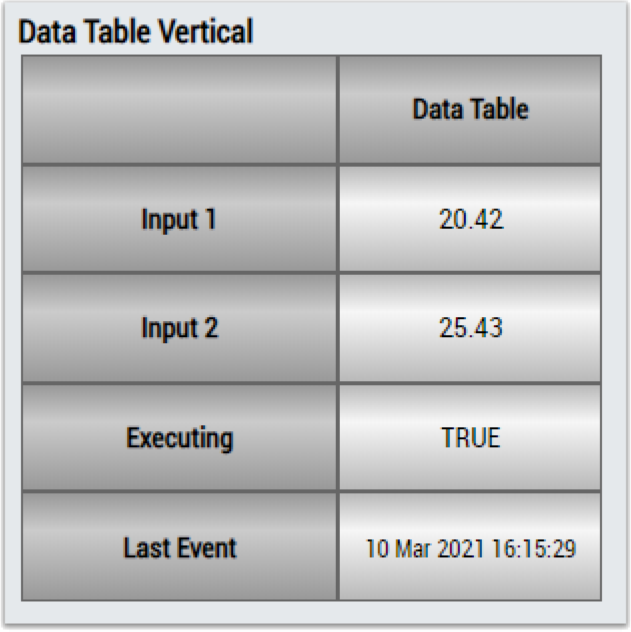
2. The EnergyMonitoring control visualizes all output values: Min, Max, Avg, Time Min, Time Max and the current input value.



3. The Thermometer control visualizes the average (Avg) temperature.

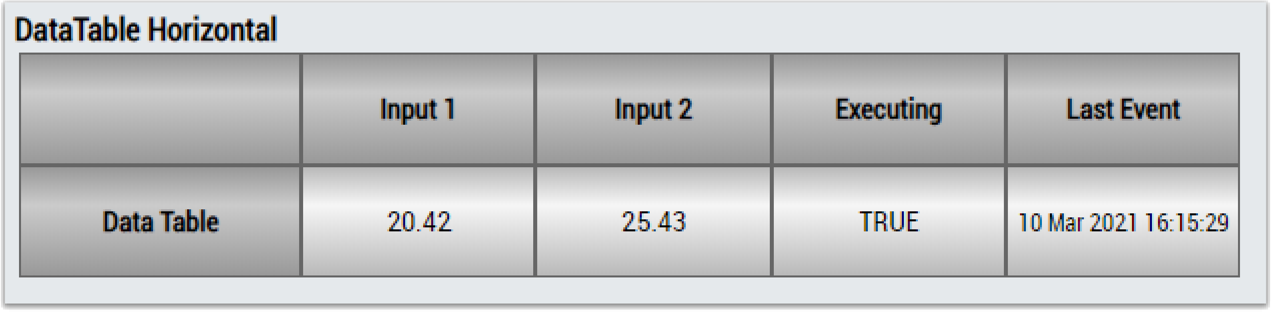


4. The Table Control or Multivalue Control visualizes all output values: Min, Max, Avg, Time Min, Time Max.



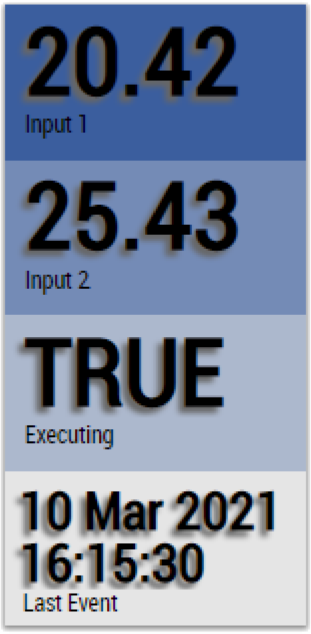
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

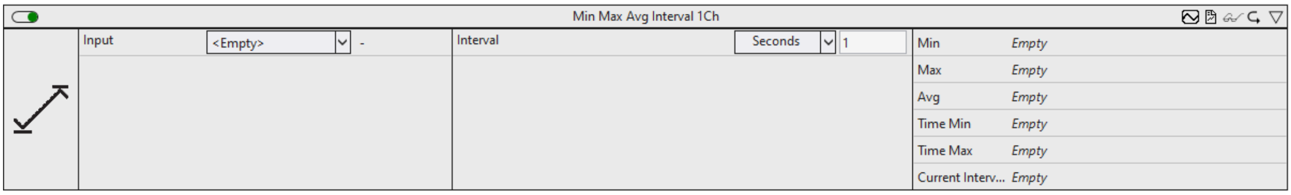
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Min Max Avg 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.16 Min Max Avg Interval 1Ch



The *Min Max Avg Interval 1Ch* calculates the minimum, maximum and the average of the input values for the time period of the configured Interval. Furthermore the time values of minimum and maximum are shown. Note that all values are from the relative last interval and that they will only be updated when the interval is over. The calculation restarts when the time of the interval has elapsed.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

If only individual samples within the configured time signal are to be included in the calculation of the output values, this can optionally be implemented via the Boolean input *Add Sample*. All input values are considered for which the input *Add Sample* has the value *TRUE*.

Configuration Options

- **Interval:** Time Interval in which the values should be calculated.

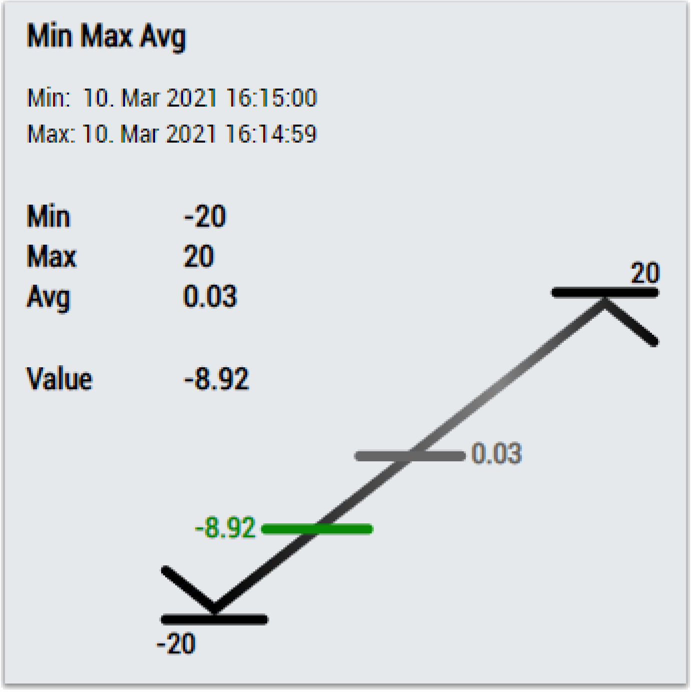
Output values

- **Min:** Indicates the minimum of the input values in the last time interval.
- **Max:** Indicates the maximum of the input values in the last time interval.
- **Avg:** Indicates the average of the input values in the last time interval.
- **Time Min:** Indicates the time value of the minimum in the last time interval→ the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Time Max:** Indicates the time value of the maximum in the last time interval → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Current Interval Time:** indicates the amount of time that has already elapsed from the current interval.

Standard HMI Controls

For the Min Max Avg Interval 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The MinMaxAvg control visualizes the output values Min, Max, Avg, Time Min and Time Max as well as the input value of the data.

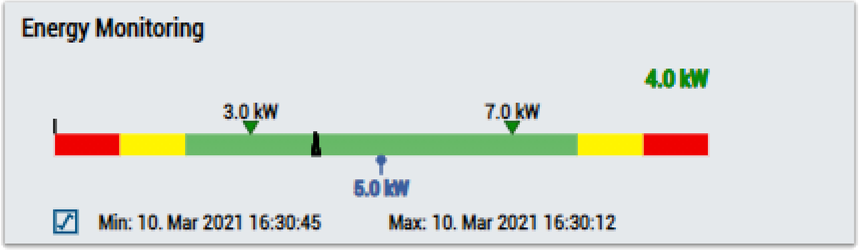


Min Max Avg

Min: 10. Mar 2021 16:15:00
Max: 10. Mar 2021 16:14:59

Min	-20
Max	20
Avg	0.03
Value	-8.92

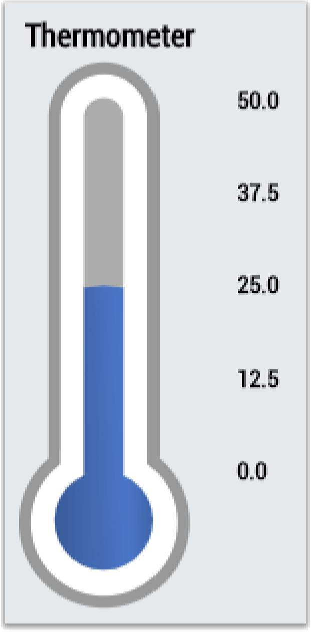
2. The EnergyMonitoring control visualizes the output values: Min, Max, Avg, Time Min, Time Max and the current input value.



Energy Monitoring

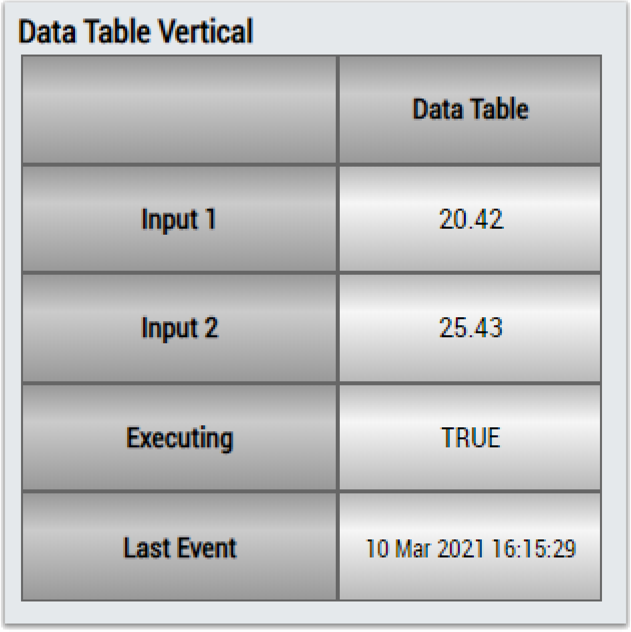
Min: 10. Mar 2021 16:30:45 Max: 10. Mar 2021 16:30:12

3. The Thermometer control visualizes the average (Avg) temperature.



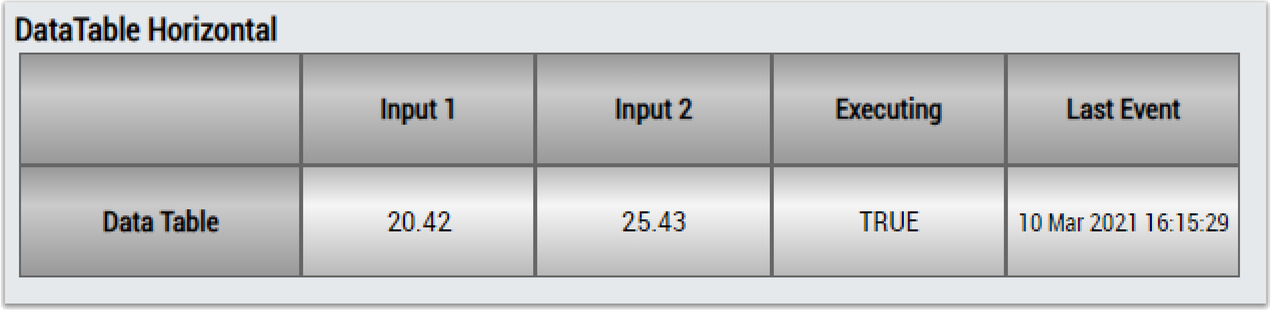
Thermometer

4. The Table Control or Multivalue Control visualizes all output values: Min, Max, Avg, Time Min, Time Max, Current Interval Time.



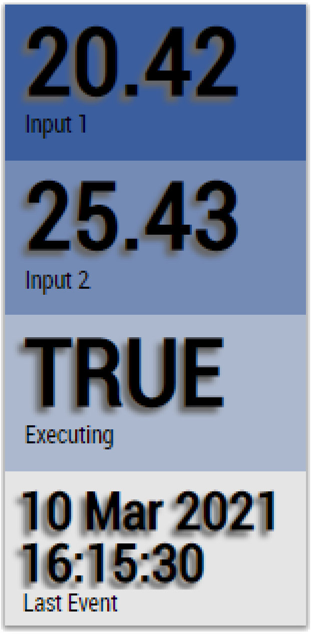
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

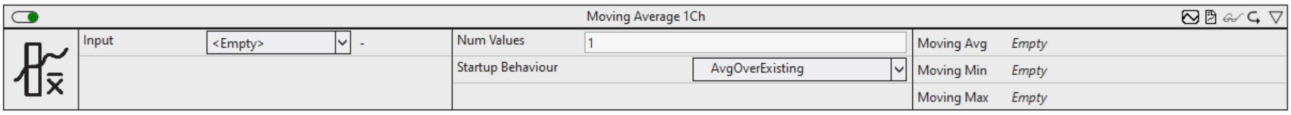
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Min Max Avg Interval 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.17 Moving Average 1Ch



Moving Average 1Ch

Input <Empty> -

Num Values 1

Startup Behaviour AvgOverExisting

Moving Avg *Empty*

Moving Min *Empty*

Moving Max *Empty*

The *Moving Average 1Ch* calculates the moving average, the minimum and the maximum of the most recent input values in an interval of specified length. Furthermore the time values of minimum and maximum are shown. The calculation of the moving average depends on the configuration parameters *Num Values* and *Startup Behaviour*.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Num Values:** Amount of values which will be included in the calculation of the moving average, the minimum and the maximum.
- **Startup Behaviour:** Calculation behaviour at the beginning of the analysis before at least *Num Values* input values exist.

  ZeroPadding: The missing values are filled with zeros.

  UseFirstValue: The first value is used until the amount of values is equivalent to *Num Values*.

  WaitUntilFilled: The first result is calculated when the amount of values is equivalent to *Num Values*.

  AvgOverExisting: The average will be calculated with the already existing values until the amount of values is equivalent to *Num Values*.

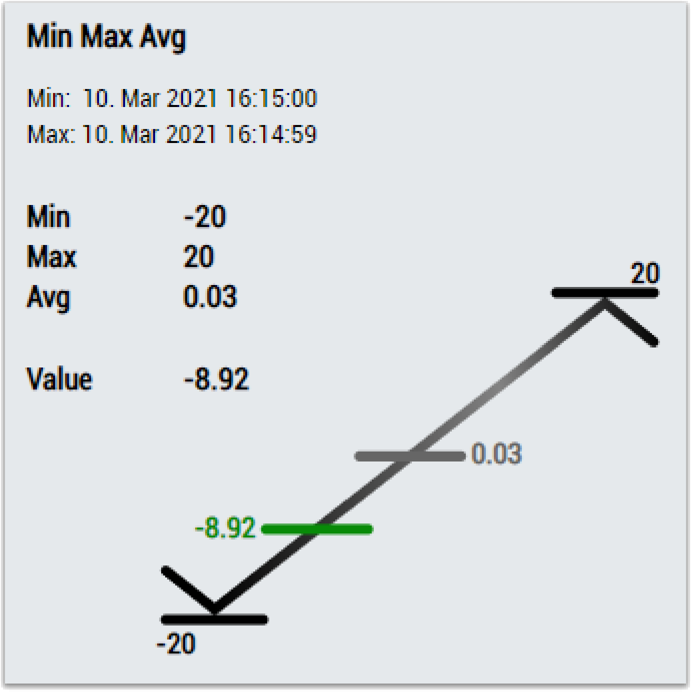
Output Values

- **Moving Avg:** Shows the current average value.
- **Moving Min:** Shows the minimum of the last n input values.
- **Moving Max:** Shows the maximum of the last n input values.

Standard HMI Controls

For the Moving Average 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The MinMaxAvg control visualizes the output values Moving Min, Moving Max and Moving Avg as well as the input value of the data.

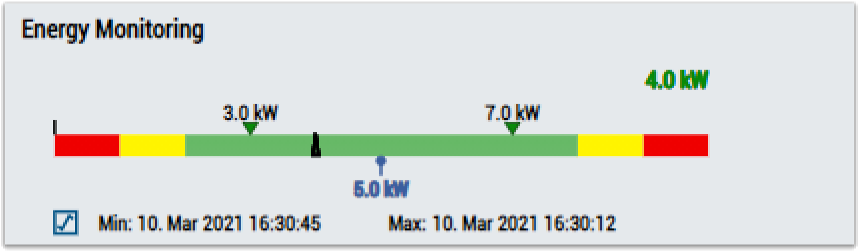


Min Max Avg

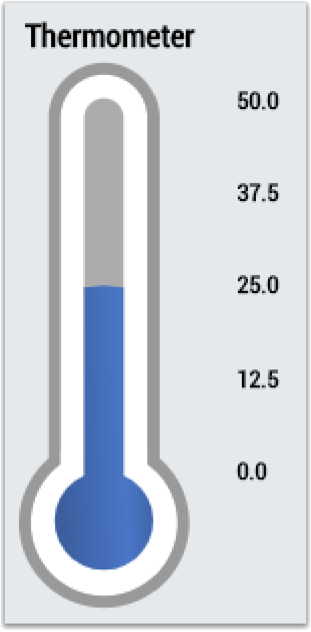
Min: 10. Mar 2021 16:15:00
Max: 10. Mar 2021 16:14:59

Min	-20
Max	20
Avg	0.03
Value	-8.92

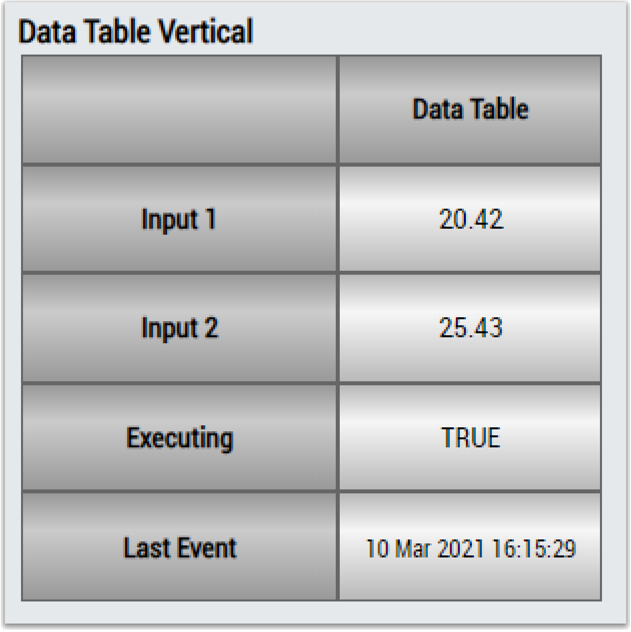
2. The EnergyMonitoring control visualizes the output values Moving Min, Moving Max and Moving Avg, and the input value of the data.



3. The Thermometer control visualizes the average (Moving Avg) temperature.

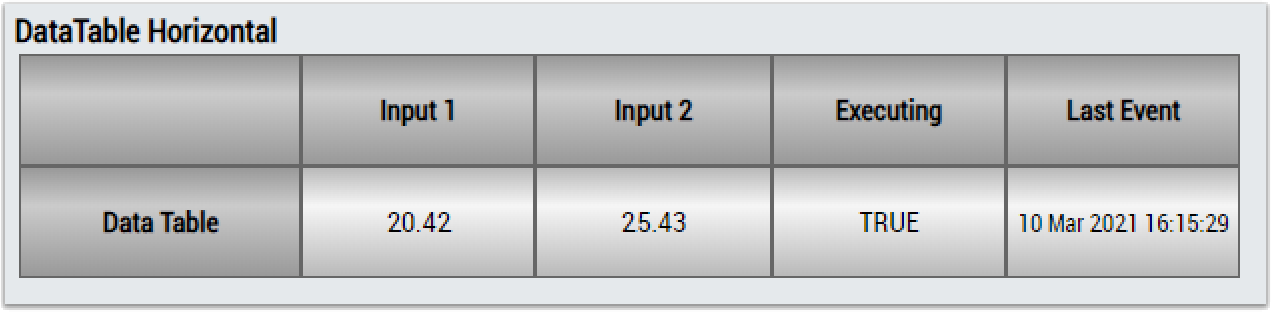


4. The Table Control or Multivalue Control visualizes all output values: Moving Min, Moving Max, Moving Avg.



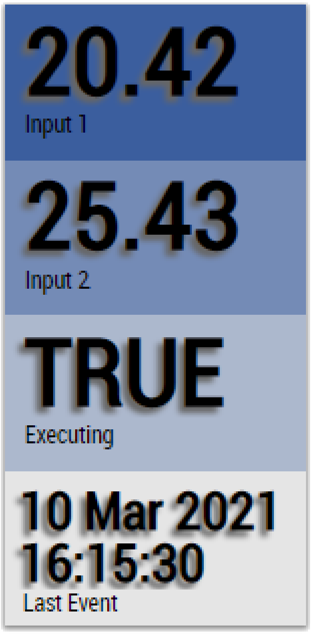
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

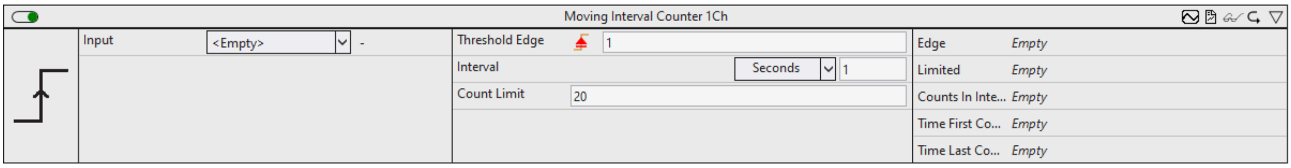
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Moving Average 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.18 Moving Interval Counter 1Ch



Moving Interval Counter 1Ch

Input	<Empty>	-	
Threshold Edge		1	
Interval	Seconds	1	
Count Limit	20		

Edge	Empty
Limited	Empty
Counts In Inte...	Empty
Time First Co...	Empty
Time Last Co...	Empty

The *Moving Interval Counter 1Ch* counts the amount of raised events within a configured interval. An event is raised when the signal of the input channel passes the configured edge at a specific threshold. The calculation restarts when the time of the interval has elapsed.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Interval:** Time interval in which the values are to be calculated.
- **Count Limit:** Limits the number of edges that can be counted in an interval.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

Output values

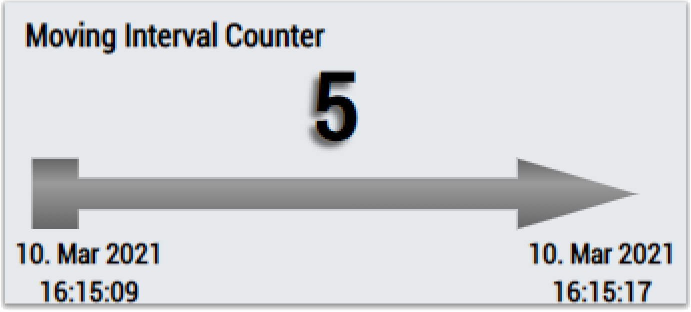
- **Edge:** Indicates *TRUE* at the time the event is triggered, otherwise *FALSE*.

- **Limited:** Indicates TRUE if the number of edges in the current interval exceeds the set Count Limit.
- **Counts in Interval:** Indicates the number of triggered events in the current interval.
- **Time First Count:** Indicates the time of the first triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Time Last Count:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Moving Interval Counter 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The MovingIntervalCounter control visualizes the output values Counts in Interval, Time First Count and Time Last Count.



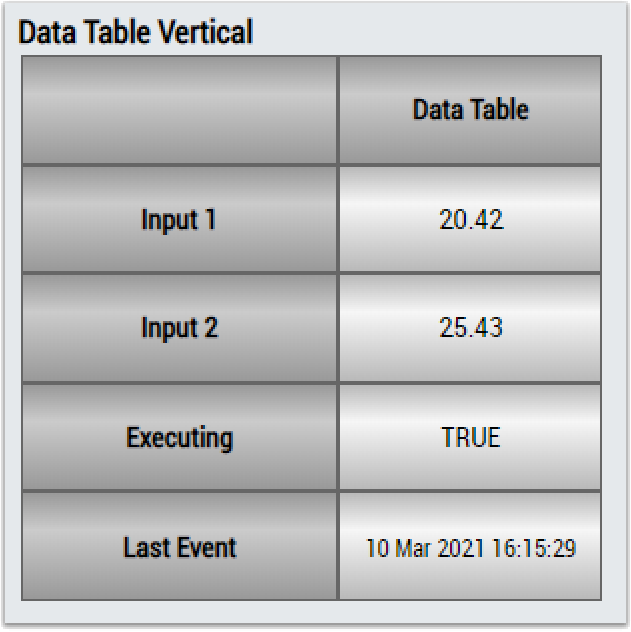
Moving Interval Counter

5

10. Mar 2021
16:15:09

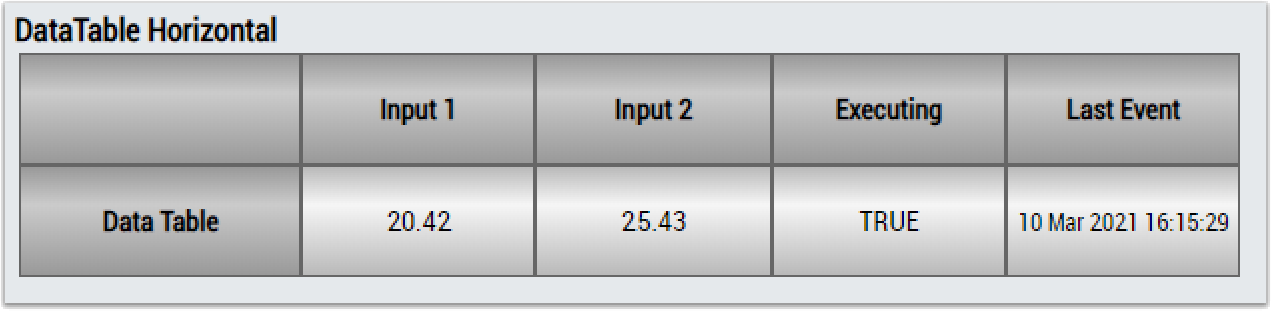
10. Mar 2021
16:15:17

2. The Table Control or Multivalue Control visualizes all output values: Edge, Counts in Interval, Time First Count, Time Last Count, Limited.



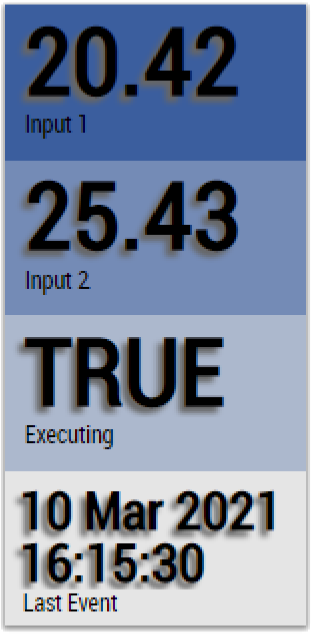
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

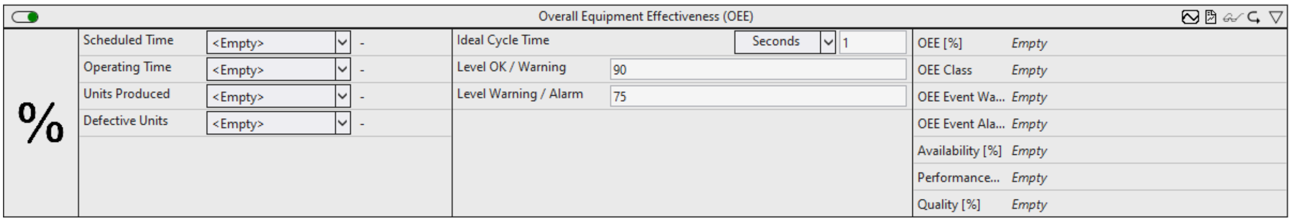
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Moving Interval Counter 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.19 Overall Equipment Effectiveness (OEE)



Overall Equipment Effectiveness (OEE)

Scheduled Time	<Empty>	-	Ideal Cycle Time	Seconds 1	OEE [%]	Empty
Operating Time	<Empty>	-	Level OK / Warning	90	OEE Class	Empty
Units Produced	<Empty>	-	Level Warning / Alarm	75	OEE Event Wa...	Empty
Defective Units	<Empty>	-			OEE Event Ala...	Empty
					Availability [%]	Empty
					Performance...	Empty
					Quality [%]	Empty

The *Overall Equipment Effectiveness (OEE)* calculates key figures that make it possible to compare the current state of the manufacturing process with its maximum potential.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Ideal Cycle Time:** ideal cycle time for the production of one unit.
- **Level Ok / Warning:** the overall equipment effectiveness greater than the configured threshold is classified as *OK*. If the overall equipment effectiveness is less than or equal to the configured threshold, it is classified as *Warning*.
- **Level Warning / Alarm:** the overall equipment effectiveness greater than the configured threshold is classified as *Warning*. If the overall equipment effectiveness is less than or equal to the configured threshold, it is classified as *Alarm* .

Input values

- **Scheduled Time:** the operating time is calculated from the calendar time minus the scheduled non-production.
- **Operating Time:** the running time is calculated from the operating time minus the downtimes.
- **Units Produced:** corresponds to the number of units produced including defective units.
- **Defective Units:** corresponds to the number of defective units.

Output values

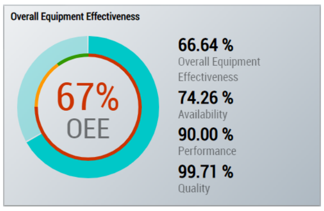
- **OEE:** Overall equipment effectiveness in percent. It is calculated by multiplying the availability factor, the performance factor and the quality factor.

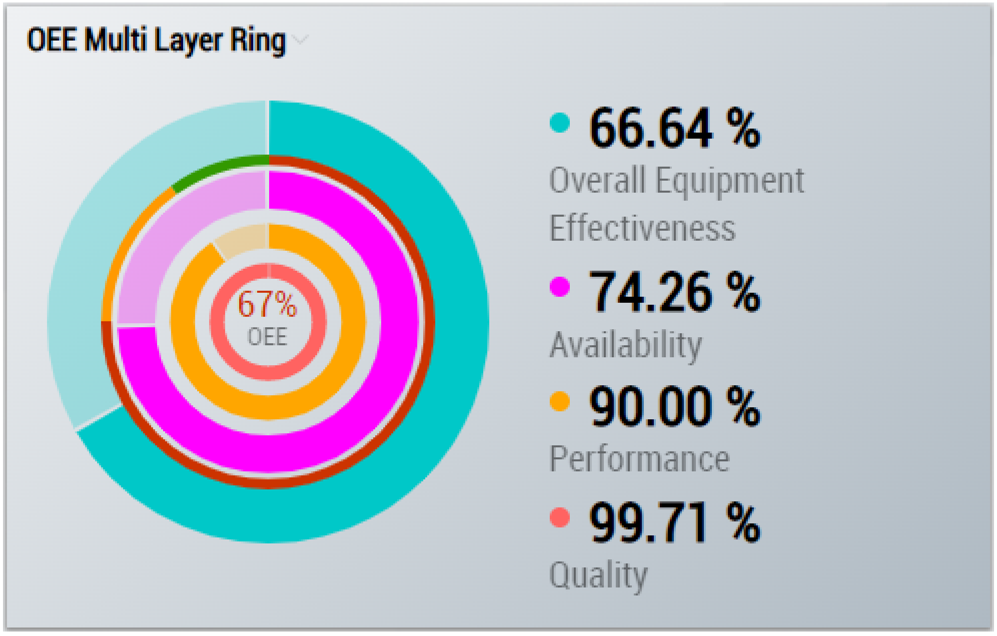
- **OEE Class:** Classification of overall equipment effectiveness.
- **OEE Event Warning:** Indicates the time of the last triggered warning event → the event can be dragged and dropped into the scope chart to display it as a trigger-event.
- **OEE Event Alarm:** Indicates the time of the last triggered alarm event → the event can be dragged and dropped into the scope chart to display it as a trigger-event.
- **Availability:** Availability factor in percent. It is calculated from the ratio between the runtime and the operating time.
- **Performance:** Performance factor in percent. It is calculated from the ratio of units actually produced and the number of units produced in the ideal case.
- **Quality:** Quality factor in percent. It is calculated as the ratio of intact produced units to produced units.

Standard HMI Controls

For the *Overall Equipment Effectiveness (OEE)* algorithm the following HMI controls are available for generating an Analytics Dashboard:

1. The OEE Control comes with three designs. It visualizes all output values as well as the parameters *Level OK / Warning* and *Level Warning / Alarm*.





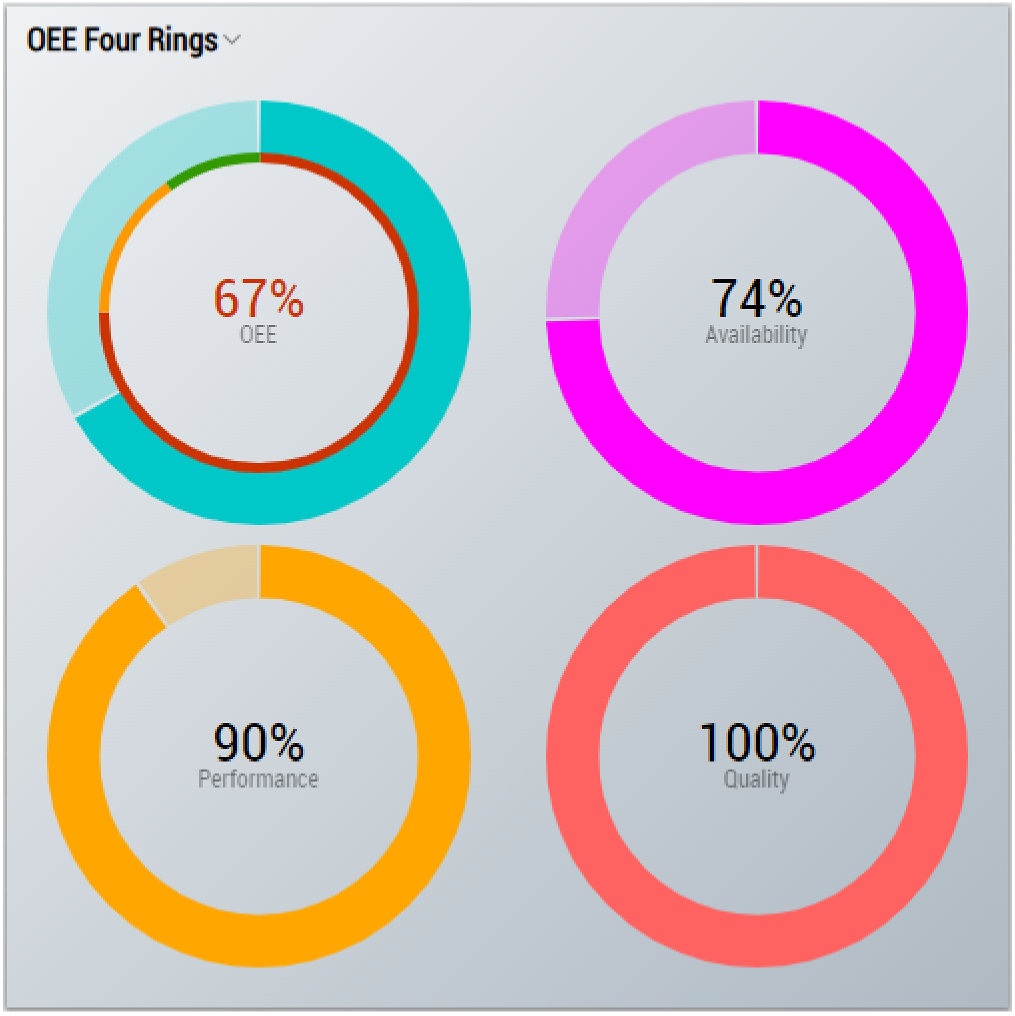
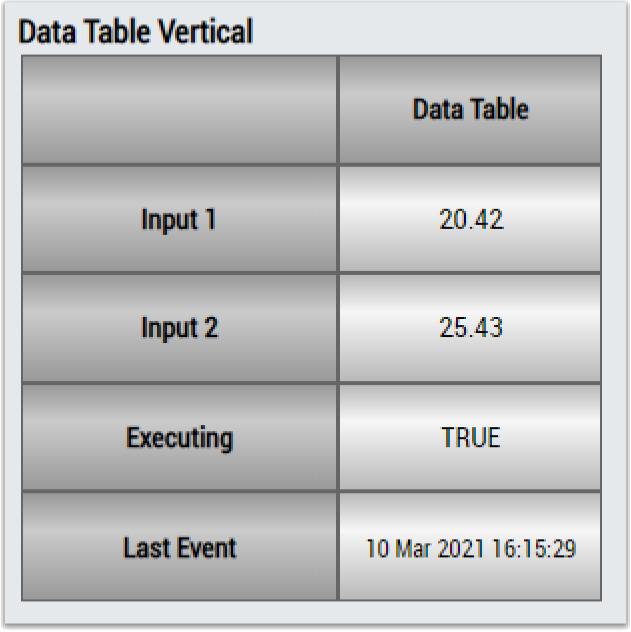
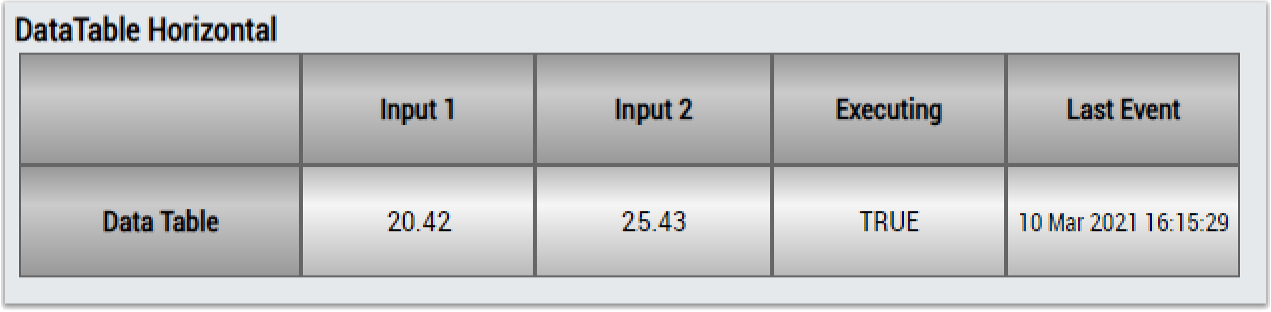


Fig. 1:

2. The Table Control and Multivalue Control visualize the output values: *OEE, Availability, Performance and Quality*.

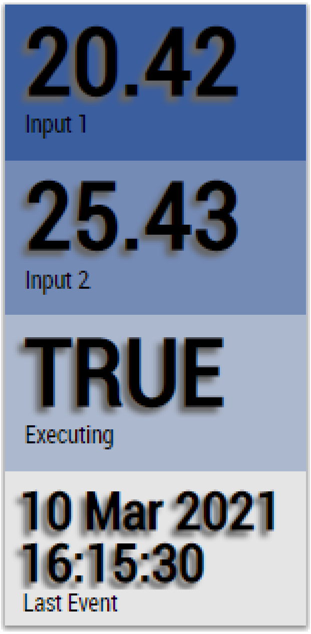

Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

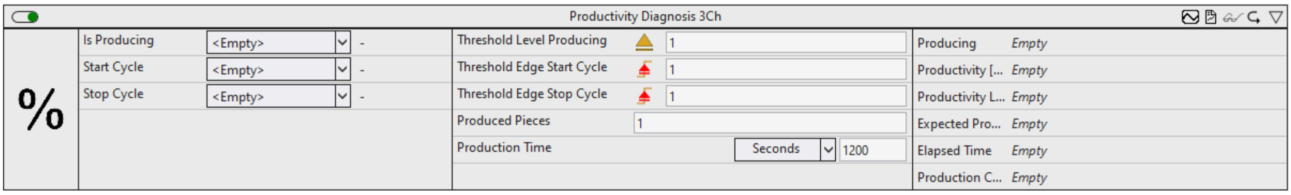
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Overall Equipment Effectiveness (OEE)* algorithm using the Mapping Wizard [▶ 432].

6.4.1.20 Productivity Diagnosis 3Ch



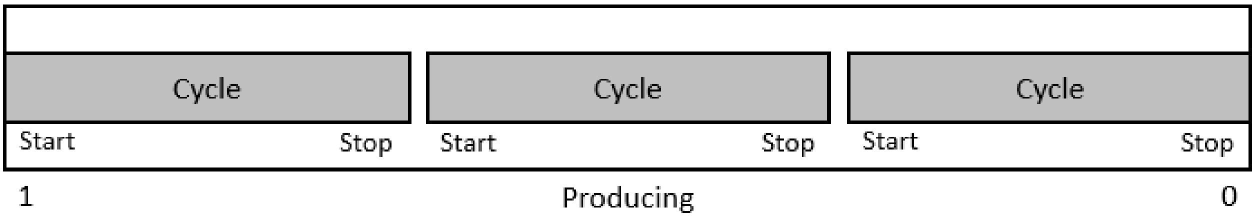
Productivity Diagnosis 3Ch

Is Producing	<Empty>	-
Start Cycle	<Empty>	-
Stop Cycle	<Empty>	-

Threshold Level Producing	1
Threshold Edge Start Cycle	1
Threshold Edge Stop Cycle	1
Produced Pieces	1
Production Time	Seconds 1200

Producing	*Empty*
Productivity [...	*Empty*
Productivity L...	*Empty*
Expected Pro...	*Empty*
Elapsed Time	*Empty*
Production C...	*Empty*

The *Productivity Diagnosis 3Ch* algorithm calculates the productivity of the process during a production interval. The diagram below schematically illustrates the relationship between the production process and the individual production cycles.



The production interval can be started and stopped via the input *Is Producing*. During the execution of the production interval, the production cycles are counted. Each production cycle corresponds to one piece produced. A production cycle starts with an edge at *Start Cycle* and stops with an edge at *Stop Cycle*. The productivity over the entire production interval (*Productivity*) is calculated after stopping the interval when the signal *Is Producing* no longer meets the condition for *Threshold Level Producing*. The completed production cycles and therefore all finished pieces are taken into account. Productivity is calculated as the ratio of pieces actually produced per time and the target value of pieces to be produced in a given time. The output

Productivity Last Cycle is calculated from the time required for the last production cycle in relation to the configured time for a piece. Any break times between cycles are not taken into account. The output *Expected Productivity* estimates the total productivity during the production interval. For this purpose, the previous production time is extrapolated to the total productivity for the target value of the pieces to be produced. The algorithm can be configured with the target value for the *Produced Pieces* within a configured interval *(Production Time)*, e.g. 1 piece in 30 seconds or 50 pieces per hour.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the Edge:** Specifies whether the algorithm should respond to a rising or falling edge. Can be configured individually for each threshold.
- **Threshold Level Producing**: Threshold of *Input Producing* at the respective edge. The *Production Time Interval* starts when the signal passes this threshold.
- **Threshold Edge Start Cycle:** Threshold of *Input Start Cycle* at the respective edge. The production cycle starts when the signal passes this threshold.
- **Threshold Edge Stop Cycle:** Threshold of *Input Stop Cycle* at the respective edge. The production cycle stops when the signal passes this threshold.
- **Produced Pieces:** Target value for pieces produced during the configured time interval (*Production Time*).
- **Production Time:** Time interval of the production time. It can be configured in days, hours, minutes or seconds.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

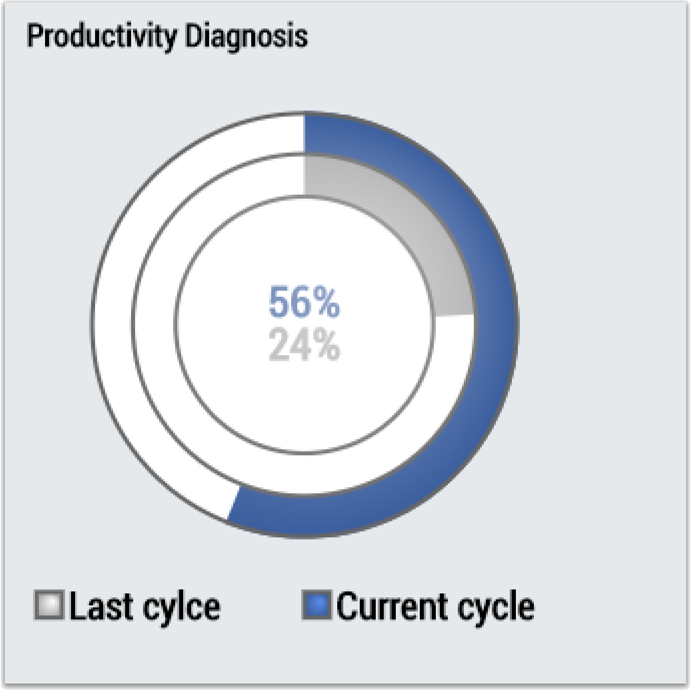
Output values

- **Producing:** Indicates whether the production interval is active.
- **Productivity:** Productivity of the entire production interval in percent.
- **Productivity Last Cycle:** Productivity of the last production cycle in percent.
- **Expected Productivity:** Estimates the productivity of the production interval. Specified in percent.
- **Elapsed Time:** Timespan since the start of the production interval.
- **Production Cycles:** Number of complete production cycles in the current production interval.

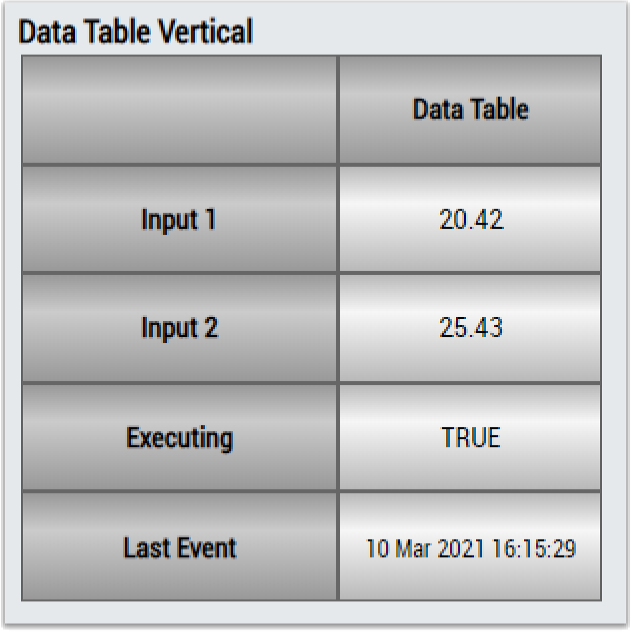
Standard HMI Controls

For the Productivity Diagnosis 3Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The ProductivityDiagnosis control visualizes the output values Productivity and Productivity Last Cycle.

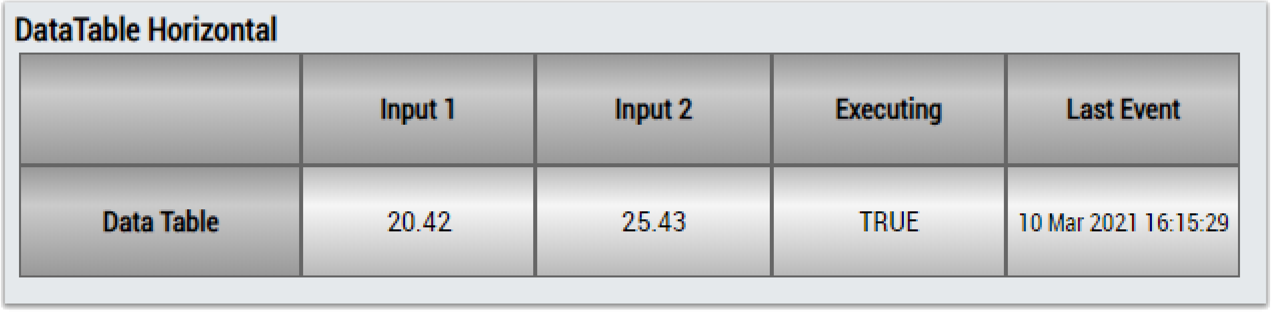


2. The Table Control or Multivalue Control visualizes all output values: New Result, Producing, Cycle Finished, Productivity, Productivity Last Cycle.



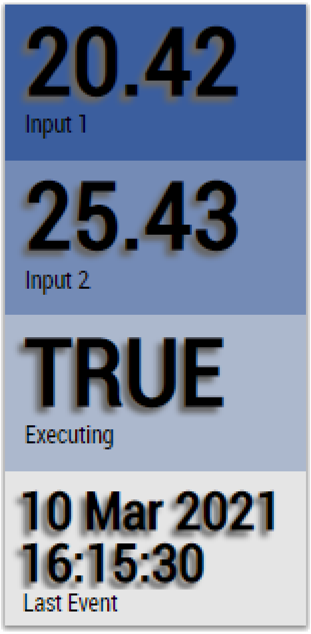
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

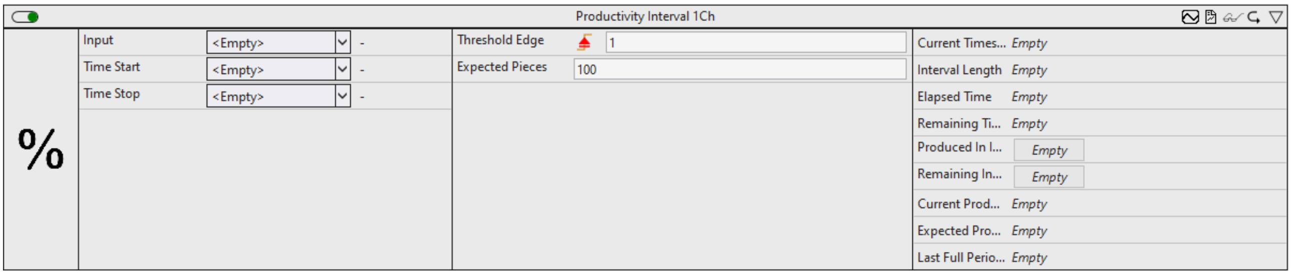
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Productivity Diagnosis 3Ch algorithm using the Mapping Wizard [▸ 432].

6.4.1.21 Productivity Interval 1Ch



Productivity Interval 1Ch

Input <Empty> -
Time Start <Empty> -
Time Stop <Empty> -

Threshold Edge 1
Expected Pieces 100

Current Times... Empty
Interval Length Empty
Elapsed Time Empty
Remaining Ti... Empty
Produced In I... Empty
Remaining In... Empty
Current Prod... Empty
Expected Pro... Empty
Last Full Perio... Empty

The algorithm *Productivity Interval 1Ch* calculates the productivity of the process during a given interval. The interval can be defined by the inputs *tTimeStart* and *tTimeStop*. The pieces produced are taken into account during execution. A produced element is counted when an edge is applied to the input. The estimated productivity of the current interval and the productivity of the last complete interval are provided as output values. The algorithm can be configured with the target value of the produced pieces within a given interval.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Type of the Edge:** Specifies whether the piece counter should respond to a rising or falling edge.
- **Threshold Edge**: Threshold of *Input* at which a manufactured piece is counted.
- **Expected Pieces:** Specification of the pieces to be produced within the defined timespan.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

Input values

- **Input:** Input value for counting the pieces produced.
- **Time Start:** Start time of the specified interval. Allows the interval to be specified from process data. Specified as time of day with a resolution of 1 ns.
- **Time Stop:** End time of the specified interval. Allows the interval to be specified from process data. Specified as time of day with a resolution of 1 ns.

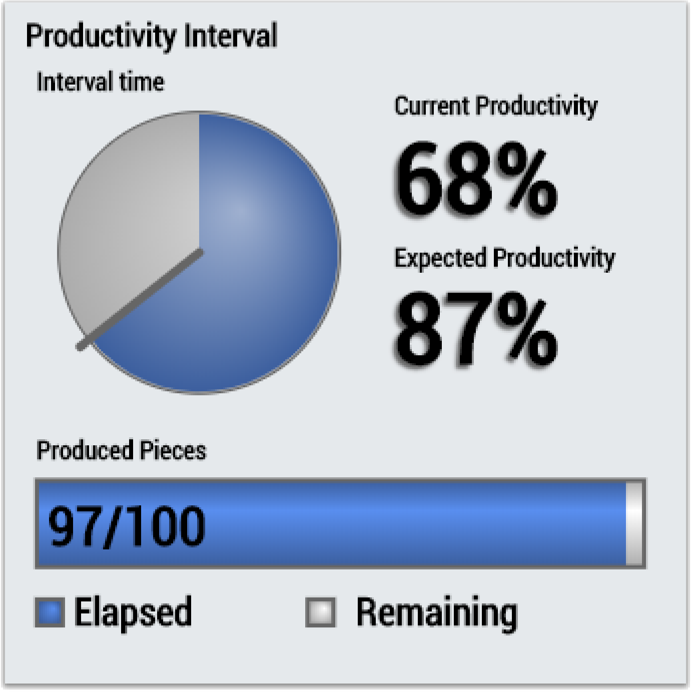
Output values

- **Within Interval:** Indicates whether the current time is within the interval.
- **Current Timestamp:** Current timestamp.
- **Interval Length:** Length of the interval.
- **Elapsed Time:** Elapsed time within the interval.
- **Remaining Time:** Remaining time within the interval.
- **Produced In Interval:** Produced pieces within the interval.
- **Remaining In Interval:** Remaining pieces within the interval.
- **Current Productivity:** Current productivity of the interval in percent. Takes into account the length of the interval, the time already elapsed, the pieces to be produced and the pieces already produced. The output is in percent.
- **Expected Productivity:** Expected productivity of the interval in percent. The production time of the last piece is used to estimate the number of pieces that can be produced in the remaining time.
- **Last Full Period Productivity:** Productivity of the last complete interval in percent. This is only calculated if the interval was fully processed.

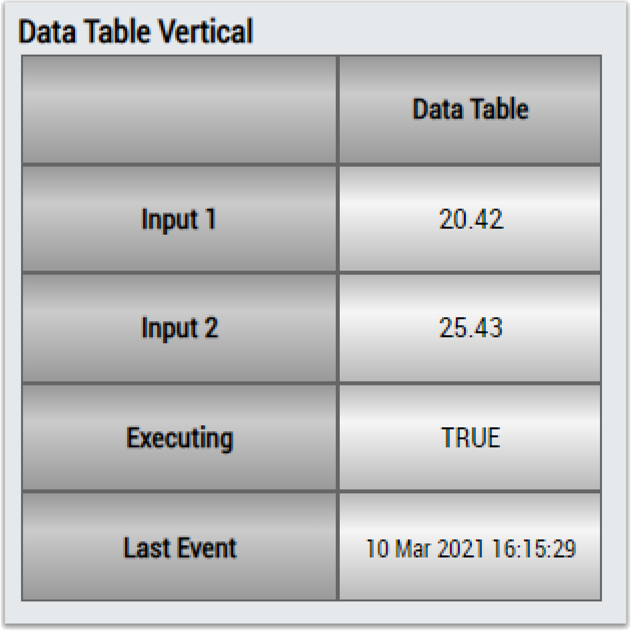
Standard HMI Controls

For the Productivity Interval 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The ProductivityInterval control visualizes the output values: time elapsed, time remaining, number of pieces produced in the interval, number of pieces remaining in the interval, productivity.

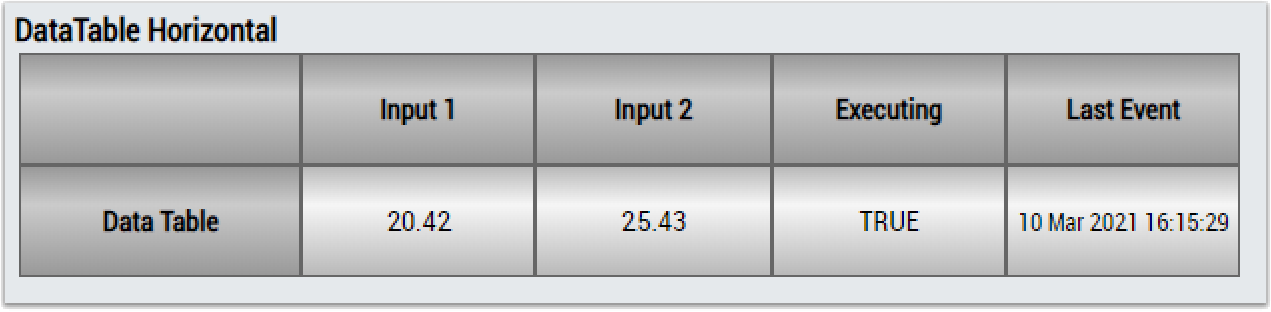


2. The Table Control or Multivalue Control visualizes all output values: Current time, interval length, elapsed time, remaining time, number of pieces produced in the interval, number of pieces remaining in the interval, productivity.



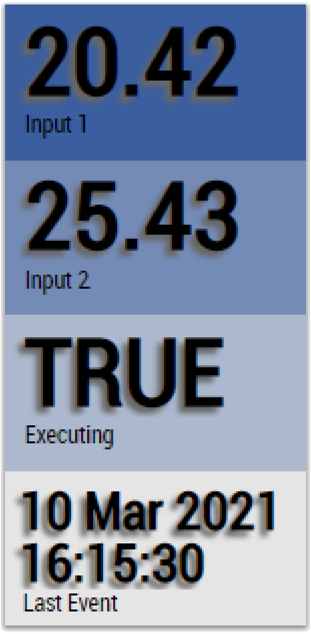
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

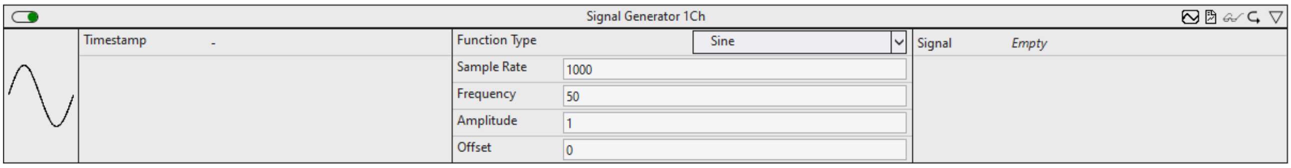
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Productivity Interval 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.22 Signal Generator 1Ch



Signal Generator 1Ch

Timestamp -

Function Type: Sine

Sample Rate: 1000

Frequency: 50

Amplitude: 1

Offset: 0

Signal: *Empty*

Signal Generator 1Ch can be used to generate various signal curves. The signal type, the frequency, the amplitude and the offset can be set individually. A timestamp is required as a reference value because the algorithm needs a time context in which to operate. This reference timestamp is automatically set if the configuration includes another algorithm. Therefore it is not possible to use the Signal Generator 1Ch individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Amplitude:** Configuration of the signal amplitude.
- **Frequency:** Frequency of the generated signal.
- **Function Type:** Function type of the generated signal.
  Const: constant
  Rectangle: rectangle function
  Sawtooth: sawtooth function
  Sine: sine function
  Triangle: triangle function
- **Offset:** constant offset of the generated signal.
- **Sample Rate:** sample rate of the system to be analyzed.

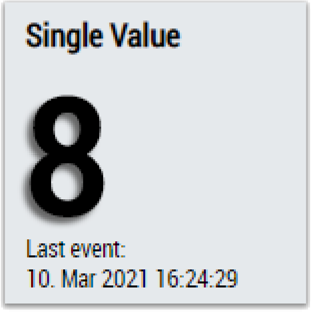
Output values

- **Signal:** outputs the generated signal.

Standard HMI Controls

For the Signal Generator 1Ch, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue control visualizes the signal output value.



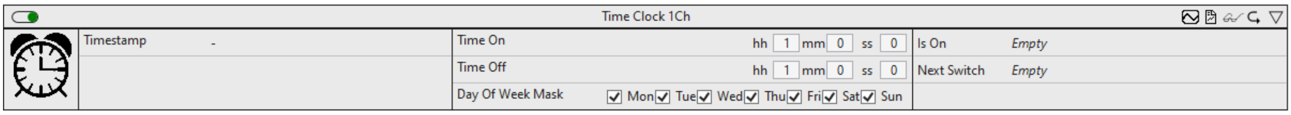
Single Value

8

Last event:
10. Mar 2021 16:24:29

Alternatively, customer-specific HMI controls can be mapped in the Signal Generator 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.23 Time Clock 1Ch



Time Clock 1Ch

Timestamp -

Time On: hh 1 mm 0 ss 0

Time Off: hh 1 mm 0 ss 0

Day Of Week Mask: ☑ Mon ☑ Tue ☑ Wed ☑ Thu ☑ Fri ☑ Sat ☑ Sun

Is On: *Empty*

Next Switch: *Empty*

Time Clock 1Ch executes a timer which can be configured with switch-on time, switch-off time and the days of the week on which the time switch should be active. A timestamp is required as a reference value because the algorithm needs a time context in which to operate. This reference timestamp is automatically set if the configuration includes another algorithm. Therefore it is not possible to use the Time Clock 1Ch individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Time On:** switch-on time.
- **Time Off:** switch-off time.
- **Day of Week Mask:** weekdays on which the timer should be active.

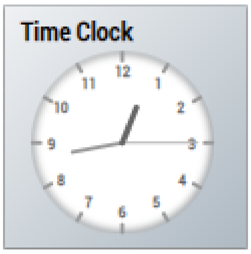
Output Values

- **Is On:** Shows *TRUE*, if the time switch is on, otherwise *FALSE*.
- **Next Switch:** Shows the remaining time up to the next switch.

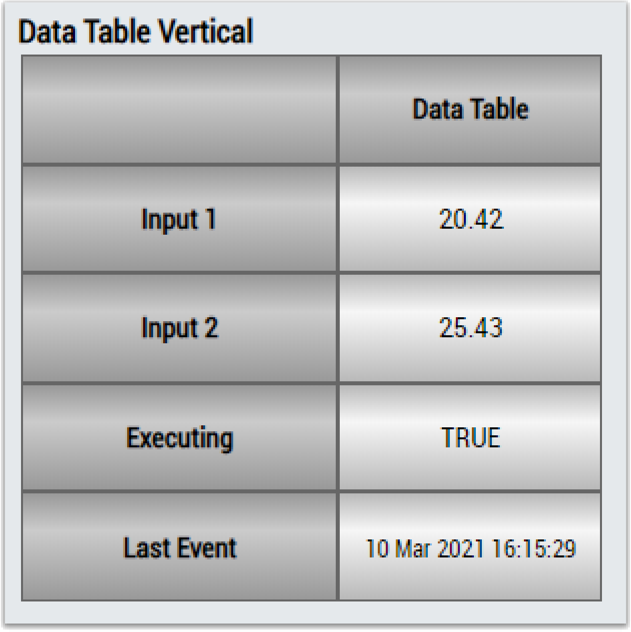
Standard HMI Controls

For the Timer Clock 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Timer Control visualizes the output value Next Switch.

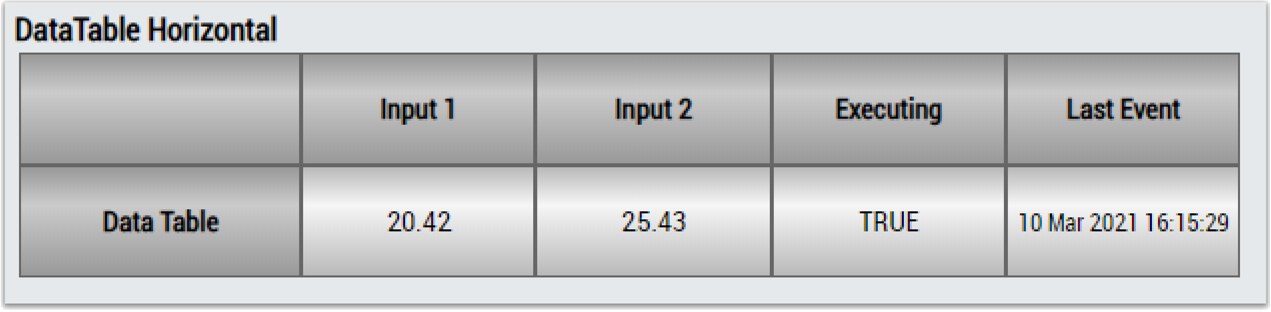


2. The Table Control or Multivalue Control visualizes all output values: Output, Next Switch.



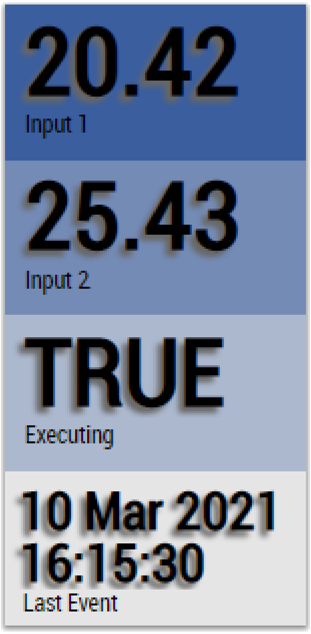
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

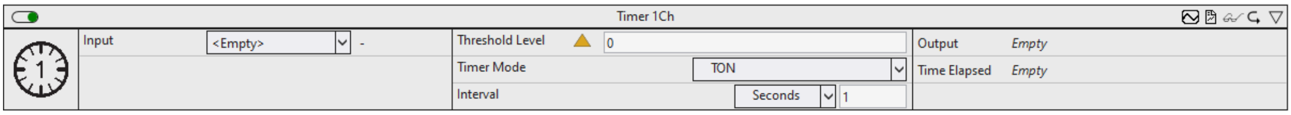
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Timer Clock 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.24 Timer 1Ch



Timer 1Ch

Input	<Empty>	Threshold Level	0	Output	Empty
		Timer Mode	TON	Time Elapsed	Empty
		Interval	Seconds 1		

The *Timer 1Ch* starts a timer which can be configured by timer mode and interval. According to the specific timer mode the timer is started, if the configured condition becomes *TRUE* (TON, TP) or the condition becomes *FALSE* (TOF).

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** Specifies whether the input value should be greater than, greater than or equal to, equal to, less than or equal to, or less than the threshold.
- **Threshold:** Signal threshold.
- **Timer Mode:** mode of the timer:

  TON: The TON timer is a switch-on delay timer that enables the output after the threshold condition becomes *TRUE* and the timespan specified in the interval has elapsed.

  TOF: The TOF timer is a switch-off delay timer that disables the output after the threshold condition becomes *FALSE* and the timespan specified in the interval has elapsed.

  TP: The TP timer is a pulse generator that activates the output for the time specified in the interval after the threshold condition becomes *TRUE*.
- **Interval:** Time interval of the configured timer.

- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

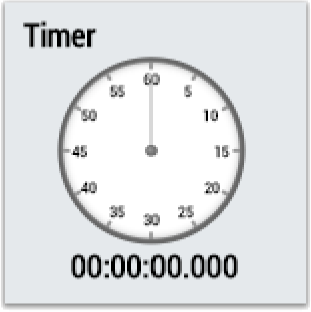
Output Values

- **Output:** Shows the output value which is affected by the configured timer.
- **Elapsed Time:** Shows the elapsed time of the timer.

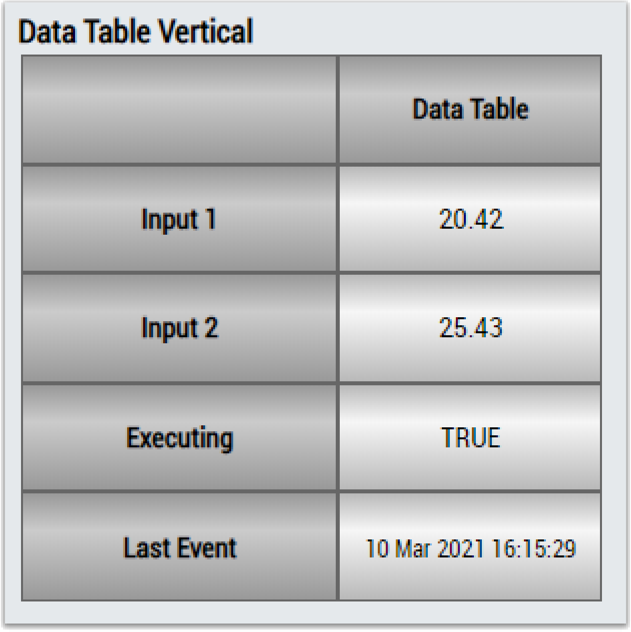
Standard HMI Controls

For the Timer 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

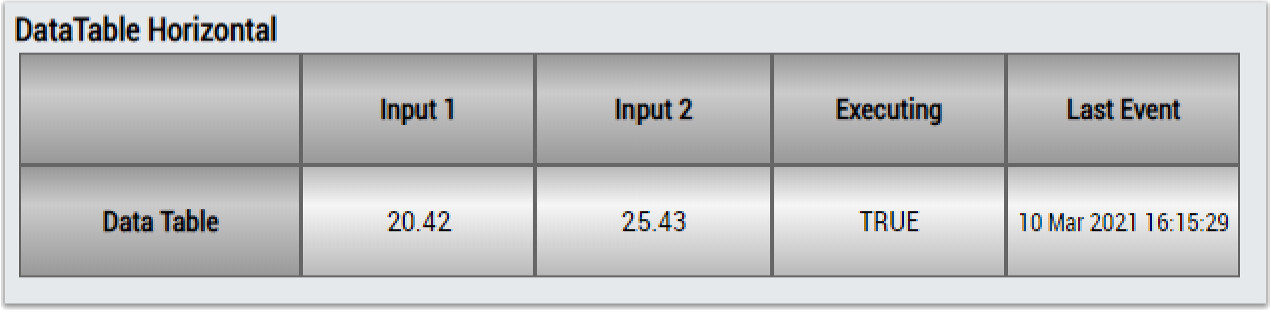
1. The Timer Control visualizes the output value Elapsed Time.



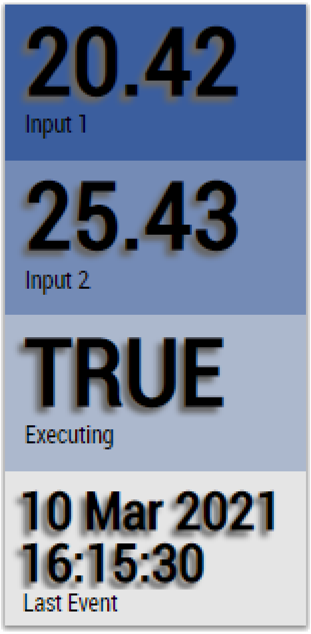
2. The Table Control or Multivalue Control visualizes all output values: Output, Elapsed Time.


Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29


DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

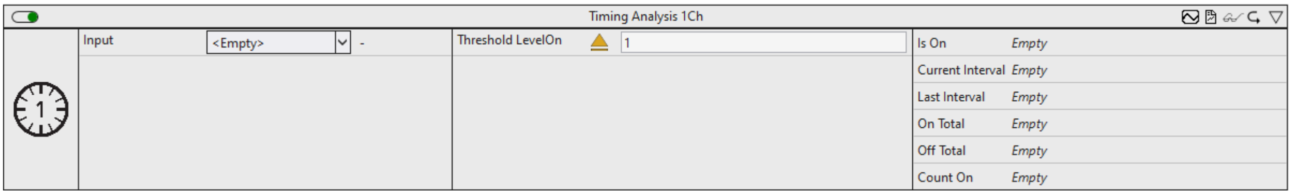
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Timer 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.1.25 Timing Analysis 1Ch



Timing Analysis 1Ch

Input <Empty> -

Threshold LevelOn 1

Is On	Empty
Current Interval	Empty
Last Interval	Empty
On Total	Empty
Off Total	Empty
Count On	Empty

The *Timing Analysis 1Ch* measures time differences between on- and off-periods and counts the amount of on-periods. The on-period starts when the condition of operator and threshold is met.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** Specifies whether the input value should be greater than, greater than or equal to, less than or equal to, less than or not equal to the threshold.
- **Threshold:** Signal threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

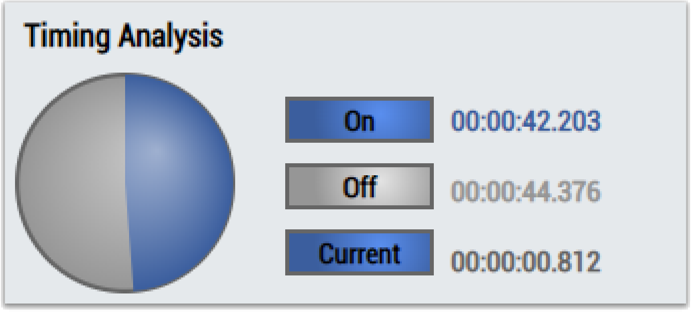
Output values

- **Is On:** Indicates *TRUE* within the timespan of the On period, otherwise *FALSE*.
- **Current Interval:** Indicates the time of the current interval.
- **Last Interval:** Timespan of the last completed interval.
- **On Total:** Indicates the total time for which the “Is On” value is *TRUE*.
- **Off Total:** Displays the total time for which the “Is On” value is *FALSE*.
- **Count On:** Counts the number of triggered On events.

Standard HMI Controls

For the Timing Analysis 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The TimingAnalysis Control visualizes the output values Is On, On Total, Off Total and Current Interval.



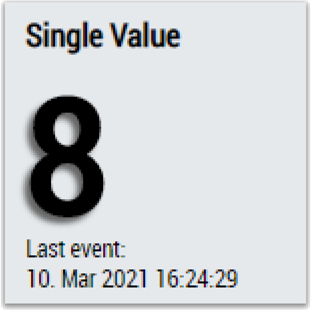
Timing Analysis

On 00:00:42.203

Off 00:00:44.376

Current 00:00:00.812

2. The SingleValue control visualizes the output value Count On.

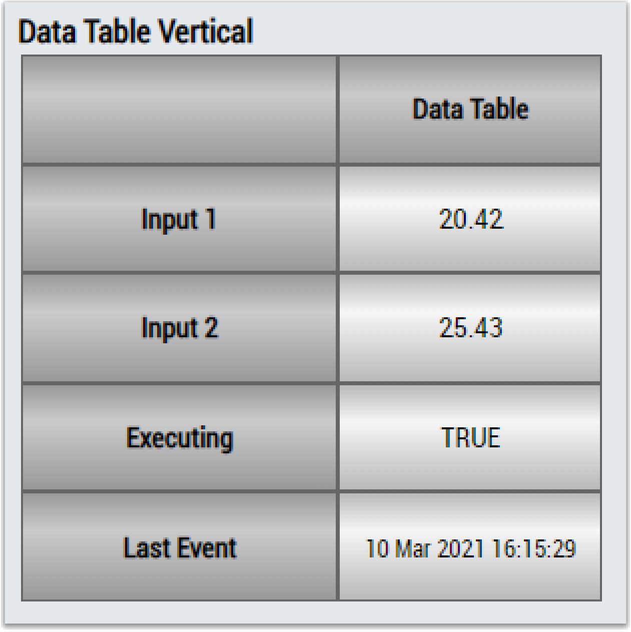


Single Value

8

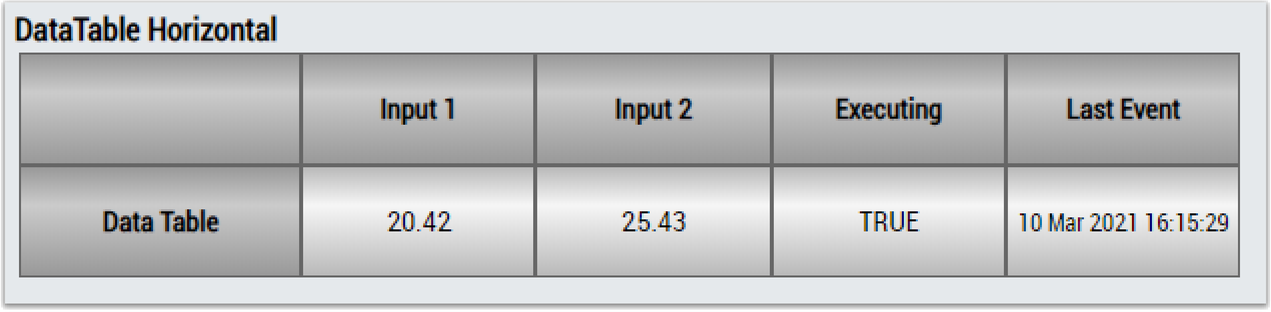
Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Is On, Count On, Current Interval, On Total, Off Total.



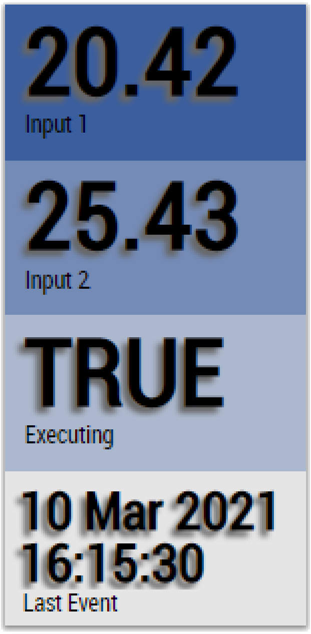
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

TRUE
Executing

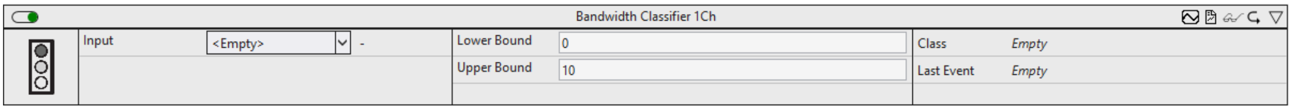
10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Timing Analysis 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2 Analytics - Classification

The algorithms of the category *Analytics-Classification* provide functionalities for classification and state detection.

6.4.2.1 Bandwidth Classifier 1Ch



Input <Empty> -

Lower Bound 0

Upper Bound 10

Class *Empty*

Last Event *Empty*

Bandwidth Classifier 1Ch determines whether the input signal is within the configured limits or is less than or greater than the limits.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Lower Bound:** Lower Bound for the comparison.
- **Upper Bound:** Upper Bound for the comparison.

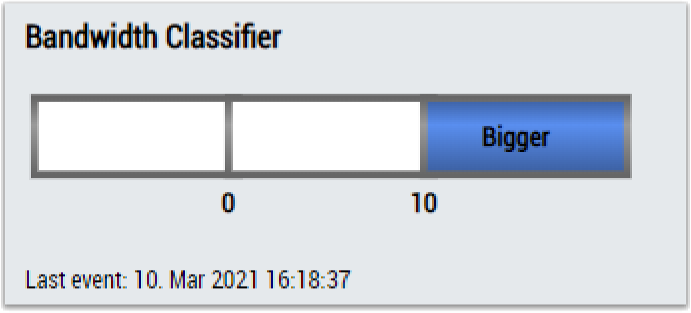
Output values

- **Class:** Indicates the class to which the input values belong (WithinBounds/Smaller/Bigger).
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

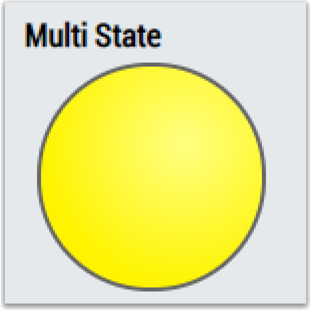
For the Bandwidth Classifier 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The BandwidthClassifier control visualizes the output values Class and Last Event and the configuration options Lower Bound and Upper Bound.

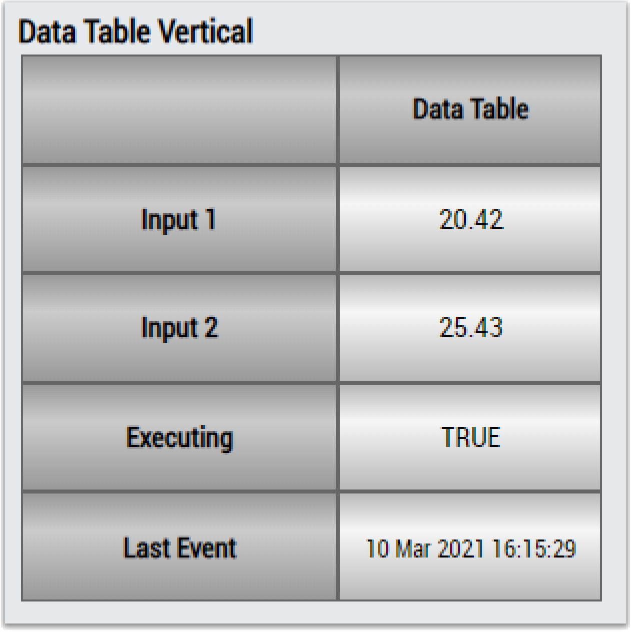


Last event: 10. Mar 2021 16:18:37

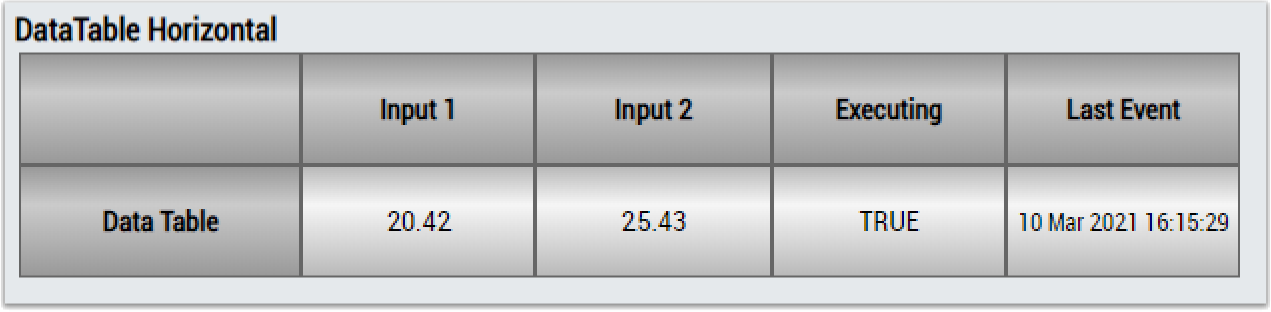
2. The MultiState control visualizes the output value Class.



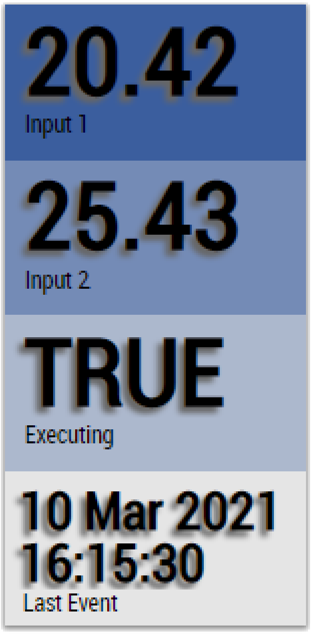
3. The Table Control or Multivalue Control visualizes all output values: Class, Last Event.


Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29


DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

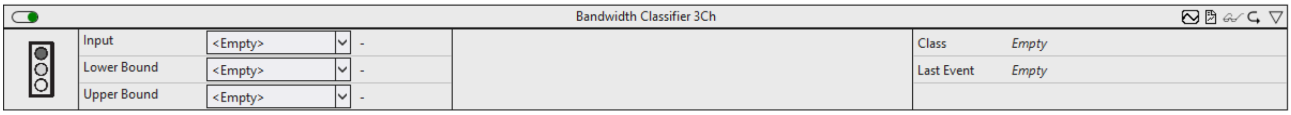
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Bandwidth Classifier 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.2 Bandwidth Classifier 3 Ch



Bandwidth Classifier 3Ch

Input	<Empty>	-	Class	Empty
Lower Bound	<Empty>	-	Last Event	Empty
Upper Bound	<Empty>	-		

Bandwidth Classifier 3Ch determines whether the input signal is within the limits or is less than or greater than the limits. The limits can be configured with input signals, so it is possible to use curves as lower and upper band.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

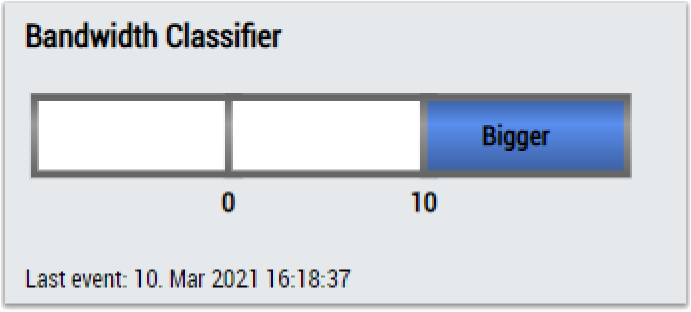
Output values

- **Class:** Indicates the class to which the input values belong (WithinBounds/Smaller/Bigger).
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Bandwidth Classifier 3Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

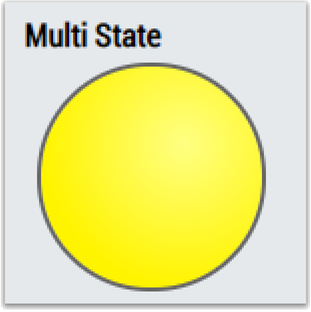
1. The BandwidthClassifier control visualizes the output values Class and Last Event and Input Lower Bound and Input Upper Bound.



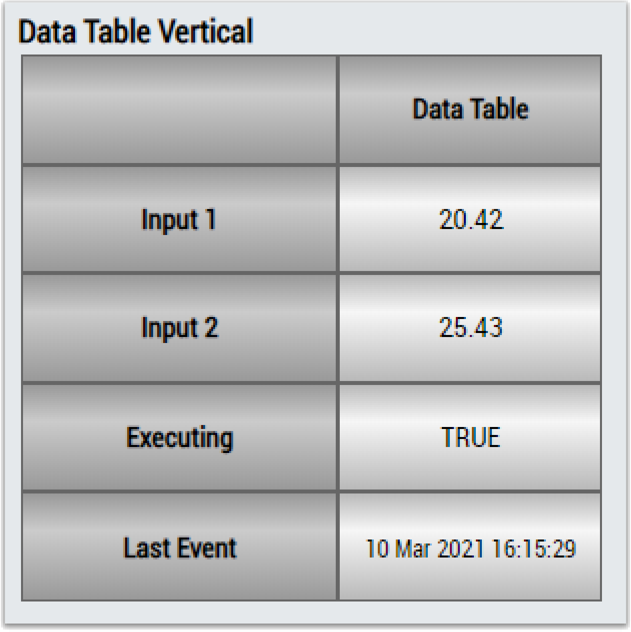
Bandwidth Classifier

Last event: 10. Mar 2021 16:18:37

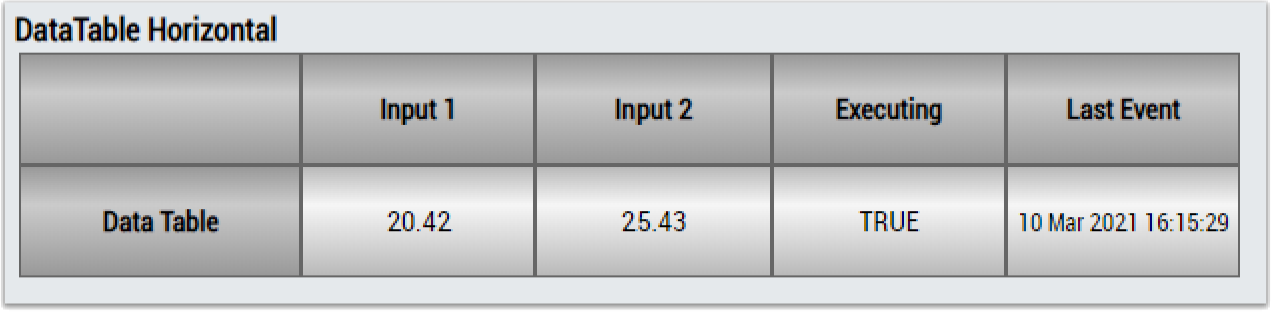
2. The MultiState control visualizes the output value Class.



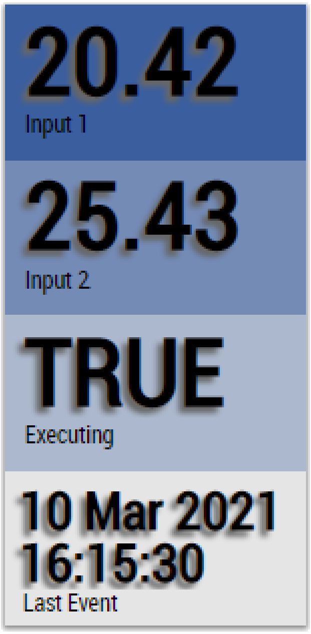
3. The Table Control visualizes all output values: Class, Last Event.


Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29


DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

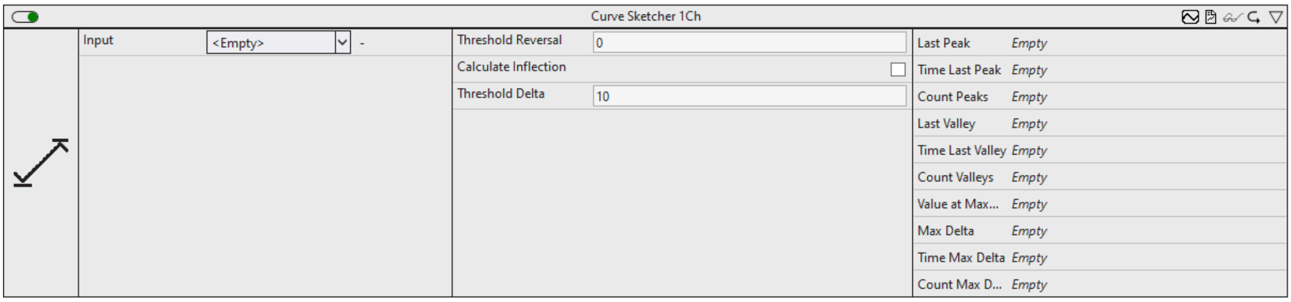
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Bandwidth Classifier 3Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.3 Curve Sketcher 1Ch



Curve Sketcher 1Ch

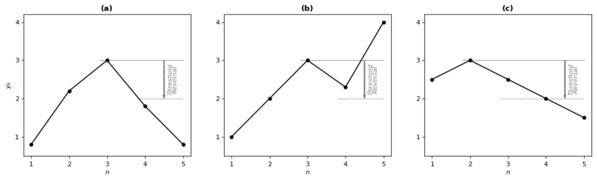
Input	<Empty> -	Threshold Reversal	0	Last Peak	Empty
		Calculate Inflection	□	Time Last Peak	Empty
		Threshold Delta	10	Count Peaks	Empty
				Last Valley	Empty
				Time Last Valley	Empty
				Count Valleys	Empty
				Value at Max...	Empty
				Max Delta	Empty
				Time Max Delta	Empty
				Count Max D...	Empty

Curve Sketcher 1Ch identifies inversions (*peaks* and *valleys*) in an input data stream. Furthermore, local maxima of the absolute difference between two consecutive values (referred to as *Delta*) can be identified. Analogous to a continuous curve, the identified peaks and valleys correspond to local maxima and minima. Delta corresponds to the slope, so that a maximum of the absolute values of Delta can be associated with an inflection point.

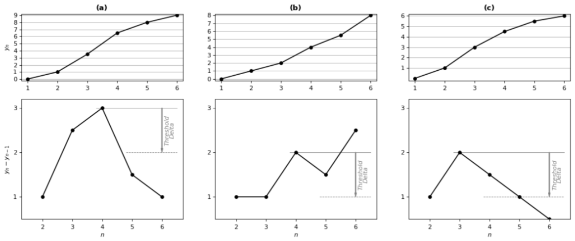
Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Calculate Inflection:** Boolean flag. Maxima of the rate of change are only identified if this flag is *True*. Otherwise, the values for *Count Max Delta, Max Delta*, *Time Max Delta* and*Value at Max Delta* will not be calculated.
- **Threshold Reversal:** threshold for identifying reversals. Reversals are only detected if their difference from the next reversal exceeds the value of Threshold Reversal.
  Below are three examples of peak identification using the parameter *Threshold Reversal*.
  (a) The value y_3 is identified as a peak immediately after processing the value y_4 because the difference between y_3 and y_4 is greater than *Threshold Reversal*.
  (b) The value y_3 is not identified as a peak because the difference between y_3 and y_4 is smaller than *Threshold Reversal* and the curve starts rising again after y_4.
  (c) The value y_2 is identified as a peak after processing the value y_5 because the difference between y_2 and y_5 *exceeds Threshold Reversal*. The value y_2 cannot be identified as a peak beforehand because the difference between y_2 and y_3 (y_4) is less than/equal to *Threshold Reversal* and it is not known whether the values will continue to decrease.



- **Threshold Delta:** threshold for identifying Delta maxima. Maxima of the absolute difference of two successive values (delta) are detected only if the difference between successive deltas exceeds *Threshold Delta*.
  Below are three examples of identifying the Delta maxima with the parameter *Threshold Delta* . The upper diagrams show the original input signals, the lower ones the corresponding delta.
  (a) The value y_4 is identified as a maximum after processing the value y_5 because the difference between the two deltas exceeds *Threshold Delta*.
  (b) No maximum is identified because the difference between the deltas is less than *Threshold Delta*.
  (c) The value y_3 is identified as a maximum after processing the value y_6.



Regardless of *Threshold Delta*, at least one maximum of the Delta between two reversals is detected.

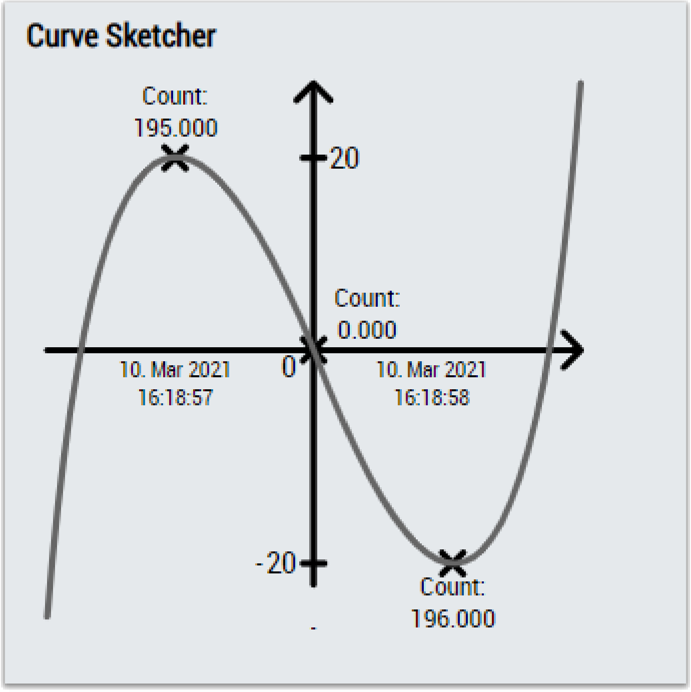
Output Values

- **Last Peak:** Indicates the y-value of the last identified peak.
- **Time Last Peak:** Indicates the timestamp of the last identified peak.
- **Count Peaks:** Indicates the total number of counted peaks.
- **Last Valley:** Indicates the y-value of the last identified valley.
- **Time Last Valley:** Indicates the timestamp of the last identified valley.
- **Count Valleys:** Indicates the total number of counted valleys.
- **Value at Max Delta:** Indicates the y-value of the analyzed stream (input variable) that is led by the last detected maximum of delta. The value delta is the difference between *Value at Max Delta* and the value that reached one cycle before.
- **Max Delta:** Indicates the last identified local maximum of the absolute difference between two successive values in the input stream.
- **Time Max Delta:** This is the timestamp of *Value at Max Delta.*
- **Count Max Delta:** Indicates the total number of counted local maxima of delta.

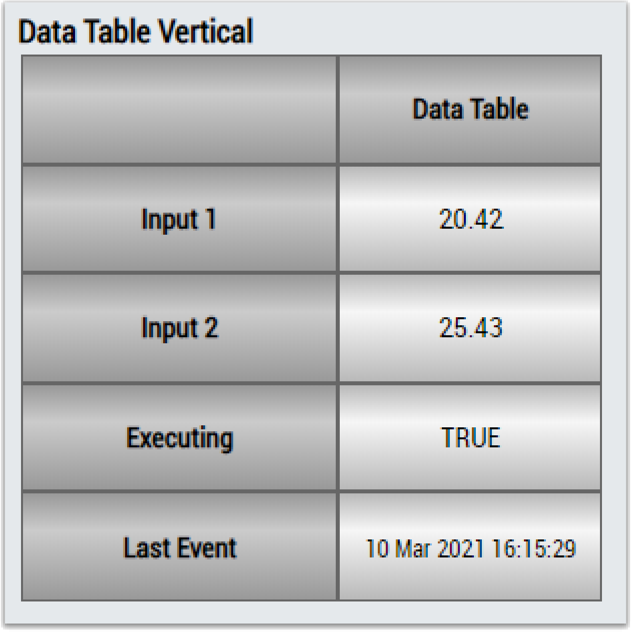
Standard HMI Controls

For the Curve Sketcher 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. CurveSketcher control visualizes the output values Last Peak, Time Last Peak, Count Peaks, Last Valley, Time Last Valley, Count Valley, Last Delta, Time Last Delta and Count Delta as well as the input value of the data.

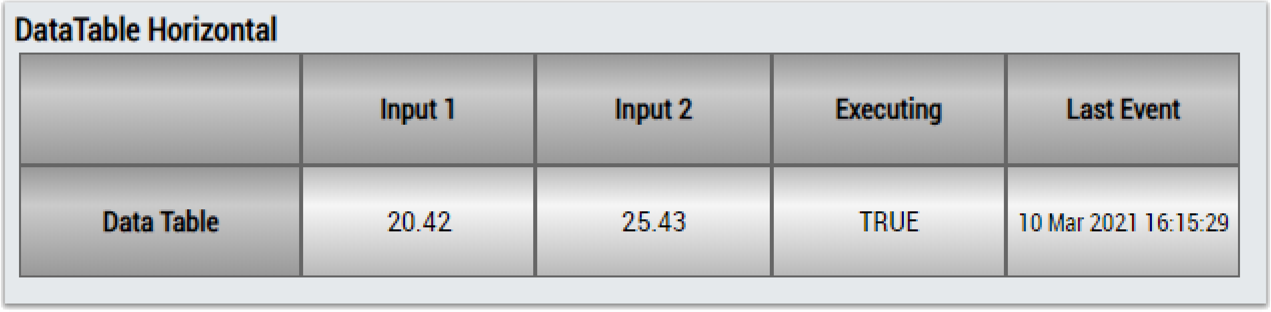


2. The Table Control visualizes all output values: Last Peak, Time Last Peak, Count Peaks, Last Valley, Time Last Valley, Count Valley, Value at Max Delta, Last Delta, Time Last Delta and Count Delta.



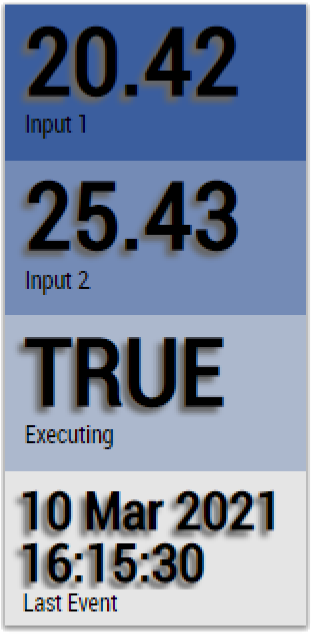
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

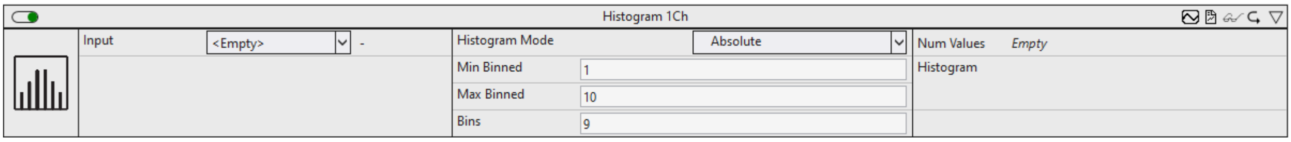
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Curve Sketcher 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.4 Histogram 1Ch



Histogram 1Ch

Input	<Empty>	-	
Histogram Mode	Absolute	Num Values	*Empty*
Min Binned	1	Histogram	
Max Binned	10		
Bins	9		

The *Histogram 1Ch* calculates the distribution of a single channel input value cyclically. It can be configured with minimal bin, maximal bin and the total amount of bins. The dimension of the output array is the number of bins + 2. Because values that are less than the minimal bin are stored in the first array element and values greater than the maximal bin are stored in the last array element.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Histogram Mode:** The operation mode of the histogram. A distinction is made between absolute and relative output. The latter can be used to display the percentage distribution.
- **Min Binned:** Minimum value to be analyzed.
- **Max Binned:** Maximum value to be analyzed.
- **Bins:** Total number of histogram classes to be calculated.
- **Enable Display Names**: Enable Histogram Display Names. This configuration is visible hidden in the property window.
- **Histogram Display Names:** Display names of the individual bins. This configuration is hidden in the Property window and is used for the HMI.

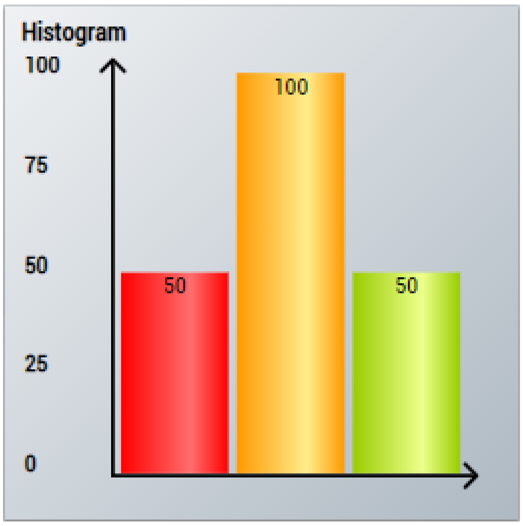
Output values

- **Num Values:** Indicates the total number of analyzed values for the distribution.
- **Histogram:** The histogram is displayed below the Num Values. If you move the cursor over the bars, you will see a tooltip with the value interval and the corresponding value of the histogram bin.

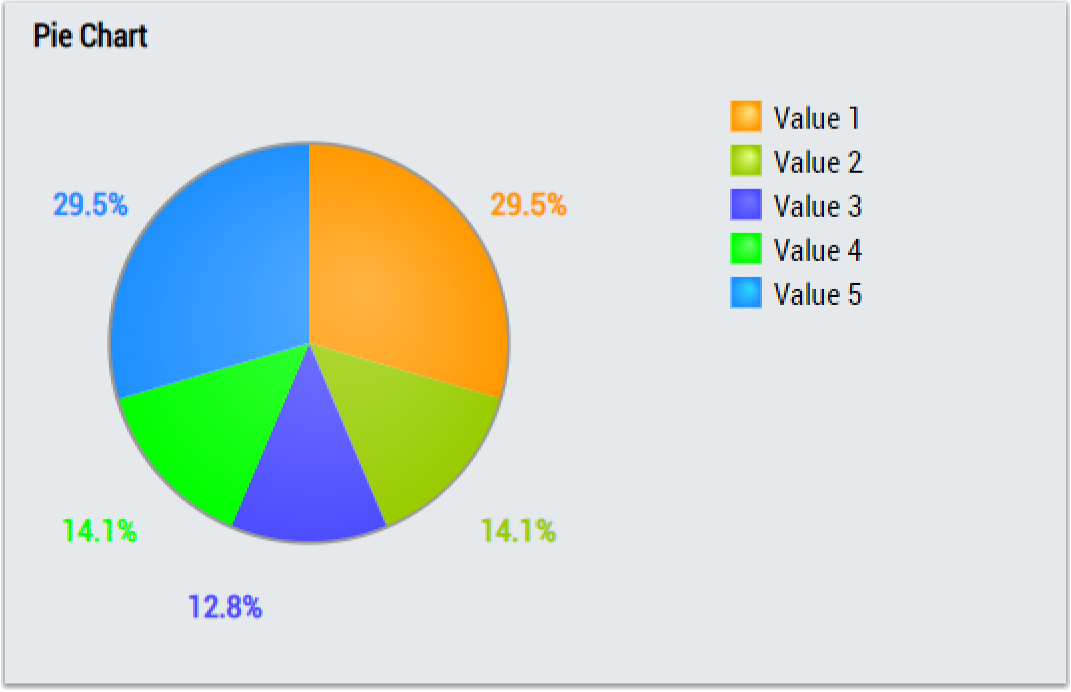
Standard HMI Controls

For the Histogram 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

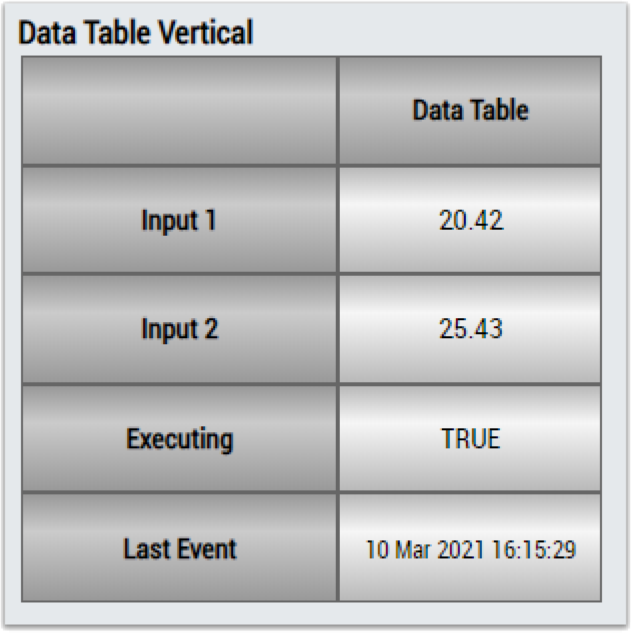
1. The Histogram control visualizes the output value Num Values.



2. The PieChart control visualizes the output value Num Values.

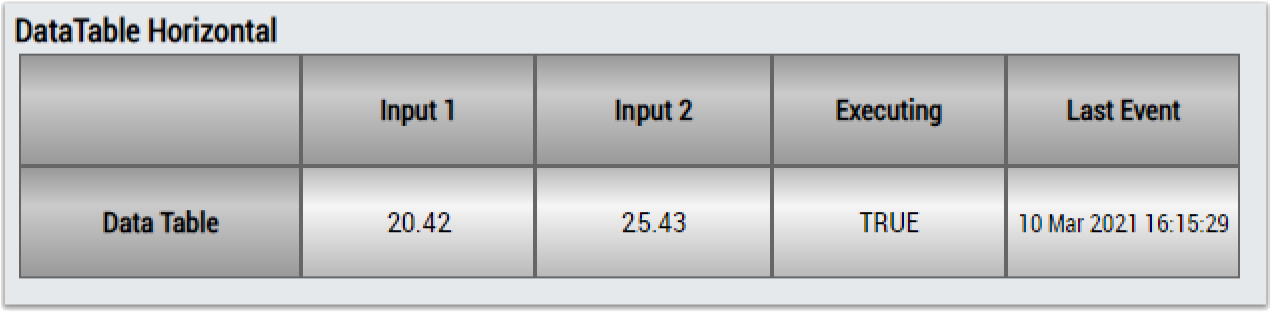


3. The Table Control or Multivalue Control visualizes all output values: Num Values, Histogram.



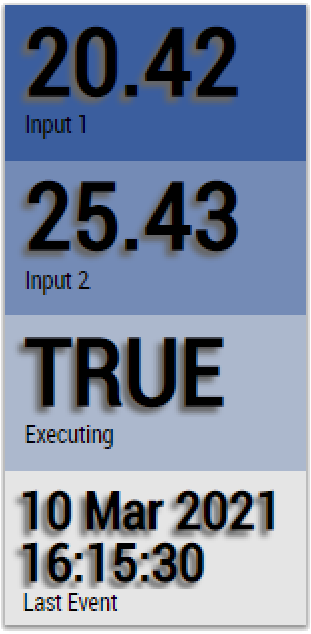
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

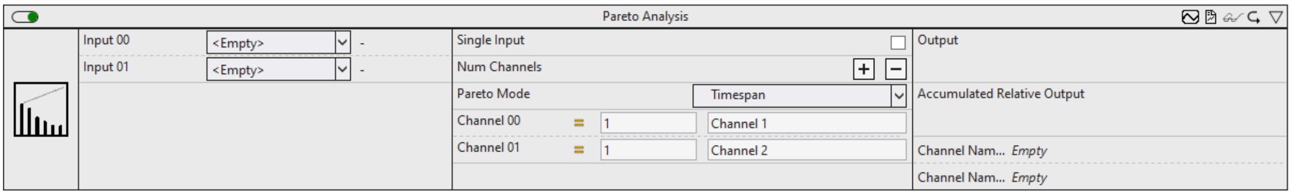
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Histogram 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.5 Pareto Analysis



Pareto Analysis

Input 00 <Empty>
Input 01 <Empty>

Single Input □
Num Channels
Pareto Mode: Timespan
Channel 00 = 1 Channel 1
Channel 01 = 1 Channel 2

Output
Accumulated Relative Output
Channel Nam... *Empty*
Channel Nam... *Empty*

The *Pareto Analysis* module allows you to combine different input data to display a Pareto chart. This is used in the following applications, among others:

Error analysis: Identifying the most common sources of errors

Quality management: Determining the main causes of quality problems

Time management: Diagnosis of which tasks take up most of the time

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Single Input:** Selection of whether a single input or several independent input channels should be used.
- **Num Channels:** Number of input channels if *Single Input* is `FALSE`.
- **Num Elements:** Number of elements to be compared if *Single Input* is `TRUE`.
- **Pareto Mode:** Mode for calculating the output values.

  Count: The cycles in which the respective input channel is `TRUE` are counted.

  Timespan: The timespans in which the input channel is `TRUE` are added up.
- **Threshold 00, Threshold 01, ..., Threshold n**: Comparison operator and threshold for monitoring input values.
- **Channel Name 00, Channel Name 01, ..., Channel Name n**: Alias of the input channel.

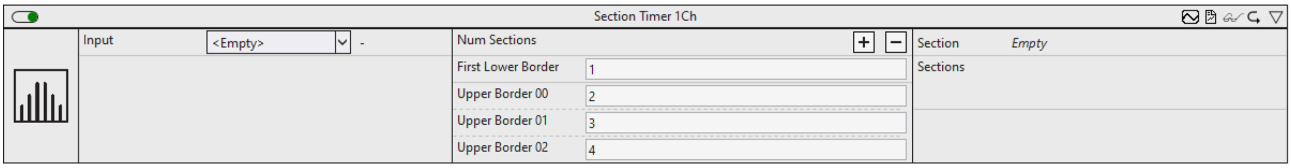
If the configuration option *Single Input* is set and an input of type Enumeration is linked to the input, configuration options are set automatically:

Number of input channels: 1
Number of elements to be compared: Number of enumerations
Threshold values: Values of the individual enumerations

Output values

- **Output:** Sorted absolute values. The output type depends on the Pareto mode.
- **Accumulated Relative Output**: Accumulated relative output values.
- **Channel Name 00, Channel Name 01, ... Channel Name n:** Sorted alias names of the input channels.

6.4.2.6 Section Timer 1 Ch



Section Timer 1Ch

Input <Empty>

Num Sections
First Lower Border 1
Upper Border 00 2
Upper Border 01 3
Upper Border 02 4

Section *Empty*
Sections

The *Section Timer 1Ch* calculates the timespan the input is in range of each configured section. It can be configured with the amount of sections and the borders of each section. Each section is defined with lower border (greater than or equal to) and upper border (less than). The following sections lower border is set by the previous upper border. Values that are less than the minimal border are stored in the first array element. Values that are greater than or equal to the maximal border are stored in the last array element.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Sections:** This is the number of sections.
- **First Lower Border:** This is the lower border of the first section.
- **Upper Border 00, Upper Border 01, ..., Upper Border n**: These are the upper borders of all sections.
- **Enable Display Names**: Enable Section Display Names. This configuration is visible hidden in the property window.
- **Section Display Names:** Display names of the individual sections. This configuration is hidden in the Property window and is used for the HMI.

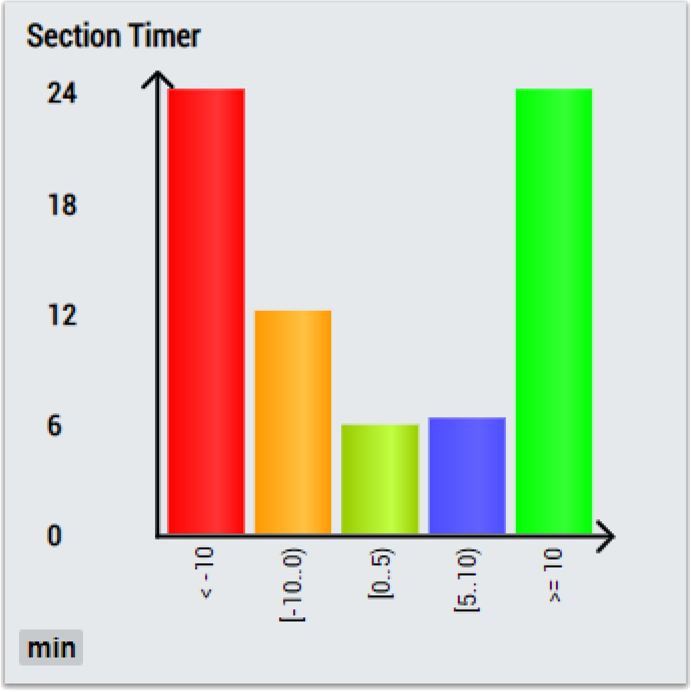
Output values

- **Section:** specifies the section of the last input value. If the input value is less than the *First Lower Border*, the return value is zero. If the input value is in the interval [*First Lower Border*, *Upper Border 00*), the return value is one, for the interval [*Upper Border 00*, *UpperBorder 01*) it is two, etc. If the input value is greater than the last specified limit *Upper Border 0n*, the return value is *NumSections*+1.
- **Array of Timespans**: total time during which the input value was sorted into each section.

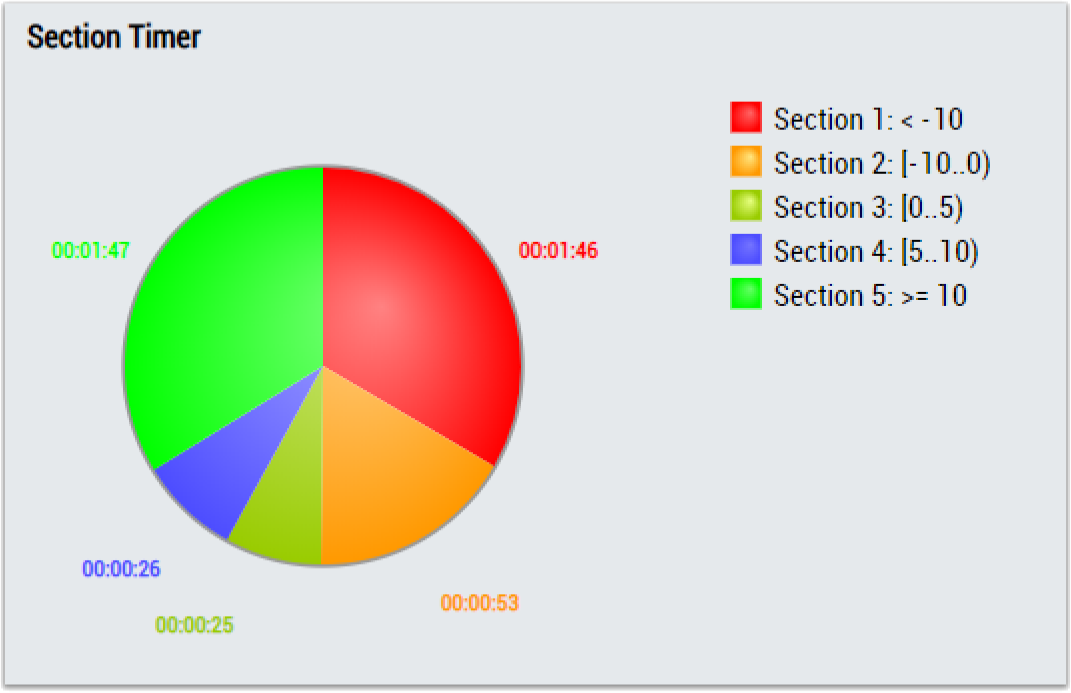
Standard HMI Controls

For the Section Timer 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SectionTimer control visualizes the input value Array of Timespans and the configuration options First Lower Border and Upper Border.

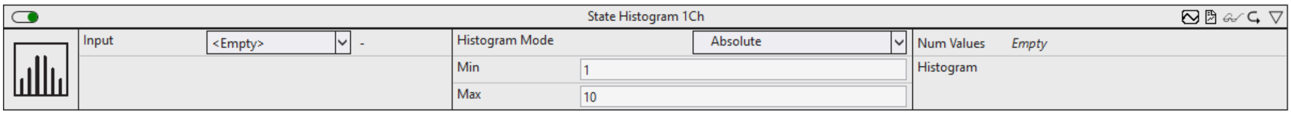


2. The PieChart control visualizes the input value Array of Timespans and the configuration options First Lower Border and Upper Border.



Alternatively, customer-specific HMI controls can be mapped in the Section Timer 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.7 State Histogram 1Ch



State Histogram 1Ch

Input <Empty> -

Histogram Mode Absolute

Min 1

Max 10

Num Values *Empty*

Histogram

The *State Histogram 1Ch* counts how often the input signal (INT) has a specific value between the configured minimum and maximum and shows the distribution in a histogram. The first bar represents the boundary values which are smaller than the minimum and the last bar represents the boundary values which are greater than the maximum. The *State Histogram 1Ch* is suitable for state-machines to show how often the different states are executed.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Histogram Mode:** The operation mode of the histogram. A distinction is made between absolute and relative output. The latter can be used to display the percentage distribution.
- **Min:** Minimum value to be analyzed.
- **Max:** Maximum value to be analyzed.
- **Enable Display Names**: Enable Histogram Display Names. This configuration is visible hidden in the property window.
- **Histogram Display Names:** Display names of the individual bins. This configuration is hidden in the Property window and is used for the HMI.

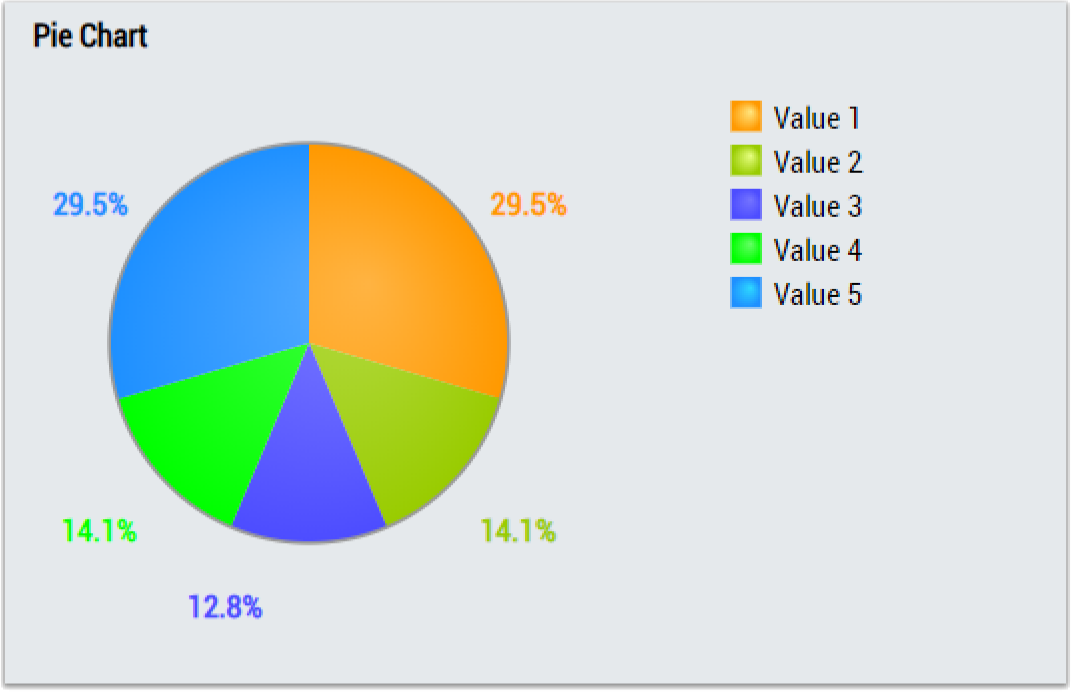
Output values

- **Num Values:** Indicates the total number of states executed between the configured borders.
- **Histogram:** The histogram is displayed below the Num Values. The respective value is indicated on each bar. If you move the cursor over the bars, you will see a tooltip with the value interval and the corresponding value of the histogram bin.

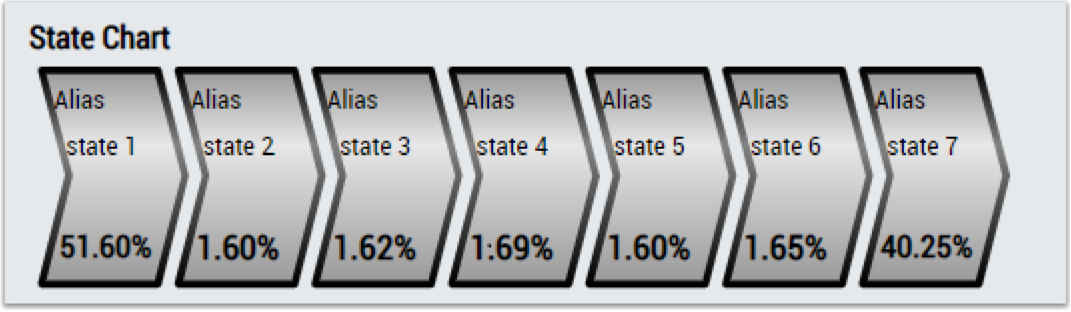
Standard HMI Controls

For the State Histogram 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

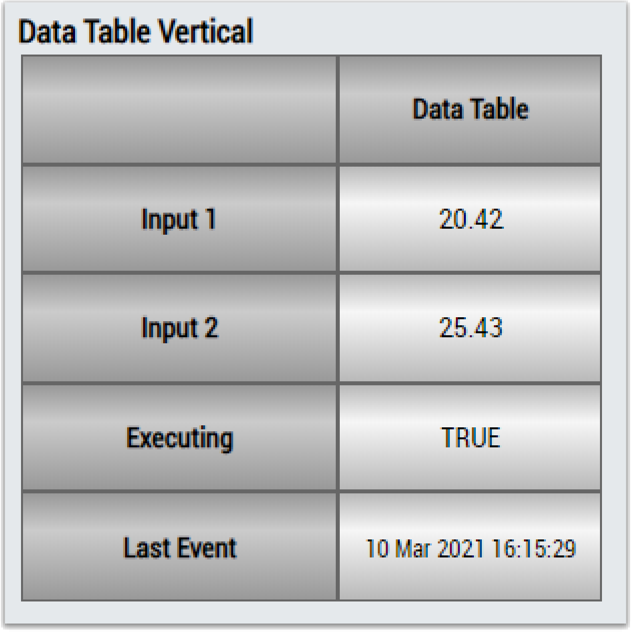
1. The PieChart control visualizes the output value histogram.



2. The StateHistogram control visualizes the output value histogram.

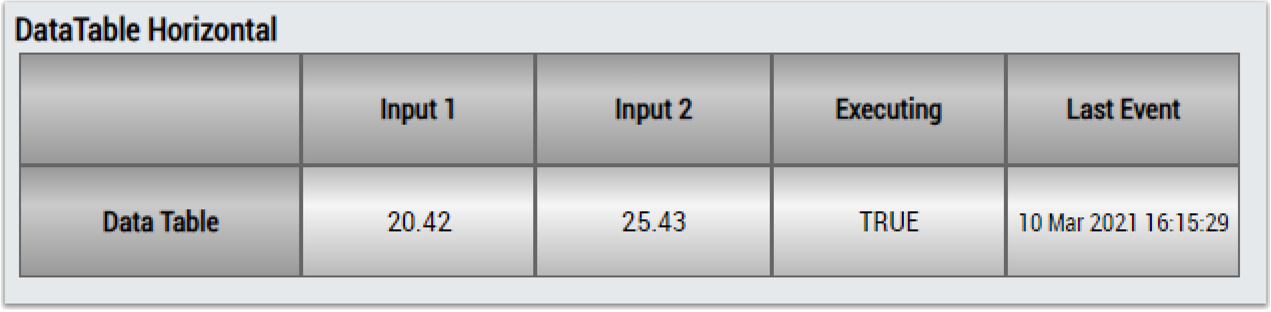


3. The Table Control or Multivalue Control visualizes all output values: Num Values, Histogram.



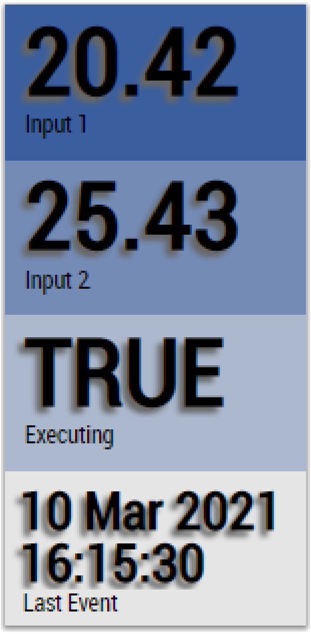
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

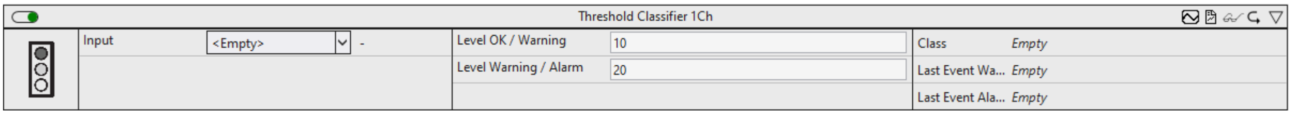
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the State Histogram 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.8 Threshold Classifier 1Ch



Threshold Classifier 1Ch

Input <Empty> -

Level OK / Warning 10

Level Warning / Alarm 20

Class *Empty*

Last Event Wa... *Empty*

Last Event Ala... *Empty*

Threshold Classifier 1Ch classifies the input values into three different classes: *OK*, *Warning* and *Alarm* according to the configured thresholds.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Level Ok / Warning:** Input values that are less than the configured threshold are classified as *OK;* input values that are equal to or greater than the configured threshold are classified as *Warning*.
- **Level Warning / Alarm:** Input values that are less than the configured threshold are classified as *Warning;* input values that are equal to or greater than the configured threshold are classified as *Alarm*.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

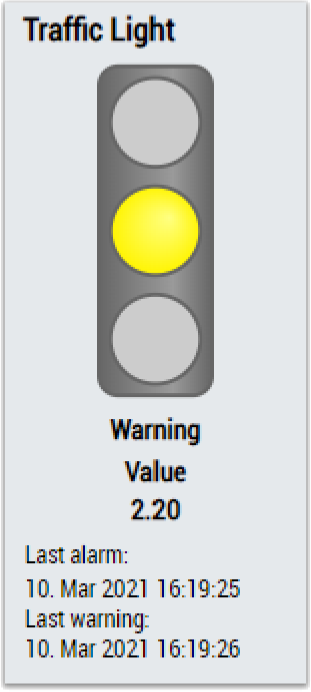
Output values

- **Class:** Indicates the class to which the input values belong.
- **Last Event Warning:** Indicates the time of the last triggered warning event → the event can be dragged and dropped into the scope chart to display it as a trigger-event.
- **Last Event Alarm:** Indicates the time of the last triggered alarm event → the event can be dragged and dropped into the scope chart to display it as a trigger-event.

Standard HMI Controls

For the Threshold Classifier 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The TrafficLight Control visualizes the output values Class, Last Event Warning and Last Event Alarm as well as the input value of the data.



Traffic Light

Warning

Value

2.20

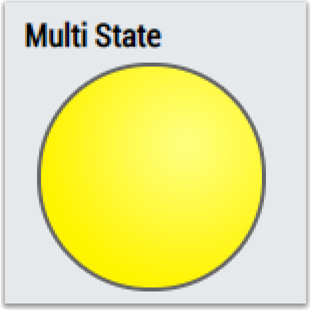
Last alarm:

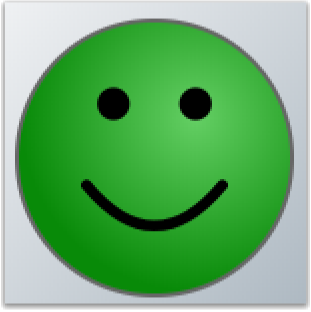
10. Mar 2021 16:19:25

Last warning:

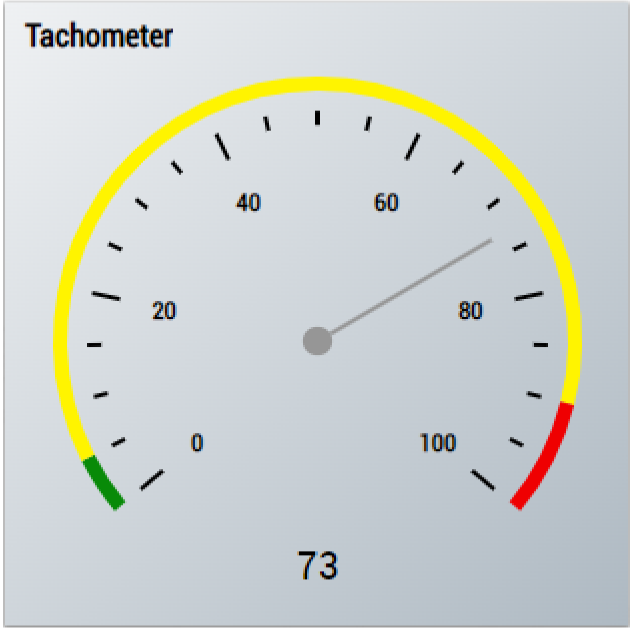
10. Mar 2021 16:19:26

2. The MultiState Control visualizes the Class output value, optionally with a smiley.

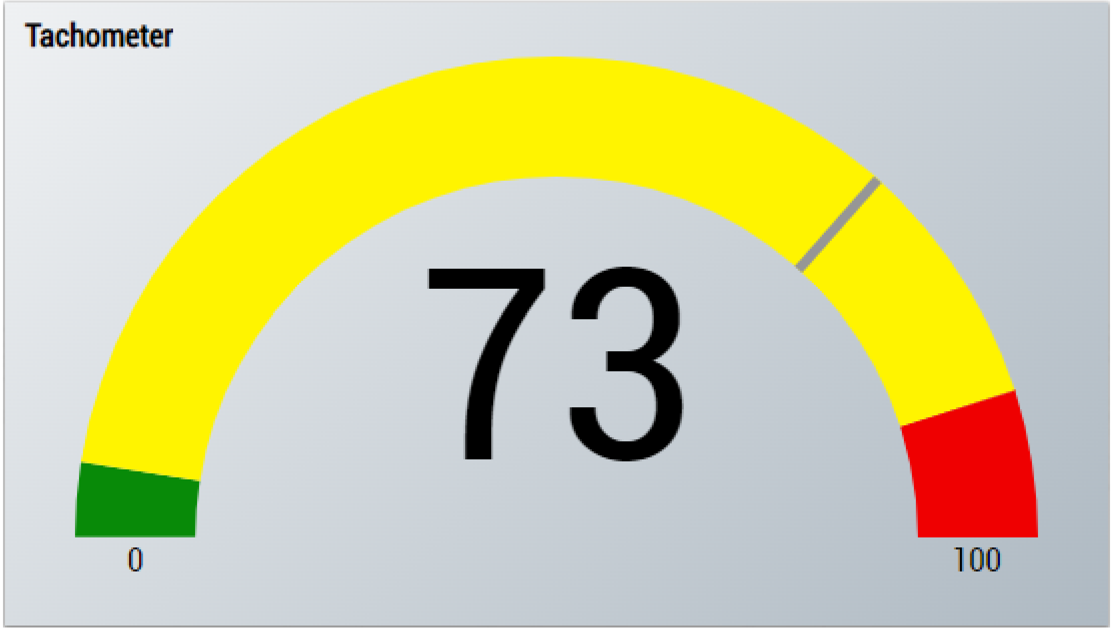




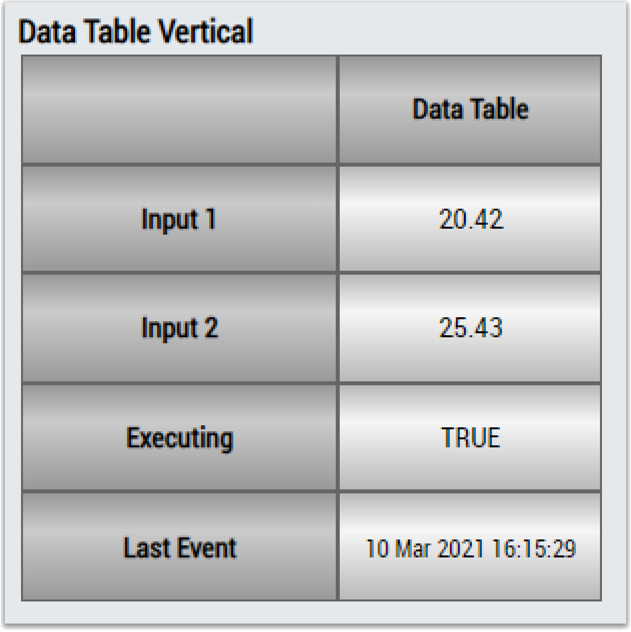
3. The Tachometer Control visualizes the input value including the configured limit values.



4. The Radial Gauge Control visualizes the input value including the configured limit values.

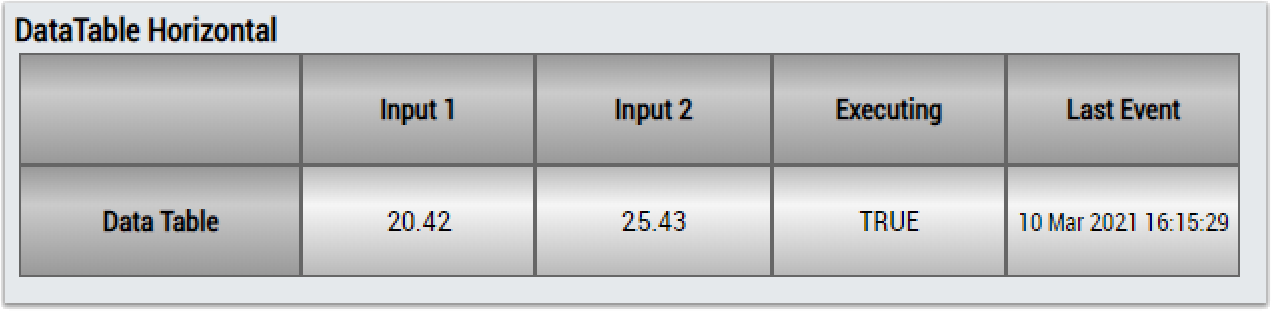


5. The Table Control or Multivalue Control visualizes all output values: Class, Last Event Warning Last Event Alarm.



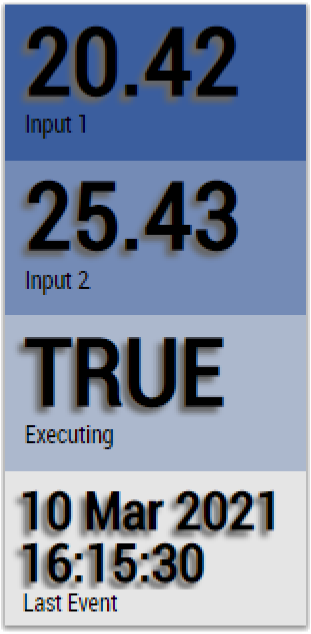
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

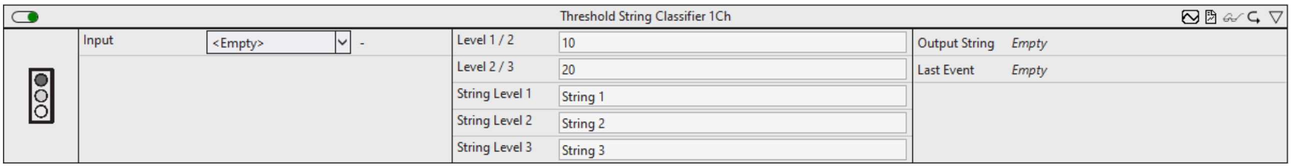
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Threshold Classifier 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.9 Threshold String Classifier 1Ch



Threshold String Classifier 1Ch

Input	<Empty>	Level 1 / 2	10	Output String	Empty
		Level 2 / 3	20	Last Event	Empty
		String Level 1	String 1		
		String Level 2	String 2		
		String Level 3	String 3		

The *Threshold String Classifier 1Ch* algorithm classifies the input values into three different classes according to the configured thresholds. The class names (output string) can be configured individually as *String 1*, *String 2* and *String 3*.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Level 1 / 2:** Input values that are less than the configured threshold are assigned to the *first* class; input values that are equal to or greater than the configured threshold are assigned to the *second* class.
- **Level 2 / 3:** Input values that are less than the configured threshold are assigned to the *second* class; input values that are equal to or greater than the configured threshold are assigned to the *third* class.
- **String 1:** Name of the first class.
- **String 2:** Name of the second class.
- **String 3:** Name of the third class.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.

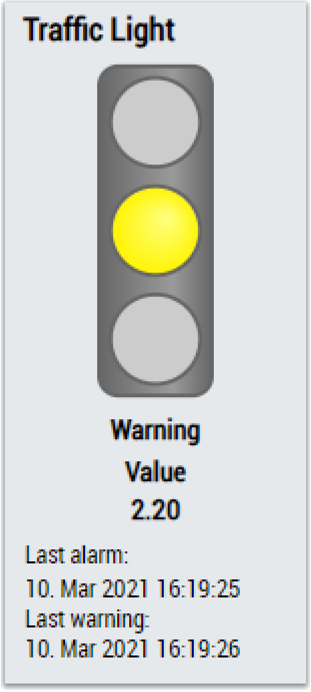
Output values

- **Output String:** Indicates the class to which the input values belong.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the Threshold String Classifier 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The TrafficLight control visualizes the output values Output String and Last Event as well as the input value of the data.



Traffic Light

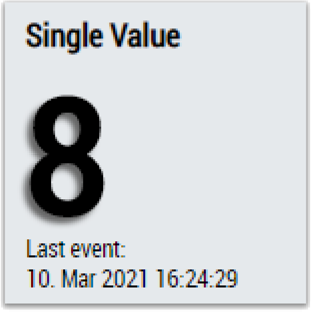
Warning

Value

2.20

Last alarm:
10. Mar 2021 16:19:25
Last warning:
10. Mar 2021 16:19:26

2. The SingleValue control visualizes the output value Output String.

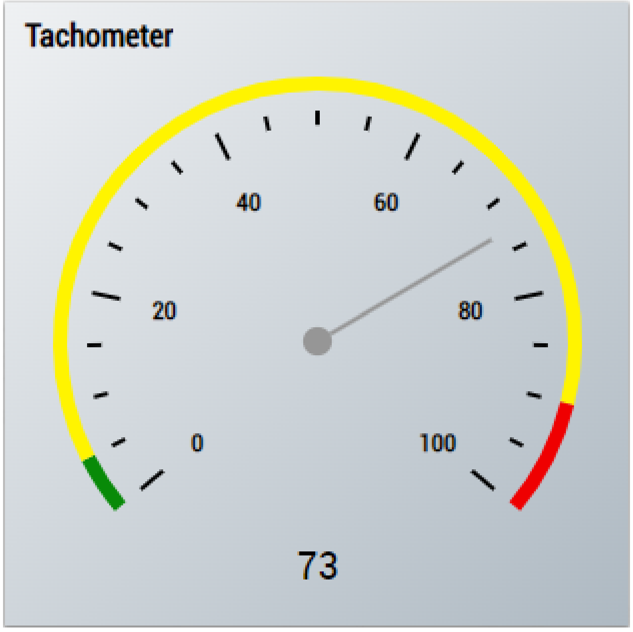


Single Value

8

Last event:
10. Mar 2021 16:24:29

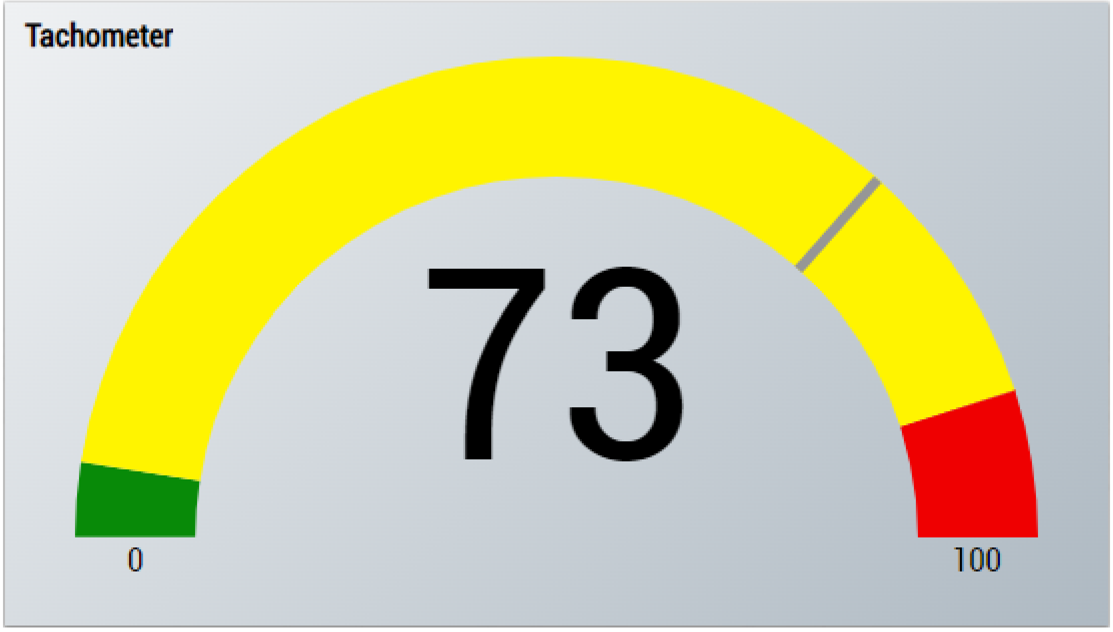
3. The Tachometer Control visualizes the input value including the configured limit values.



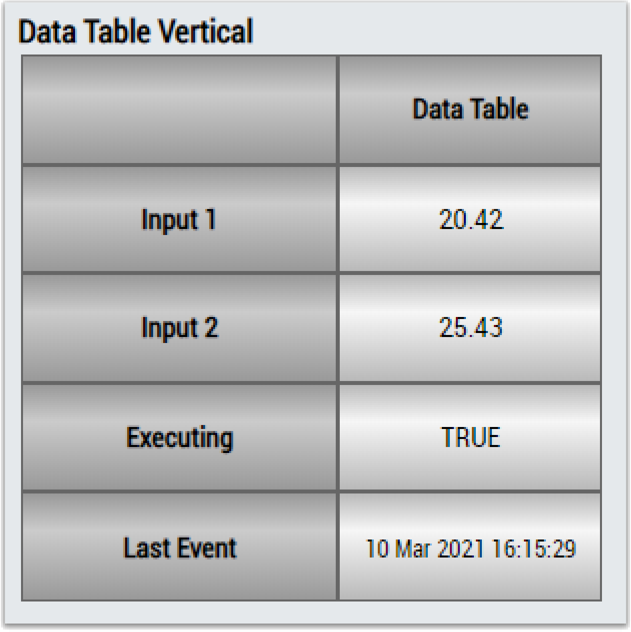
Tachometer

73

4. The Radial Gauge Control visualizes the input value including the configured limit values.

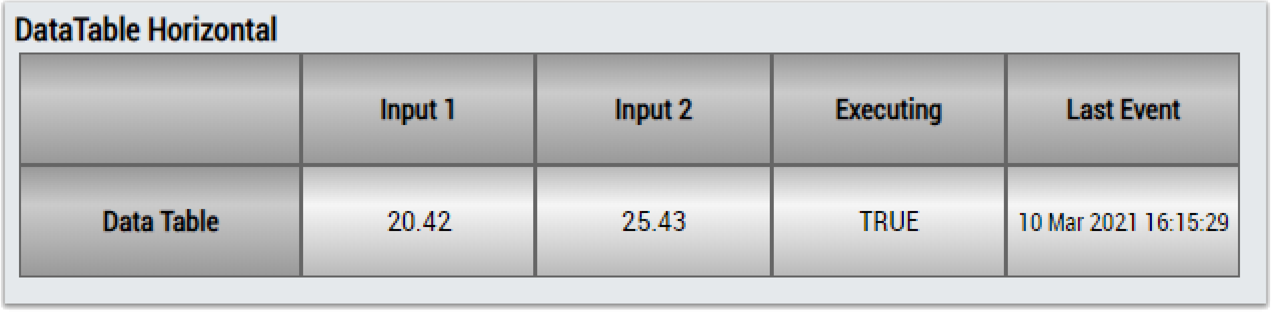


5. The Table Control or Multivalue Control visualizes all output values: Output String, Last Event.



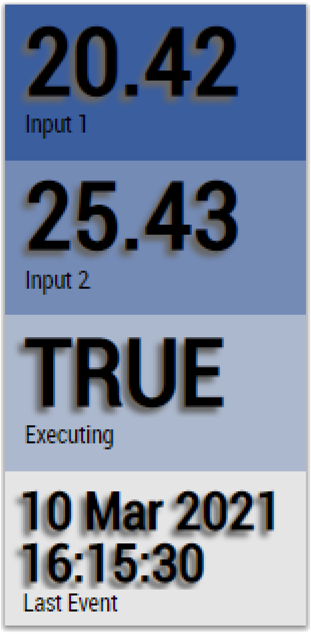
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

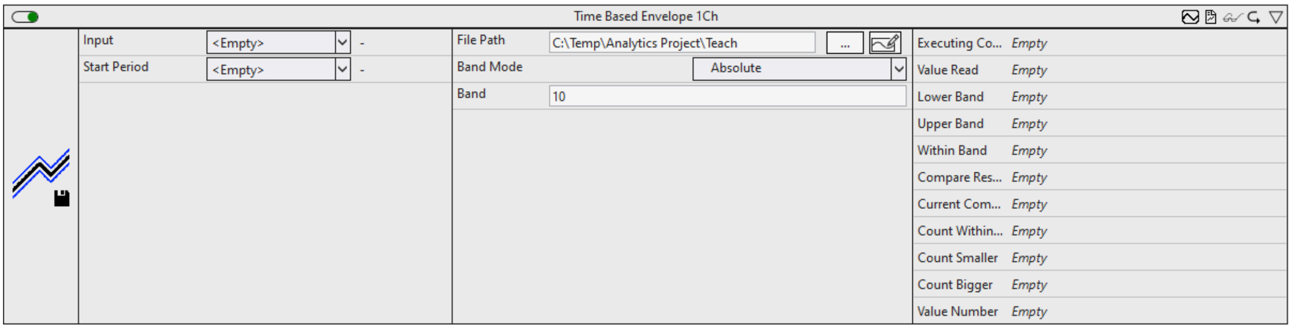
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Threshold String Classifier 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.2.10 Time Based Envelope 1Ch



Time Based Envelope 1Ch

Input	<Empty>	-	
Start Period	<Empty>	-	
File Path	C:\Temp\Analytics Project\Teach		
Band Mode	Absolute		
Band	10		

Executing Co...	Empty
Value Read	Empty
Lower Band	Empty
Upper Band	Empty
Within Band	Empty
Compare Res...	Empty
Current Com...	Empty
Count Within...	Empty
Count Smaller	Empty
Count Bigger	Empty
Value Number	Empty

The *Time Based Envelope 1Ch* algorithm checks whether the periodic input data is within a configured range of values read from a file. This can be a reference signal that was previously learned with Time Based Teach Path 1Ch [▶ 251], for example. The comparison starts when the signal of the Start Period flag is *TRUE*. It is recommended not to use *Time Based Envelope 1Ch* simultaneously with Time Based Teach Path 1Ch [▶ 251] due to concurrent file access. Instead, a reference signal should first be taught in with Time Based Teach Path 1Ch [▶ 251] and only then should the evaluation be carried out with the aid of the *Time Based Envelope 1Ch*.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **File Path:** Path to the previously teached data file.
- **Band Mode:** Mode of the band operation (use absolute or relative values).
- **Band:** Bandwidth of the band operation.

Output values

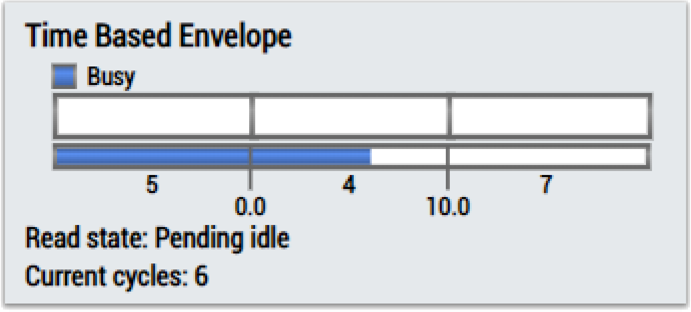
- **Executing Compare:** displays *TRUE* if the algorithm is processing the envelope, otherwise *FALSE*. The envelope process begins when the Start Period flag is *TRUE*.
- **Lower Band:** displays the value of the lower band depending on the band mode.
- **Upper Band:** displays the value of the upper band depending on the band mode.
- **Within Band:** displays *TRUE* if the current values are within the band, otherwise *FALSE*.

- **Compare Result:** result of the current comparison. Indicates whether the current values are within the band or are smaller or larger than the band.
- **Current Compared Cycles:** number of cycles that have been compared.
- **Count Within Band:** counts how often the values were within the band.
- **Count Smaller:** counts how often the values were smaller than the band.
- **Count Bigger:** counts how often the values were larger than the band.
- **Value Number:** displays the value of the data point in the file that is currently being compared.

Standard HMI Controls

For the Time Based Envelope 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

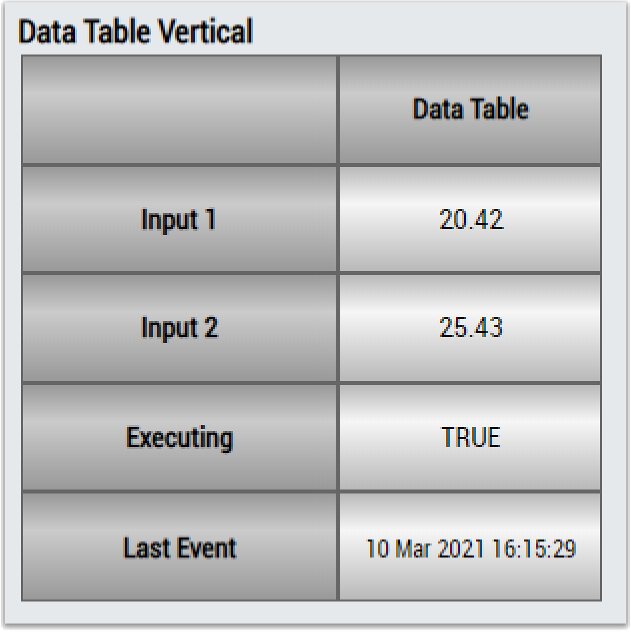
1. The TimeBasedEnvelope control visualizes the output values Executing Comparison, Lower Band, Upper Band, Within Band, Compare Result, Count Within Band, Count Smaller, Count Bigger, Current Compared Cycles, State and Value Number.



Read state: Pending idle

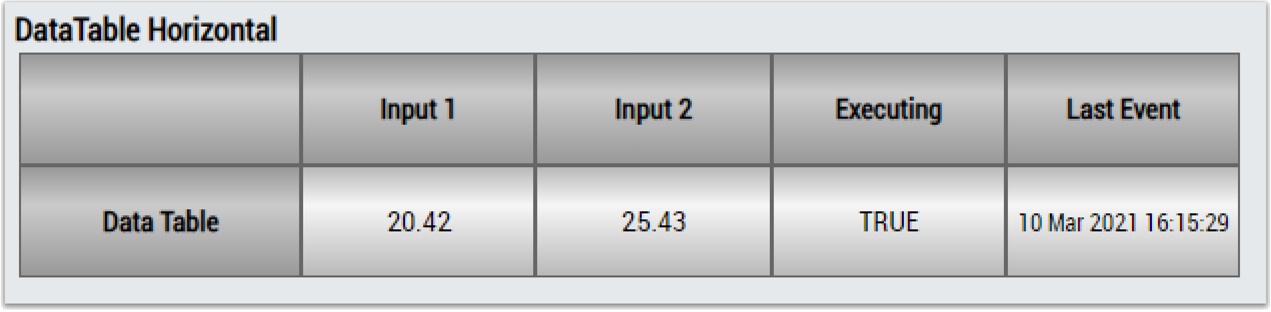
Current cycles: 6

2. The Table Control or Multivalue Control visualizes all output values: Executing Comparison, State, Lower Band, Upper Band, Within Band, Compare Result, Current Compared Cycles, Count Within Band, Count Smaller, Count Bigger, Value Number.



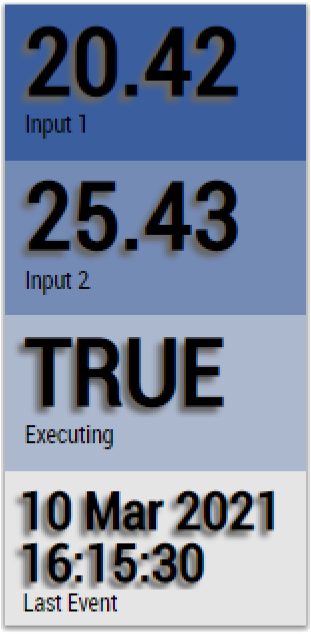
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

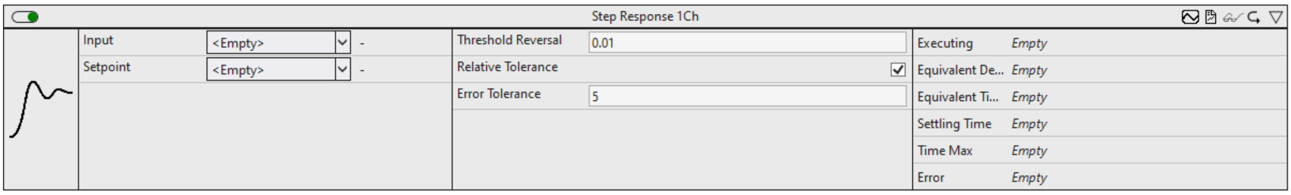
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Time Based Envelope algorithm using the Mapping Wizard [▶ 432].

6.4.2.11 Step Response 1Ch



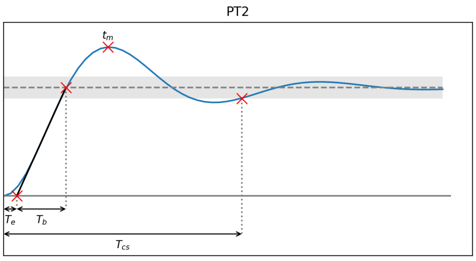
Step Response 1Ch

Input	<Empty>	-
Setpoint	<Empty>	-

Threshold Reversal	0.01
Relative Tolerance	☑
Error Tolerance	5

Executing	*Empty*
Equivalent De...	*Empty*
Equivalent Ti...	*Empty*
Settling Time	*Empty*
Time Max	*Empty*
Error	*Empty*

Step Response 1Ch identifies parameters of the step response of a PT2 track. These include the delay time T_e, the compensation time T_b, the settling time T_{cs}, and the time of maximum t_m.



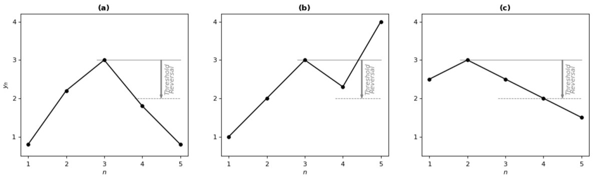
To detect whether the track is steady, a local minimum or maximum is searched for. If this is within the tolerance band (marked in gray), it is assumed that the track is steady. Only then is the settling time set.

The algorithm starts when a new setpoint is outside the previously stored tolerance band.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Threshold Reversal:** threshold for identifying reversals. Reversals are only detected if their difference from the next reversal exceeds the value of Threshold Reversal.
  Below are three examples of peak identification using the parameter *Threshold Reversal*.
  (a) The value y_3 is identified as a peak immediately after processing the value y_4 because the difference between y_3 and y_4 is greater than *Threshold Reversal*.
  (b) The value y_3 is not identified as a peak because the difference between y_3 and y_4 is smaller than *Threshold Reversal* and the curve starts rising again after y_4.
  (c) The value y_2 is identified as a peak after processing the value y_5 because the difference between y_2 and y_5 *exceeds Threshold Reversal*. The value y_2 cannot be identified as a peak beforehand because the difference between y_2 and y_3 (y_4) is less than/equal to *Threshold Reversal* and it is not known whether the values will continue to decrease.



- **Relative Tolerance:** Boolean flag. If this flag is *True*, the parameter *Error Tolerance* refers to the setpoint at the input as a percentage. Otherwise, an absolute tolerance band is taken into account.
- **Error Tolerance:** Specifies the size of the tolerance band in relation to the parameter *Relative Tolerance*. The tolerance band is updated when parameter identification is restarted.

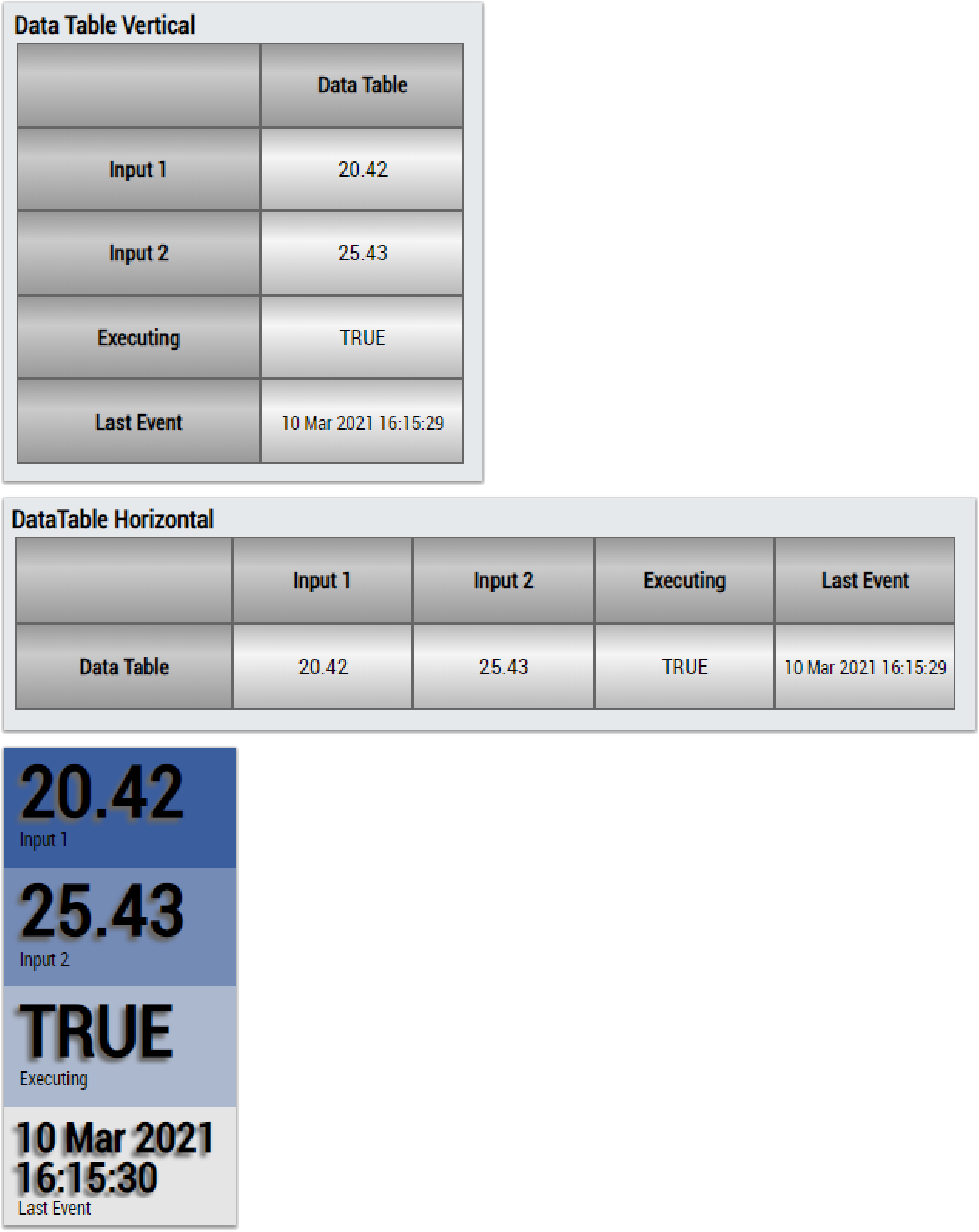
Output values

- **Executing:** True if parameter identification is active.
- **Equivalent Dead Time:** Delay time of the track. Timespan between the start of the step response and the intersection of the inflectional tangents with the start value.
- **Equivalent Time Constant:** Compensation time of the track. Timespan between the intersection point of the inflectional tangents with the start value and the intersection point of the inflectional tangents with the setpoint.
- **Settling Time:** Settling time of the track.
- **Time Max:** Time of the maximum overshoot.
- **Error:** Difference between input value and setpoint.

Standard HMI Controls

For the *Step Response 1Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control and Multivalue Control visualize all output values.



Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29

20.42
Input 1

25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

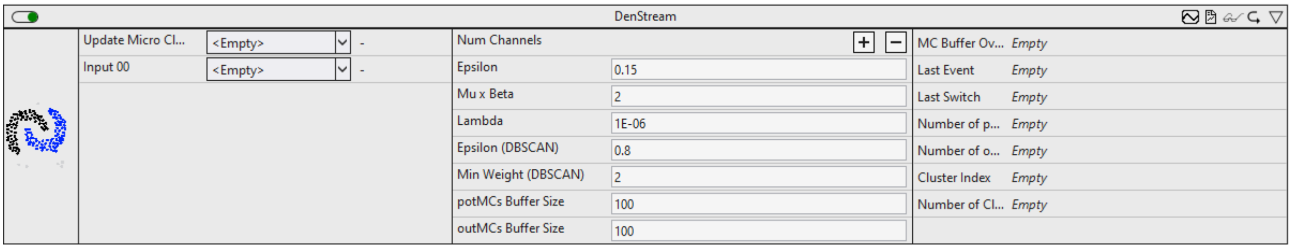
Fig. 2:

Alternatively, customer-specific HMI controls can be mapped using the Mapping Wizard [▸ 432].

6.4.3 Analytics - Clustering

The algorithms in the *Clustering* category provide functions for streaming data-based clustering of input data into various clusters that are not predefined. The clusters are detected based on the structures present in the input data. Examples of such cluster algorithms are *Sequential k-Means* and *DenStream*.

6.4.3.1 DenStream



DenStream

Update Micro Cl...	<Empty>	-
Input 00	<Empty>	-

Num Channels	
Epsilon	0.15
Mu x Beta	2
Lambda	1E-06
Epsilon (DBSCAN)	0.8
Min Weight (DBSCAN)	2
potMCs Buffer Size	100
outMCs Buffer Size	100

MC Buffer Ov...	*Empty*
Last Event	*Empty*
Last Switch	*Empty*
Number of p...	*Empty*
Number of o...	*Empty*
Cluster Index	*Empty*
Number of Cl...	*Empty*

DenStream is an implementation of the unmonitored, density-based clustering algorithm of the same name [1]. The latter is based on the well-known clustering algorithm DBSCAN [2, 3] and is particularly suitable for data streams whose structures change over time.

The number of input channels (referred to below as *n)* for this algorithm can be selected by the user. These inputs form the n-dimensional feature space in which clusters can be found. In each analysis cycle, the data stream provides the algorithm with a new feature vector that can be interpreted as a data point in this feature space. Clusters are separable areas with a high density of data points in the feature space.

In the first phase of the algorithm, the incoming data points are assigned to so-called micro-clusters (*MCs*). These micro clusters have properties (such as center point, weight and variance) that depend on the data points they contain. Only micro-clusters whose weight exceeds a certain threshold enter the second phase and are clustered by the DBSCAN algorithm. Thus, it is not necessary to retain the information about each data point. This reduces memory requirements, because over time there are far fewer micro-clusters than data points. Also, the computing effort for the DBSCAN algorithm is much lower since it runs over the reduced set of micro-clusters rather than all the data points. It is also possible to apply a fading function to the weights of the micro clusters. In this way, old data points lose their importance to the clustering process over time. This allows the algorithm to capture changes (such as the movement of clusters or their disappearance/appearance over time).

The DenStream algorithm has further advantages over other clustering algorithms. The user does not need to know the number of micro-clusters in advance, as the DenStream algorithm determines this number automatically. In addition, the algorithm is able to detect outliers in the data that does not belong to any cluster. Since it is a density-based algorithm, it is even possible to detect separate clusters of any shape (even if they are intertwined).

Parameter setting

Here we give a short introduction to how the algorithm works, mainly to give the reader a quick introduction to the parameter setting. For a deep understanding of the algorithm and its parameters, we refer the reader to the publications mentioned. Most of the terms and parameter names used here come directly from these publications.

The parameters of the DenStream algorithm mainly affect the following properties of the algorithm:

- the coarseness of the micro-clusters,
- the maximum distance between data points/micro-clusters so that they are assigned to the same cluster,
- the minimum density so that data points are identified as clusters and not as outliers,
- The fading rate at which older data points lose their significance.

The *Parameters Epsilon, Lambda* and *Mu x Beta* belong to the first phase of the algorithm, the formation of micro-clusters.

If possible, a data point is inserted into the micro-cluster whose center is closest to the data point. For this purpose, the Euclidean distances between the data point and the center points of all micro-clusters are compared and the micro-cluster with the smallest distance is selected. The data point can only be inserted into the micro-cluster if the radius of the micro-cluster does not exceed the *Epsilon* threshold after insertion. The radius is analogous to the variance of all data points contained in the micro-cluster. This means that data points can also be integrated into a micro-cluster even if their Euclidean distance to the center of the cluster is greater than *Epsilon,* as long as there are enough other points in the micro cluster with a smaller distance.

If the data point cannot be inserted into the nearest micro-cluster, a new micro-cluster is created with that data point. The weight of the respective micro-cluster is increased by one with the insertion of a data point.

In the left-hand plot in the illustration, the assignment of the data points to the micro-clusters is sketched for two input channels as an example. 20 data points assigned to four different micro-clusters are shown. The first micro-cluster (#1, marked in red) contains six data points, the second (#2, marked in green) also contains six, the third (#3, marked in blue) contains seven and the fourth (#1, marked in gray) contains only one data point. The area around the center of the micro-cluster in which a new data point would have to be located in order to be accepted into the respective micro-cluster with the specified epsilon (marked by dashed line) is colored. This sphere of influence is greater if the micro-cluster already contains several data points and they have a lower variance (see, for example, micro-clusters #1 and #2). In addition, the spheres of influence of multiple micro-clusters can overlap and mutually influence one another through their existence, see micro-cluster #2 (green) and #3 (blue). The data points are always assigned to the closer micro-cluster, for which reason the spheres of influence are separated by a straight line. In the plot, a data point can be seen that is assigned to the micro-cluster #2, but if the latter did not exist, then it would be assigned to micro-cluster #3.

Like in the original study [1], micro-clusters are divided into potential and outlier micro-clusters depending on their weight. Only potential micro-clusters are subsequently clustered by the DBSCAN algorithm. The data points in the outlier micro-clusters are marked as outliers. However, outlier micro-clusters are also stored and updated with new data points, as they can still develop into potential micro-clusters. The weight of a micro-cluster must exceed the *Beta x Mu* threshold in order to be counted as a potential micro-cluster. In the left-hand sketch in the illustration, for example, the micro-cluster #4 (gray) contains only one data point, thus has a weight of less than or equal to one and would be counted as an outlier micro-cluster for *Beta x Mu* = 1.

When a fading function is applied, the weight of the micro-clusters decreases over time. This fading rate is determined by the parameter *Lambda*. If the value is set to zero, no fading function is applied, otherwise the weights decrease by a factor of 2^(-*Lambda*) every second. If the weight of an outlier micro-cluster falls below an internal threshold (depending on *Mu x Beta* and *Lambda*), it is deleted from the memory.

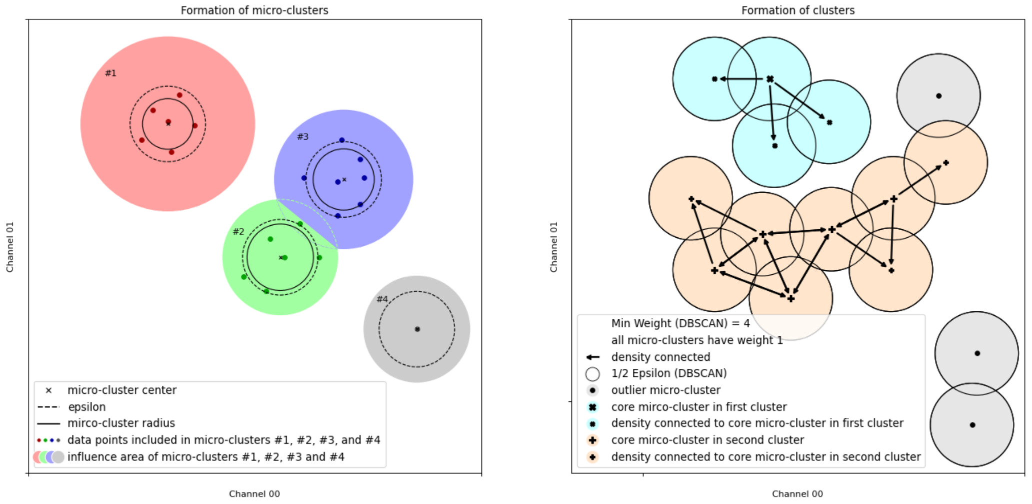
The parameters *Epsilon (DBSCAN)* and *Min Weight (DBSCAN)* affect the second phase. These parameters were adopted from the DBSCAN algorithm [3].

The DBSCAN algorithm runs over the set of potential micro-clusters and assigns them cluster designations. This can be either the index of the cluster to which they belong, or the designation outlier. The data point currently being processed is then assigned the name of the micro-cluster to which it belongs.

How does DBSCAN cluster the micro-clusters? The algorithm works according to the concept of density accessibility. Objects (in this case micro-clusters) belong to the same cluster if they are density-connected. This means that there must be a chain of micro-clusters with a maximum distance *Epsilon (DBSCAN)*. All micro-clusters forming this chain must satisfy a second condition. The sum of the weights of all micro-clusters within the distance *Epsilon (DBSCAN)* around each individual micro-cluster in this chain must exceed the threshold *Min Weight (DBSCAN)*. Micro-clusters that are not density-connected to at least one micro-cluster that meets this second condition are marked as outliers.

This is shown in the right-hand sketch in the illustration. For simplicity, it is assumed here that the weight of all micro-clusters is equal to 1. This corresponds to the case where there is exactly one data point in each micro-cluster and no fading function has been applied. The two clusters (marked with an x (turquoise) and a plus (orange)) with the two outlier micro-clusters result when the *Min Weight (DBSCAN)* parameter is set to four. The micro-clusters marked with a capital "x" or "+" are core micro-clusters. This means that at least three more micro-clusters (plus the micro cluster considered = 4) have a maximum distance of *Epsilon (DBSCAN)* to these micro-clusters. The micro-clusters marked with a small "x" or "+" are not core micro-clusters, but are located in the *Epsilon (DBSCAN)* neighborhood of a core micro-cluster and therefore belong to the same cluster. The micro-cluster in the upper right corner, marked with a small dot, is an outlier micro-cluster. Although it is located in the *Epsilon (DBSCAN)* neighborhood of a micro-cluster that is counted as belonging to a cluster, it is not a core micro-cluster.

Likewise, the two micro-clusters at the bottom right are outliers. Although they are in the immediate *Epsilon (DBSCAN)* neighborhood, there are only two of them. The *Min Weight (DBSCAN)* threshold of the weights is not exceeded.



The parameters *outMCs Buffer Size* and *potMCs Buffer Size* are specific to this implementation of the algorithm and are required because the memory for outliers and potential micro-clusters must be allocated before execution. Thus, *outMCs Buffer Size* and *potMCs Buffer Size* limit the possible number of outliers and potential micro-clusters during runtime. The user must find values for these parameters so that this limit is not exceeded.

The maximum number of outliers and potential micro-clusters during the execution of the algorithm depends on the distribution of the input data, but also on the setting of the other parameters. There are fewer micro-clusters at higher values of *Epsilon* as this results in coarser micro-clusters that can contain data points from a wider range. In general, the number of outlier micro-clusters increases at the beginning of the analysis, but decreases again when outlier micro-clusters transform into potential micro-clusters. If the patterns in the data stream do not change over time, the number of micro-clusters settles after an initial phase.

The more micro-clusters there are, the higher the computing requirements. For all outliers and potential micro-clusters we compare the distance to a data point and then all potential micro-clusters must be included in the calculation of the DBSCAN algorithm. A compromise must therefore be reached between the computing speed and the coarseness of the micro-clusters.

What happens if the values of *outMCs Buffer Size* and *potMCs Buffer Size* are set too low and at some point during the analysis more micro-clusters are required to capture the input data points? In this case, the algorithm continues to assign the data points to the existing micro-clusters and marks the data points accordingly, but the existing micro-clusters are no longer updated to prevent the buffer from overflowing. This means that the clustering of the data points continues, but with an overall stagnated feature space (older set of micro-clusters). Changes in the pattern of the data stream could then no longer be detected.

[1] F. Cao, M. Ester, W. Qian, A. Zhou. Density-Based Clustering over an Evolving Data Stream with Noise. In Proceedings of the 2006 SIAM International Conference on Data Mining, S. 326-337. SIAM.

[2] M. Ester, H.-P. Kriegel, J. Sander and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of KDD*, 1996.

[3] J. Sander, M. Ester, H.-P. Kriegel, X. Xu. Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications. Data Mining and Knowledge Discovery 2, 169-194 (1998)

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Input values

- **Update Micro Cluster:** If TRUE, micro-clusters are updated by the incoming data. If FALSE, the existing micro-clusters remain unchanged and are only used to determine the cluster index of the incoming data points.
- **Input 01, ..., Input 0n:** These inputs form the feature space for which clustering is performed.

Configuration options

- **Num Channels:** Here you can modify the number of input channels.
- **Epsilon:** Threshold for the maximum radius of micro-clusters.
- **Mu x Beta:** Threshold for the weight of a micro-cluster between outlier and potential micro-cluster.
- **Lambda:** Specifies the fading rate of the algorithm. The weight of each data point decreases by a factor of 2^(-*Lambda*) every second.
- **Epsilon (DBSCAN):** Specifies the epsilon parameter of the DBSCAN algorithm.
- **Min Weight (DBSCAN):** Threshold for the sum of weights in the epsilon neighborhood of a micro-cluster for the DBSCAN algorithm.
- **potMCs Buffer Size:** Maximum number of potential micro-clusters. The memory is allocated to *potMCs Buffer Size* micro-clusters.
- **outMCs Buffer Size:** Maximum number of outlier micro-clusters. The memory is allocated to *outMCs Buffer Size* micro-clusters.

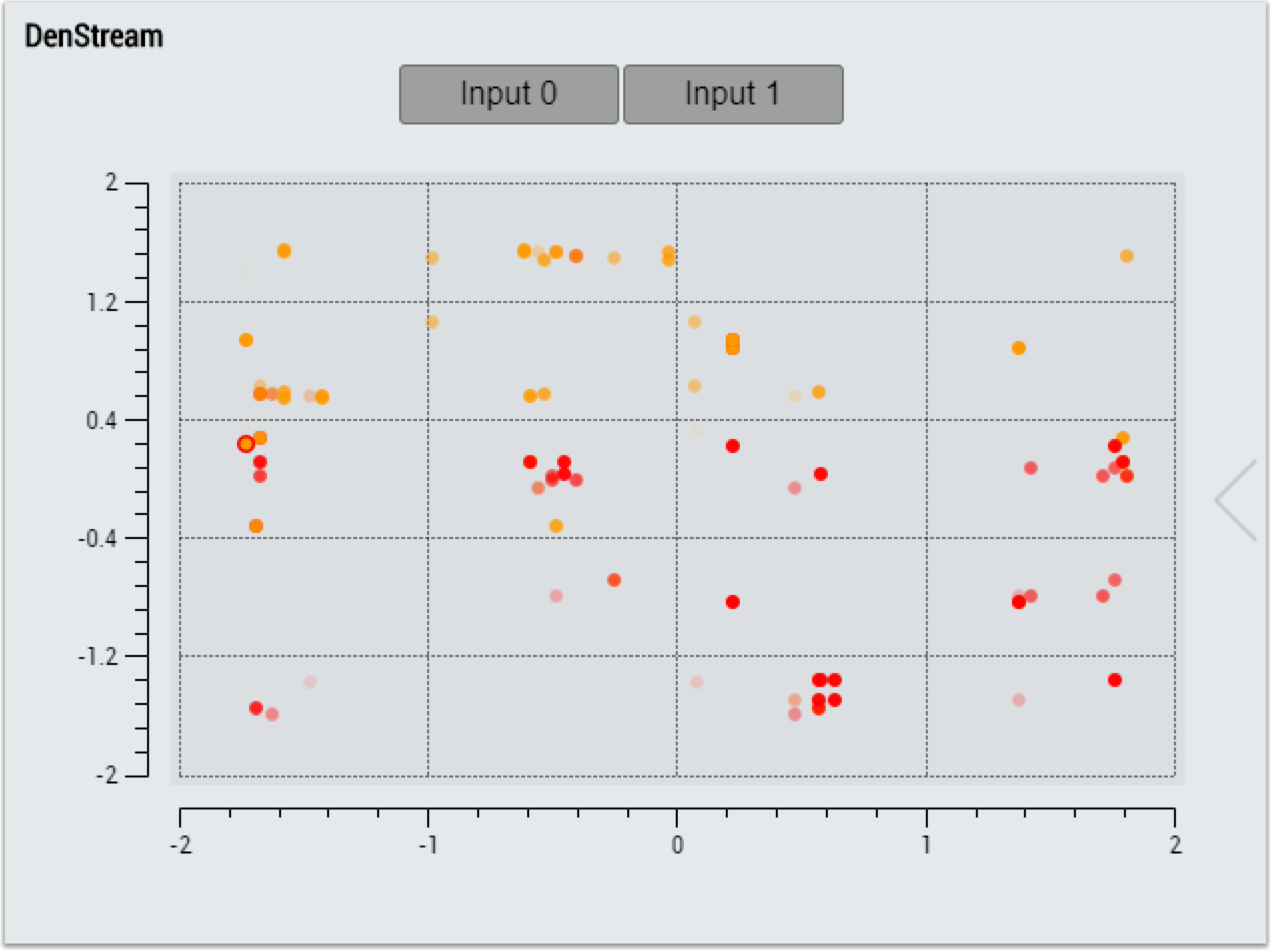
Output values

- **New Result:** Is TRUE if the new cluster index differs from the cluster index of the last cycle.
- **MC Buffer Overflow:** TRUE if the micro-cluster update is stopped to prevent overflow of *potMCs* or *outMCs Buffer Size*.
- **Last Event:** This is the timestamp of the last cycle with a change of the cluster index
- **Last Switch:** This is the timestamp of the last cycle with an alternation between updating and not updating micro-clusters (either by setting the input *Update Micro Cluster* to TRUE or by internally preventing an overflow of *potMCs/outMCs Buffer Size*).
- **Number of potMCs:** Indicates the number of potential micro-clusters currently present.
- **Number of outMCs:** Indicates the number of outlier micro-clusters currently present.
- **Cluster Index:** Specifies the cluster index that the DBSCAN algorithm outputs for the data point of the current cycle.
- **Number of Clusters:** Specifies the total number of clusters detected by the DBSCAN algorithm.

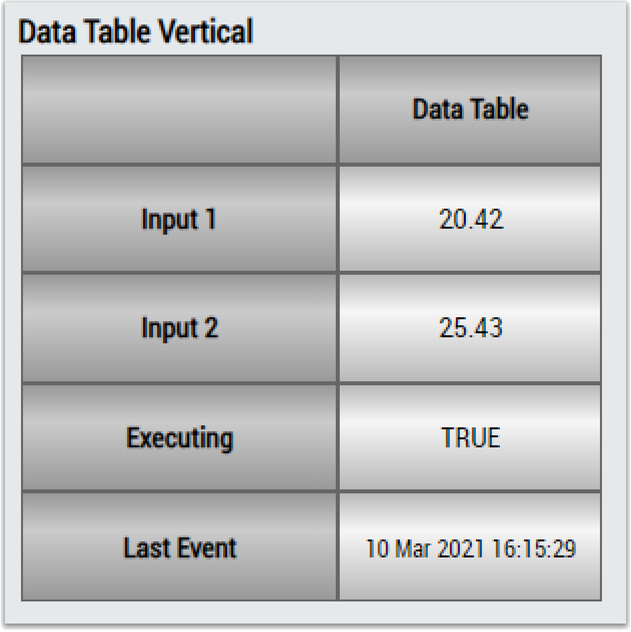
Standard HMI Controls

For the DenStream algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The DenStream control visualizes the inputs in the chart and their respective classification in a cluster (cluster index) in color. The buttons can be used to select the inputs to be displayed, up to two at a time. The slider on the right lists all output values: cluster count, number of potential micro-clusters, number of outlier micro-clusters, cluster index, last event, last switch.

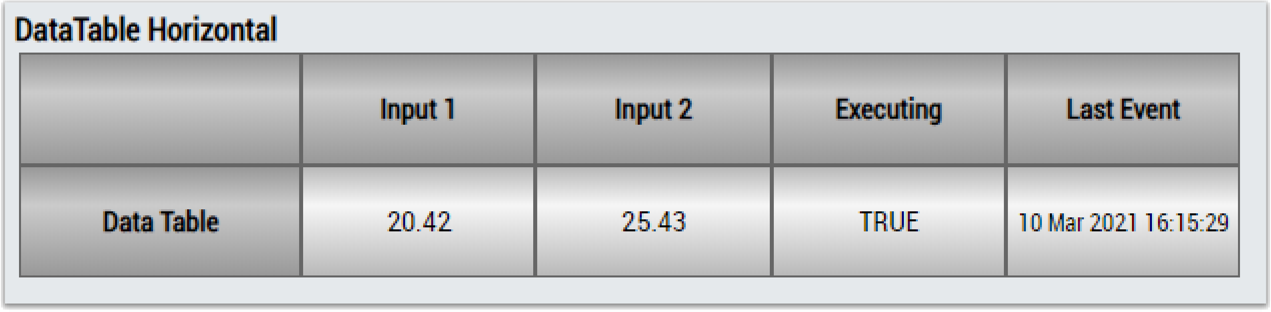


2. The Table Control or Multivalue Control visualizes all output values: cluster count, number of potential micro-clusters, number of outlier micro-clusters, cluster index, last event, last switch.



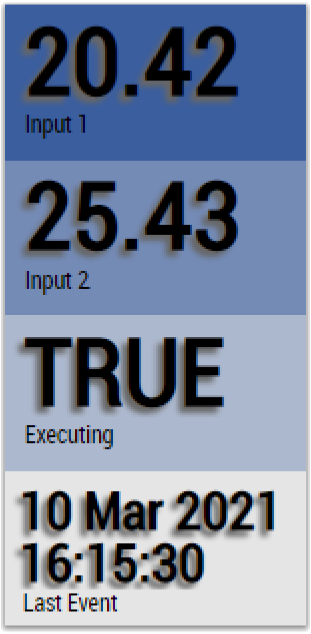
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

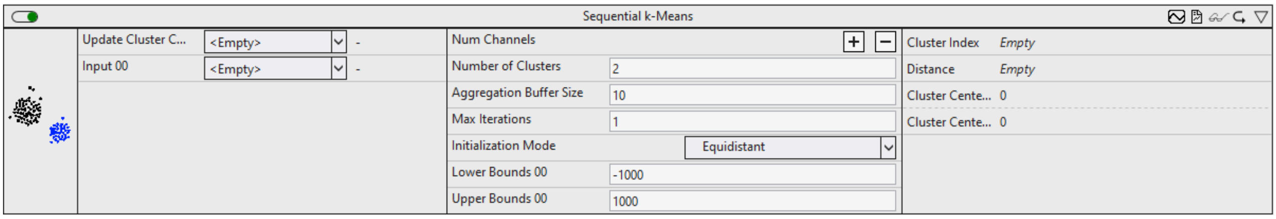
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the DenStream algorithm using the Mapping Wizard [▶ 432].

6.4.3.2 Sequential k-Means



Sequential k-Means

Update Cluster C...	<Empty>	-	
Input 00	<Empty>	-	

Num Channels	+ −
Number of Clusters	2
Aggregation Buffer Size	10
Max Iterations	1
Initialization Mode	Equidistant
Lower Bounds 00	-1000
Upper Bounds 00	1000

Cluster Index	*Empty*
Distance	*Empty*
Cluster Cente...	0
Cluster Cente...	0

The *Sequential k-Means* algorithm is an implementation of the unmonitored clustering algorithm of the same name. It is a sequential variant of the widely used k-Means clustering algorithm for streaming data. The aim of the algorithm is to find clusters based on the structure of the data, each of which contains similar data points and separates different data points from each other.

The number of input channels (referred to below as *n)* for this algorithm can be freely selected by the user. These inputs span the n-dimensional feature space in which the clusters are found. In each analysis cycle, the data stream provides the algorithm with a new feature vector that can be interpreted as a data point in this feature space. Data points that are close to each other in this feature space are assigned to the same cluster. The number of clusters present must be set by the user before the analysis begins and remains fixed.

In contrast to the k-Means algorithm for conventional batch analysis, the data for the *Sequential k-Means* are not fully available at the time of analysis. Instead, the data points arrive one by one in the form of streaming data. They are therefore processed sequentially and assigned to the corresponding cluster closest to them. This approach results in a number of differences, two of which are particularly relevant to the use of the algorithm as well as the parameter settings.

On the one hand, all data points and thus the value ranges of the individual features are already available at the beginning of a batch analysis, whereas this is not the case with sequential analysis, so that the value ranges are not necessarily fixed in advance. However, it is helpful to know the value ranges of the input channels in advance, even if the actual values only arrive during the course of the analysis. This is particularly important for the initialization of cluster centers. Three different approaches are available for initialization. The center points can be specified in the form of specific *values* via a parameter array. Alternatively, the center points can be set *randomly* or *equidistantly* in a defined range of values. For the initialization modes *Random* and *Equidistant* the value ranges are required and have to be set via the parameters *Lower Bounds* and *Upper Bounds* for the individual input channels.

On the other hand, in a batch analysis all data points are typically traversed multiple times to update the cluster centers until they change only minimally. This is not possible within the framework of the sequential analysis. However, in order to still be able to adjust the cluster centers and traverse data points multiple times, the algorithm *Sequential k-Means* has a buffering mechanism referred to as *Aggregation Buffer*, which makes it possible to store a limited number of values temporarily. When filling the buffer, all incoming data points are assigned to the closest cluster. The distance between a data point and the cluster centers is determined by the Euclidean norm. Only when the buffer is filled are the cluster centers updated based on the newly allocated data points in the buffer. The new cluster center corresponds to the mean value of all data points contained in the cluster. This can be calculated incrementally, so that the old data points are not needed for the calculation. The size of the buffer is set by the parameter *Aggregation Buffer Size;* the default value is 10. The parameter *Max Iterations* can be used to specify the number of iterations through the buffer. The default value is one. If the value is set to two, for example, after the first adjustment of the cluster centers the data points in the buffer are reassigned to the clusters and then the cluster centers are adjusted again. Due to the shift in cluster centers, it is possible for individual data points to be assigned to different clusters from one iteration to the next. Due to the limited computing capacity for data processing between two cycles, excessively high values should be avoided for the parameters *Aggregation Buffer Size* and *Max Iterations*, otherwise the update of the cluster centers may not be guaranteed. If the cluster centers are not updated for large values for these parameters but are updated for smaller parameter values, this is an indication that the computing capacity is insufficient for the set parameter values and smaller values should be selected.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Input values

- **Update Cluster Centers:** If TRUE, the centers of each cluster are updated by the incoming data. If FALSE, the cluster centers remain unchanged and are only used to determine the cluster index of the incoming data points.
- **Input 01, ..., Input 0n:** These inputs form the n-dimensional feature space for which clustering is performed.

Configuration options

- **Num Channels:** Determines the number of input channels.
- **Number of Clusters:** Defines the number of clusters.
- **Aggregation Buffer Size:** Specifies the size of the aggregation buffer and thus the number of cycles after which the cluster centers are updated. The input values of these cycles are stored internally (in the aggregation buffer). The default value for this parameter is 10.
- **Max Iterations:** Specifies how often to iterate over the values in the aggregation buffer. The default value for this parameter is 1.
- **Initialization Mode:** Specifies the way in which the cluster centers are initialized:
  - **Random:** The cluster centers are set randomly within the limits set by the *Lower Bounds* and *Upper Bounds*.
  - **Equidistant:** The cluster centers are distributed equidistantly in the range of values defined by the *Lower Bounds* and *Upper Bounds*.
  - **Values:** The cluster centers are initialized with the values set by the array *Initial Cluster Centers*.
- **Initial Cluster Centers:** For the *initialization mode Values* the values for the initial cluster centers are set here. The values for the individual clusters are set line by line. That is, the number of matrix rows corresponds to the *Number of Clusters* and the number of matrix columns corresponds to the *Number of Channels*. The first row contains the values for the first cluster for each input channel, and so on.

- **Lower Bounds:** For the modes *Random* and *Equidistant* the lower limits for the individual input channels are set.
- **Upper Bound:** For the modes *Random* and *Equidistant* the upper limits for the individual input channels are set.

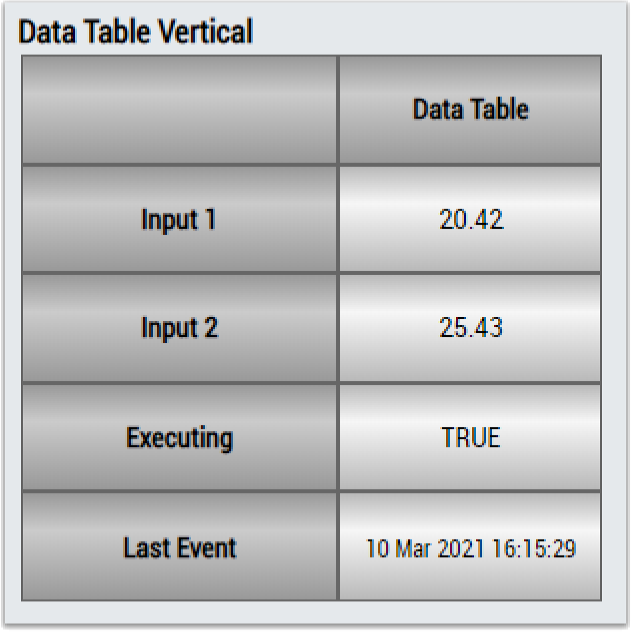
Output values

- **Cluster Index:** Specifies the cluster index assigned to the data point of the last cycle, indicating the corresponding assigned cluster.
- **Distance:** Specifies the Euclidean distance between the data point and the assigned cluster center.
- **Cluster Centers:** Outputs the cluster centers of all clusters line by line. This corresponds to a matrix of dimension *Number of Clusters* x *Number of Channels*.

Standard HMI Controls

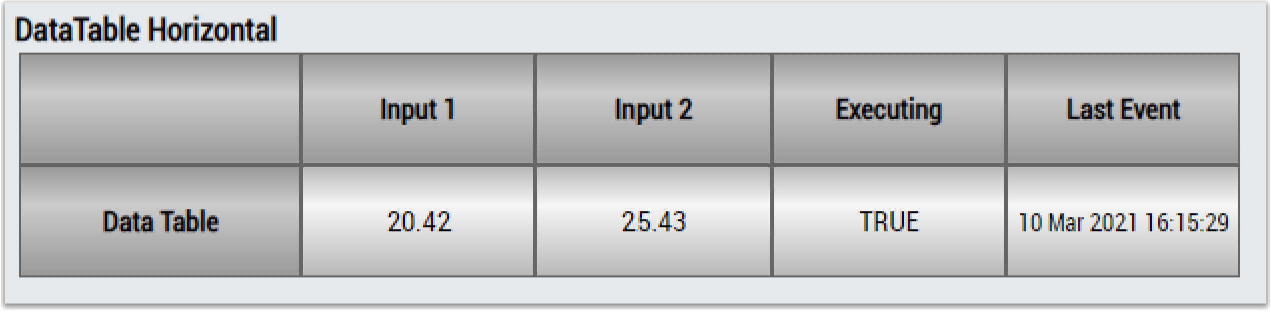
For the *Sequential k-Means* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes the output values: Lower Bounds, Upper Bounds and Initial Centers.



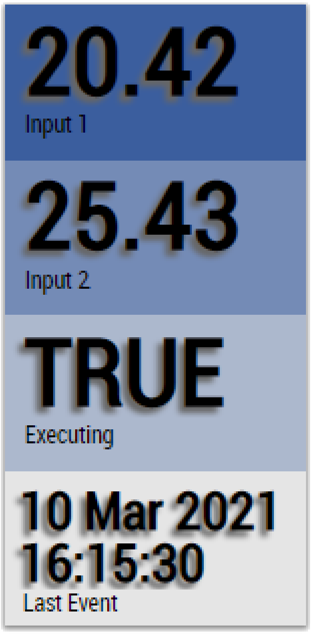
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

TRUE
Executing

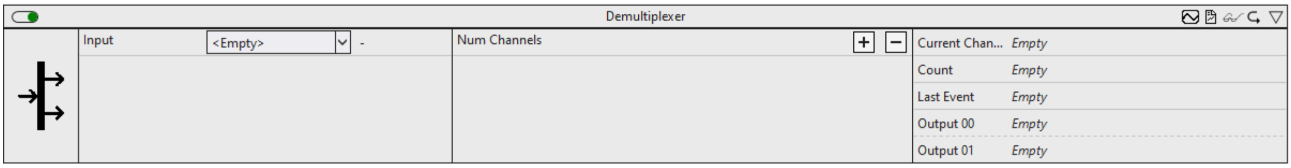
10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Sequential k-Means* algorithm using the Mapping Wizard [▶ 432].

6.4.4 Analytics - Compare

The algorithms of the category *Analytics-Compare* provide functionalities for comparative analysis and logic operations.

6.4.4.1 Demultiplexer



Demultiplexer

Input <Empty> -

Num Channels

Current Chan...	*Empty*
Count	*Empty*
Last Event	*Empty*
Output 00	*Empty*
Output 01	*Empty*

The demultiplexer selects an output channel based on the input value. For this purpose, the input value is interpreted as an integer. This value corresponds to the output channel. If the value is outside the configured number of channels, the output channel is set to 0.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **NumChannels:** Adds or removes an output channel.

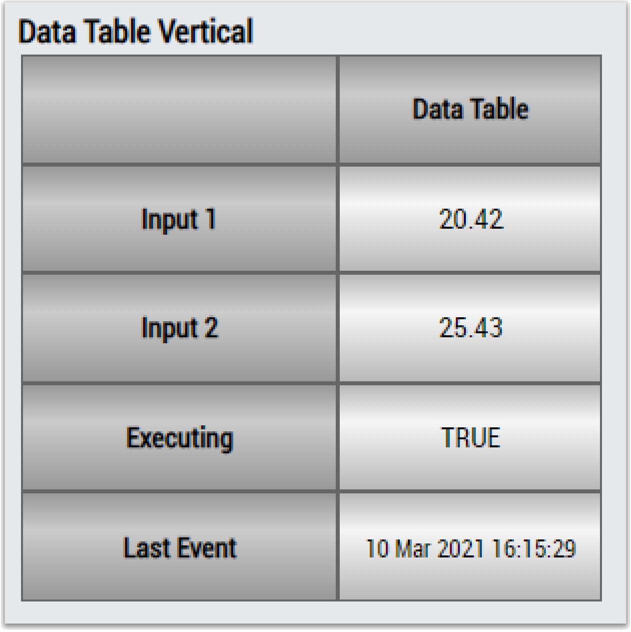
Output values

- **Current Channel:** Indicates the number of the selected channel. The value is 0 if the selected channel is outside the configured channels.
- **Count:** Starts with 1 for the channel selected at the start of the analysis and increments each time another channel is selected.
- **Last Event:** Timestamp of the last channel change.
- **Out 00..n:** Boolean output for channels 0 (selected channel < 1 or >n) to n

Standard HMI Controls

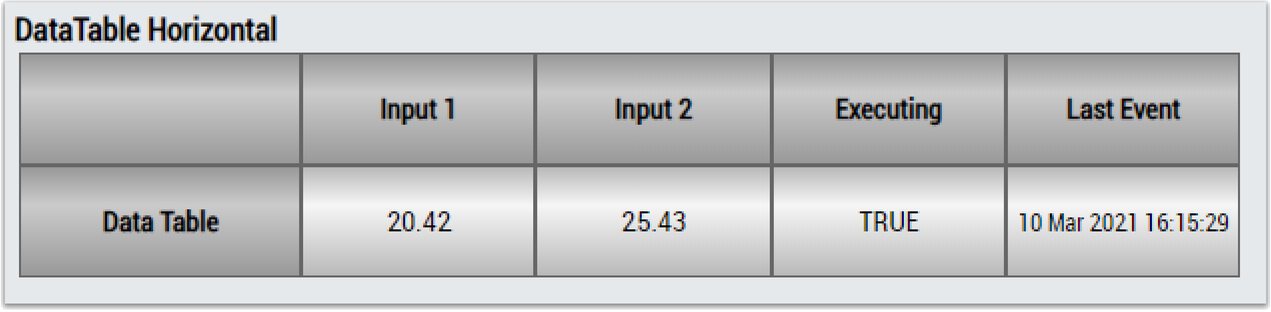
For the Demultiplexer algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes the output values Count, Current Channel and Last Event and Out.



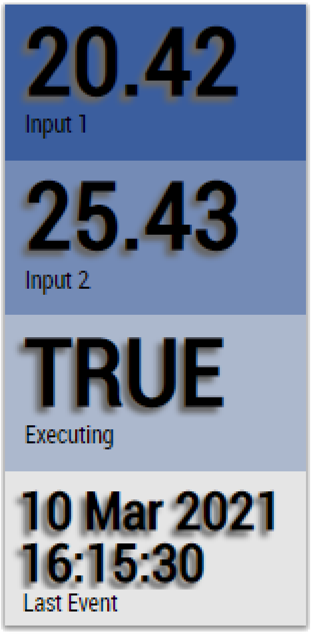
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

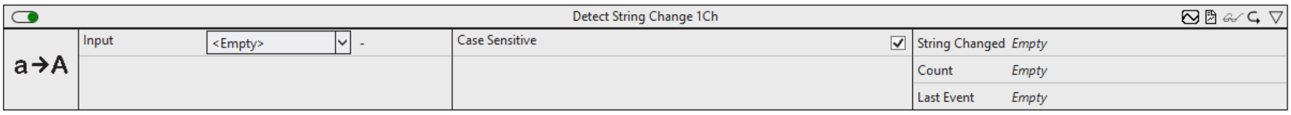
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Demultiplexer algorithm using the Mapping Wizard [▶ 432].

6.4.4.2 Detect String Change 1Ch



Detect String Change 1Ch

a→A

Input <Empty> -

Case Sensitive ☑

String Changed *Empty*

Count *Empty*

Last Event *Empty*

The *Detect String Change 1Ch* detects and counts changes of string values. Therefore, case sensitivity can be taken into account or not.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Case Sensitivity:** If the checkbox is ticked off, case sensitivity is taken into account, otherwise not.

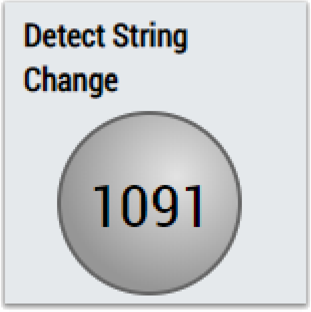
Output values

- **String Changed:** *TRUE*, if a string change was detected, otherwise *FALSE*.
- **Count:** Count up every time Boolean Switch is *TRUE*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

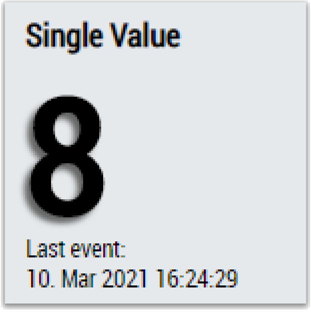
Standard HMI Controls

For the Detect String Change 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

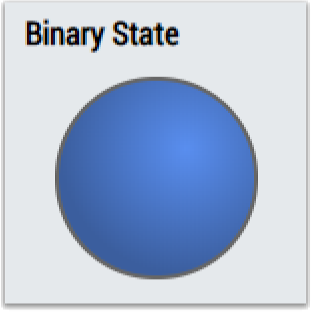
1. The Comparison control visualizes the output values Boolean Switch and Count.



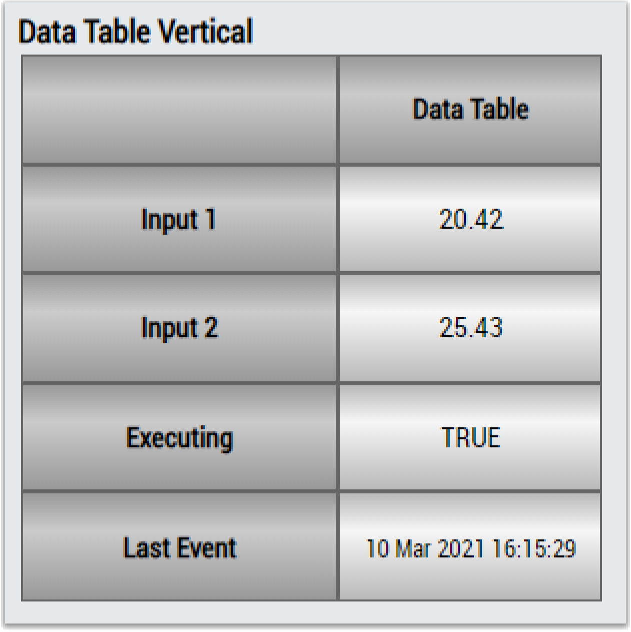
2. The SingleValue control visualizes the output values Count and Last Event.



3. The BinaryState control visualizes the output value Boolean Switch.

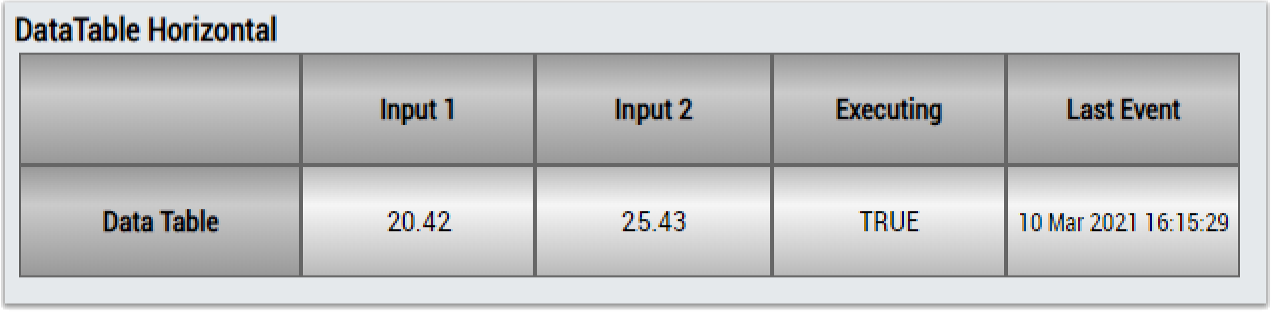


4. The Table Control or Multivalue Control visualizes all output values: Operation Out, Count, Last Event.



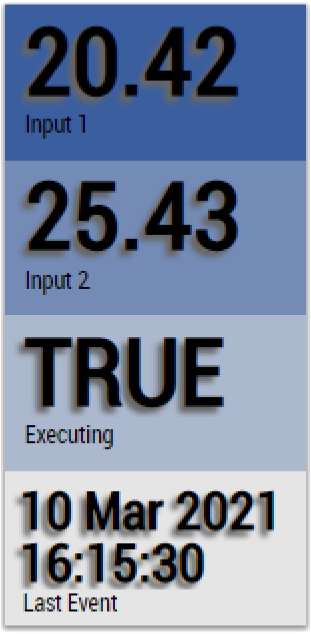
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

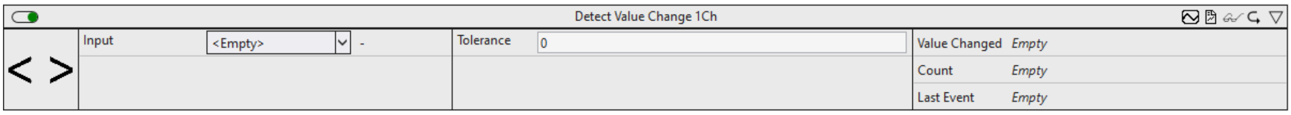
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Detect String Change 1Ch algorithm using the Mapping Wizard [▸ 432].

6.4.4.3 Detect Value Change 1Ch



Detect Value Change 1Ch

Input <Empty> -

Tolerance 0

Value Changed Empty
Count Empty
Last Event Empty

The *Detect Value Change 1Ch* detects and counts changes in numeric input values.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Tolerance:** The tolerance refers to the input value of the last detected value change. If the input value is outside this tolerance, a value change is detected.

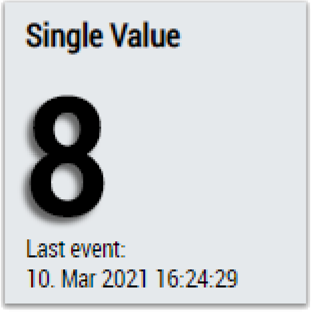
Output values

- **Value Changed:** *TRUE*, if a value change was detected, otherwise *FALSE*.
- **Count:** Count up every time Boolean Switch is *TRUE*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

Standard HMI Controls

For the *Detect Value Change 1Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue control visualizes the output values Count and Last Event.

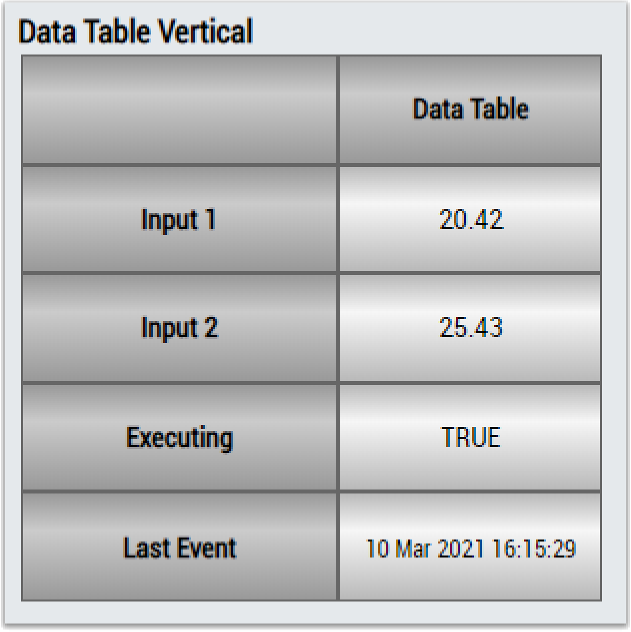


Single Value

8

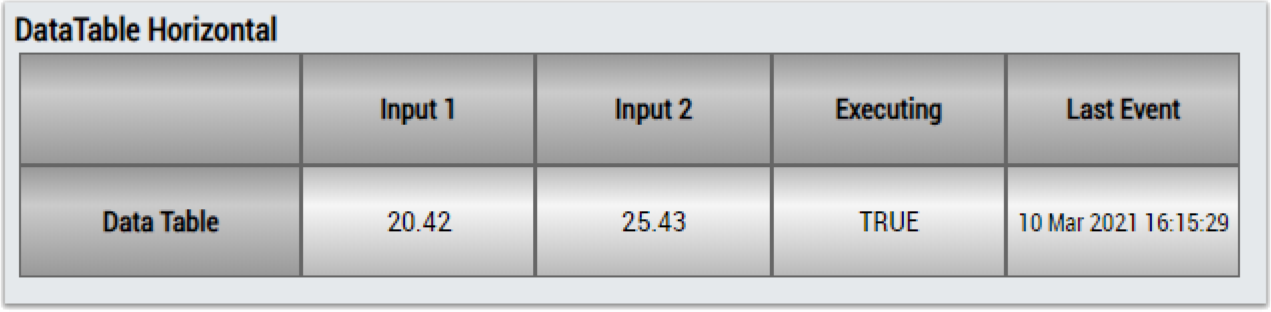
Last event:
10. Mar 2021 16:24:29

2. The Table Control and Multivalue Control visualize all output values.



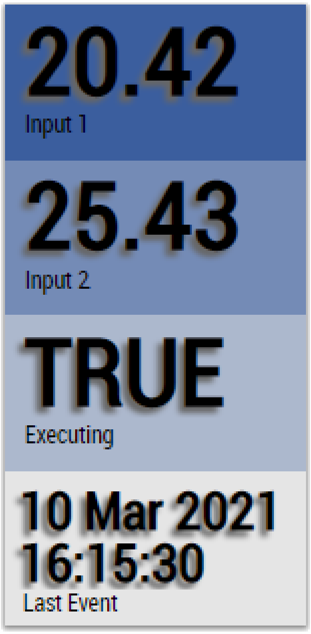
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

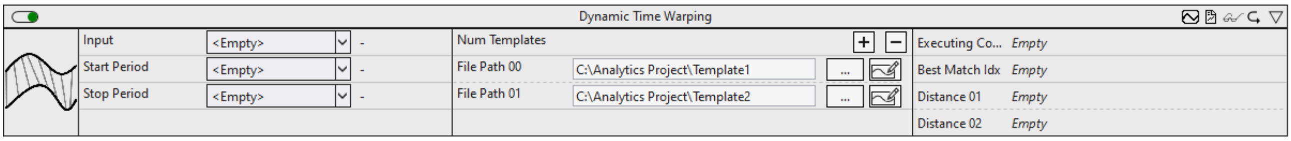
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped using the Mapping Wizard [▶ 432].

6.4.4.4 Dynamic Time Warping



Dynamic Time Warping

Input	<Empty>	Num Templates		Executing Co...	Empty
Start Period	<Empty>	File Path 00	C:\Analytics Project\Template1	Best Match Idx	Empty
Stop Period	<Empty>	File Path 01	C:\Analytics Project\Template2	Distance 01	Empty
				Distance 02	Empty

The *Dynamic Time Warping* algorithm compares input data with previously recorded templates. The special feature of the algorithm is that signals with different speeds or even shifted signals can be compared. As a result, the distance between the input signal and the respective template is output. The smaller the distance, the more equal the compared signals are. If the distance is 0, both signals are identical. The amount of distance depends on the equality but also on the length of the signals.

The comparison starts when the signal of the Start Period flag is *TRUE*. A result is output if the signal of the Stop Period flag is TRUE or the Start Period flag is TRUE again.

It is recommended not to use *Dynamic time Warping* simultaneously with Time Based Teach Path 1Ch [▶ 251] due to concurrent file access. Instead, a reference signal should first be taught in with the Time Based Teach Path 1Ch [▶ 251] and only then should the evaluation be carried out with the aid of the *Dynamic time Warping*. The templates contain reference signals previously recorded with the Time Based Teach Path 1Ch [▶ 251]. As a rule, the templates are a few hundred supporting points. Therefore, a reduction of the data with the function block Downsampling 1Ch [▶ 85] is often useful.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Templates:** adds or removes a template for comparison.
- **File Path 01 ... n:** path to the previously recorded template.

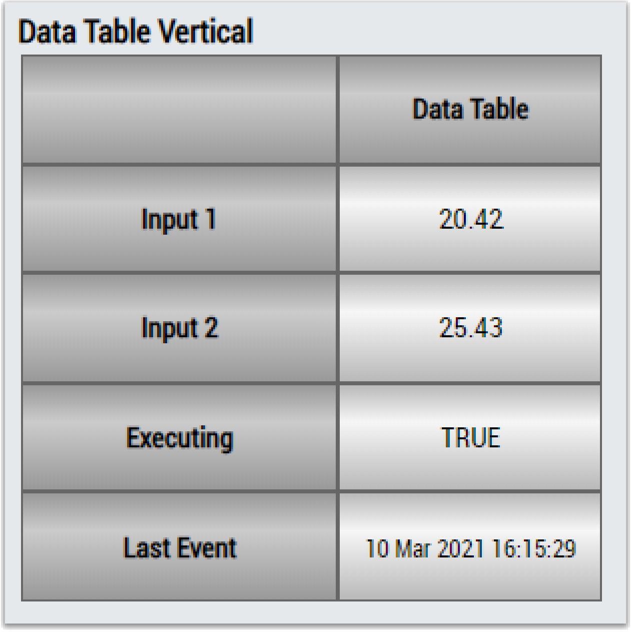
Output values

- **Executing Compare:** displays *TRUE* if the algorithm is processing the incoming values, otherwise *FALSE*. The processing begins when the Start Period flag is *TRUE* and ends when Stop Period is TRUE.
- **Best Match Idx:** outputs the index of the template with the smallest distance to the input channel.
- **Distance 01 ... n:** specifies the distance between the input signal and the respective template. The smaller the distance, the more equal the compared signals are.

Standard HMI Controls

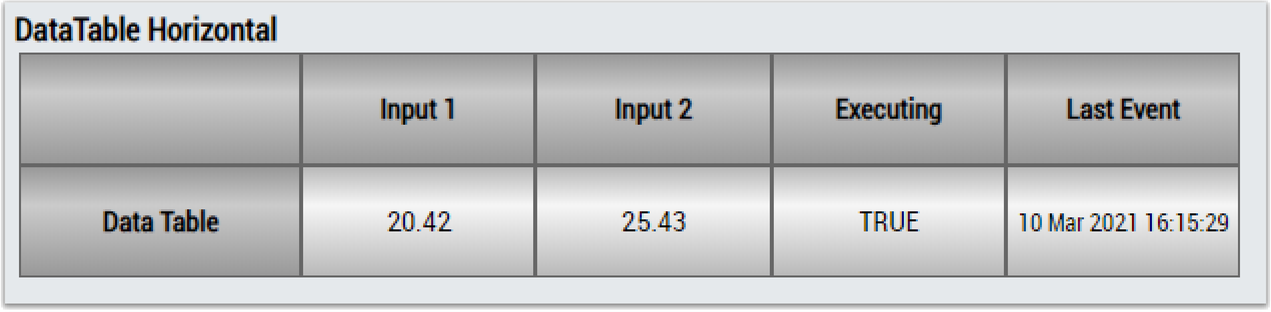
For the *Dynamic Time Warping* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control and Multivalue Control visualize all output values: *Executing Compare*, *Best Match Idx*, *Distance*.



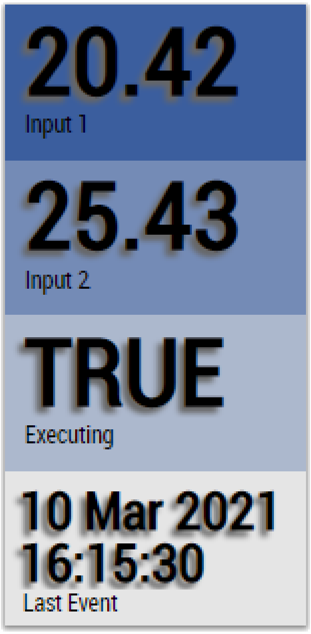
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

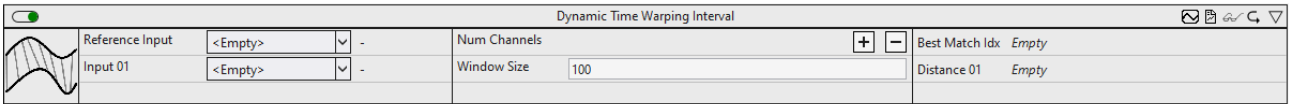
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Dynamic Time Warping* algorithm using the Mapping Wizard [▸ 432].

6.4.4.5 Dynamic Time Warping Interval



Dynamic Time Warping Interval

Reference Input <Empty> -

Input 01 <Empty> -

Num Channels

Window Size 100

Best Match Idx *Empty*

Distance 01 *Empty*

The *Dynamic Time Warping Interval* algorithm compares several input data with each other. The special feature of the algorithm is that signals with different speeds or even shifted signals can be compared. Only the signal interval of a configured window is considered for the comparison. New results are output after the window expires. As a result, the distance between the reference signal and the respective input signal is output. The smaller the distance, the more equal the compared signals are. If the distance is 0, both signals are identical. The amount of distance depends on the equality but also on the length of the signals.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** adds or removes an input channel.
- **Window Size:**specifies the number of cycles over which a calculation is made. The memory requirement of the algorithm is proportional to this parameter.

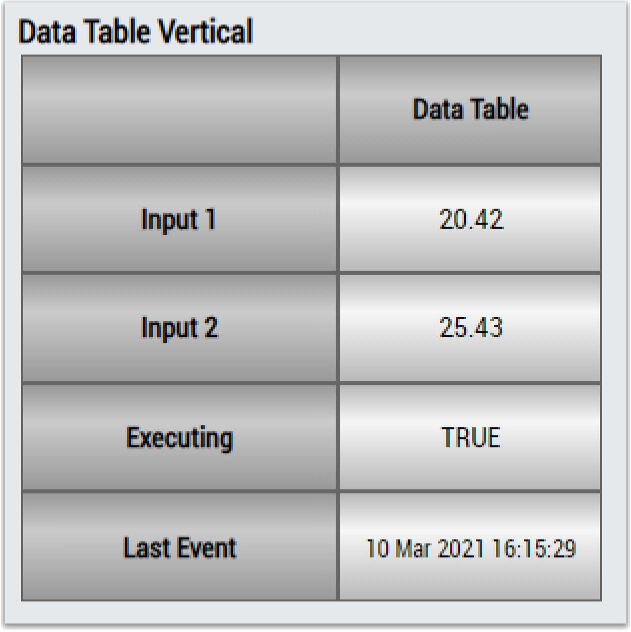
Output values

- **Best Match Idx:** outputs the index of the input channel with the smallest distance to the reference channel.
- **Distance 01 ... n:** specifies the distance between the reference signal and the respective input channel. The smaller the distance, the more equal the compared signals are.

Standard HMI Controls

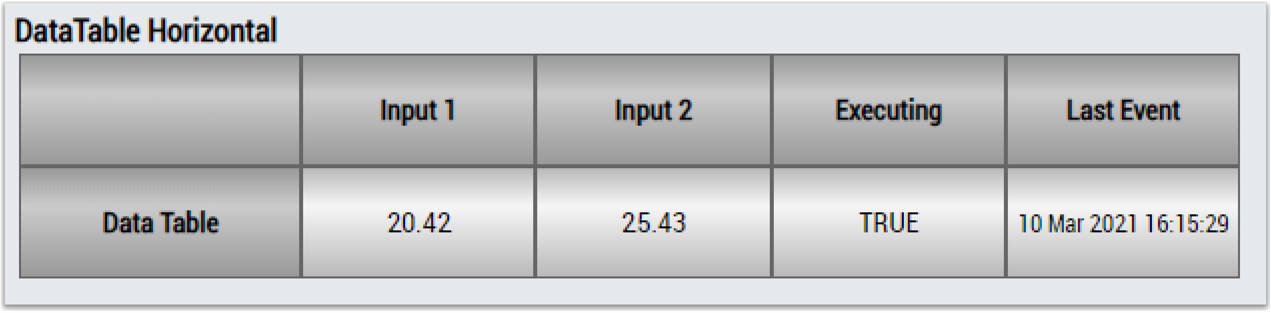
For the *Dynamic Time Warping Interval* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control and Multivalue Control visualize all output values: *Best Match Idx*, *Distance*.



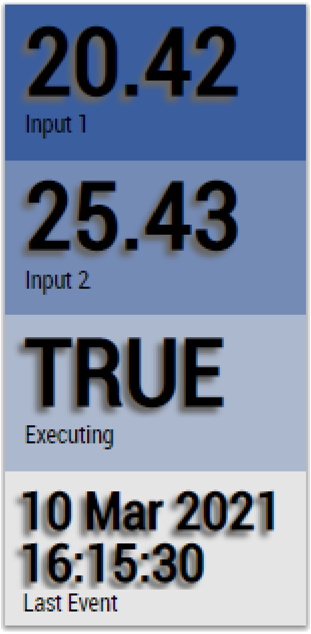
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

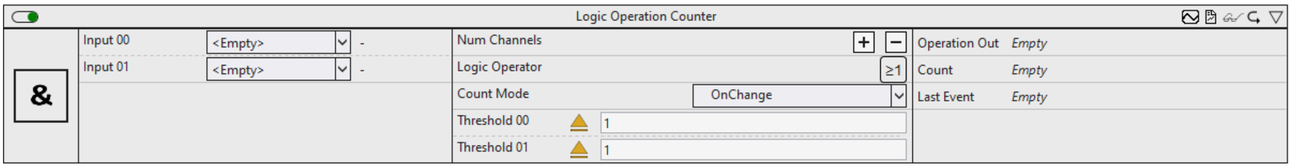
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Dynamic Time Warping Interval* algorithm using the Mapping Wizard [▶ 432].

6.4.4.6 Logic Operation Counter



Logic Operation Counter

&

Input 00	<Empty>	-
Input 01	<Empty>	-

Num Channels	+ −
Logic Operator	≥1
Count Mode	OnChange
Threshold 00	1
Threshold 01	1

Operation Out	*Empty*
Count	*Empty*
Last Event	*Empty*

The *Logic Operation Counter* executes a logical operation on the values of two or more channels and provides the result of this logical operation. Therefore, each input value can be combined with a threshold and an operator. Furthermore, the logic operator and the count mode can be configured individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** adds or removes an input channel.
- **Threshold 00:** threshold for the signal of the first channel.
- **Threshold 01 ..n:** threshold for the signal of the second to nth channel.
- **Logic Operator:** logical operator for the operation:

≥1 Logical OR

=1 Logical XOR (EXCLUSIVE OR)

& Logical AND

$\overline{\&}$ Logical NAND (NOT AND)

$\overline{\geq 1}$ Logical NOR (NOT OR)

- **Count Mode:** mode of the result counter.
  OnChange: the counter counts every time the result changes to TRUE.
  Cyclic: the counter increments every cycle when the condition is *TRUE*.
- **Use Absolute Values:** if the checkbox is checked, the absolute values of the input signal are used.
- **Tolerance (optional):** tolerance value for the Equal / NotEqual comparisons.

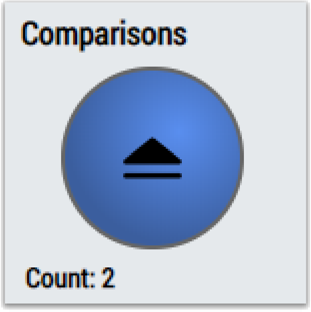
Output values

- **Operation Out:** Result of the logical operation.
- **Count:** Incremented when the output value is TRUE. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

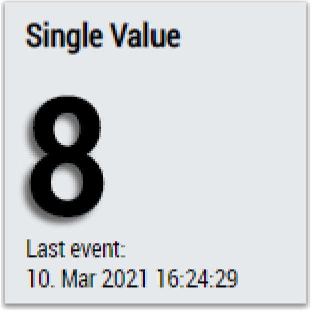
Standard HMI Controls

For the Logic Operation Counter algorithm, the following HMI controls are available for generating an Analytics Dashboard:

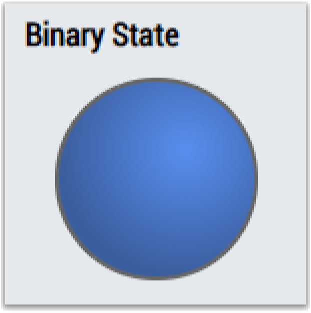
1. The Comparison control visualizes the output values Operation Out and Count as well as the configuration option Logic Operator.



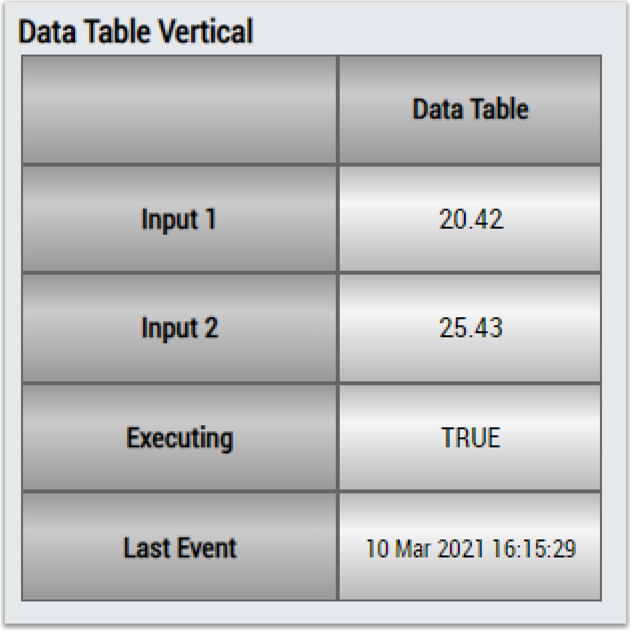
2. The SingleValue control visualizes the output values Count and Last Event.



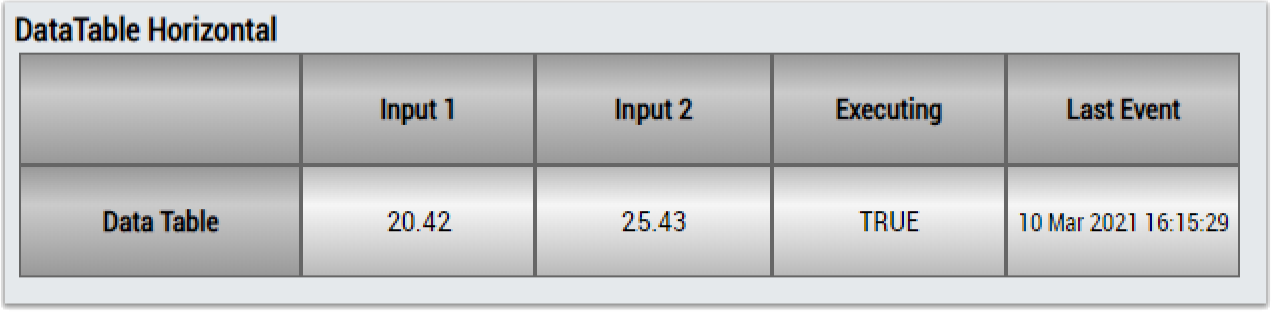
3. The BinaryState control visualizes the output value Operation Out.



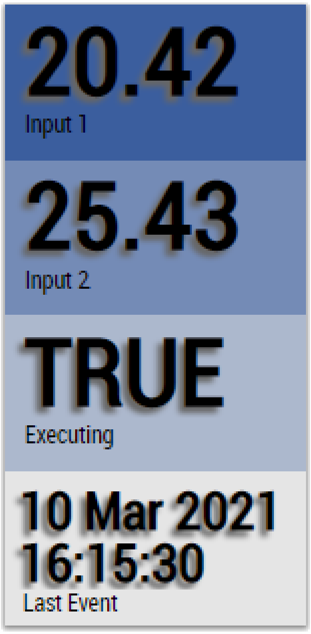
4. The Table Control or Multivalue Control visualizes all output values: Operation Out, Count, Last Event.


Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29


DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

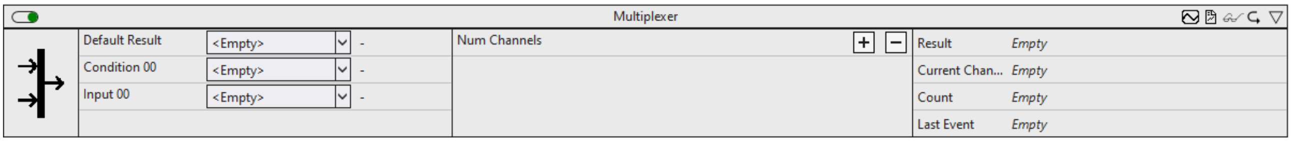
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Logic Operation Counter algorithm using the Mapping Wizard [▶ 432].

6.4.4.7 Multiplexer



Multiplexer

Default Result	<Empty>	-	Result	Empty
Condition 00	<Empty>	-	Current Chan...	Empty
Input 00	<Empty>	-	Count	Empty
			Last Event	Empty

Num Channels

The Multiplexer selects one channel out of one or more input channels. For each input channel a boolean input has to be provided additionally. The output corresponds to the first input channel, where the conditional input is TRUE. The priority of the configured channels is the order of configuration. If the condition is not met for any of the channels, the provided default channel is returned.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of channels. For each channel there are two input variables; one is a boolean condition (*Condition0n*), the other is an input value of any data type (*Input0n*) which can be passed to *Result* if the condition is met.

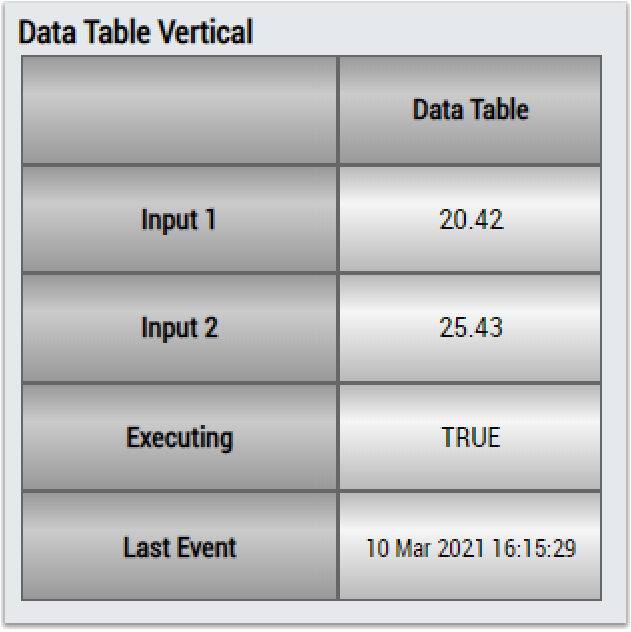
Output values

- **Result:** Delivers the signal *Input0n* of the selected input channel.
- **Current Channel:** Indicates the number of the selected channel. The value is 0 if the default result is selected. The input channels are numbered in the order of their configuration.
- **Count:** Starts with 1 for the channel selected at the start of the analysis and increments each time another channel is selected.
- **Last Event:** Timestamp of the last channel change.

Standard HMI Controls

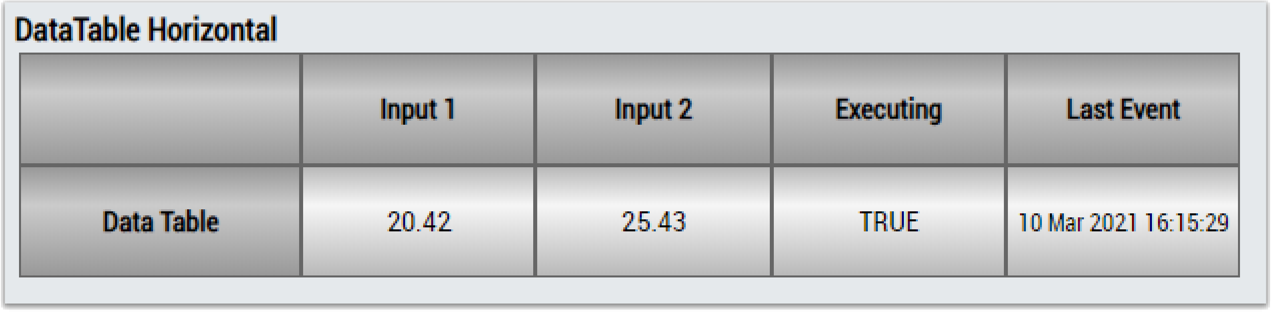
For the Multiplexer algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes the output values Count, Result, Current Channel and Last Event.



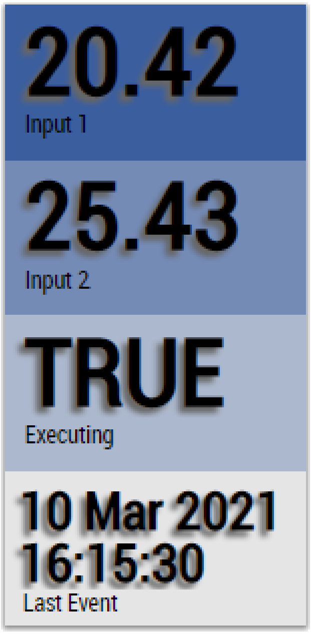
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

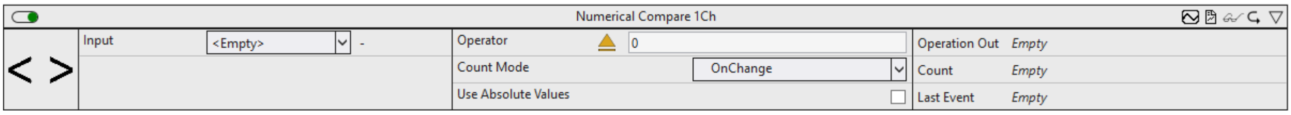
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Multiplexer algorithm using the Mapping Wizard [▶ 432].

6.4.4.8 Numerical Compare 1Ch



Numerical Compare 1Ch

Input	<Empty> -	Operator	0	Operation Out *Empty*
		Count Mode	OnChange	Count *Empty*
		Use Absolute Values	□	Last Event *Empty*

The *Numerical Compare 1Ch* compares the input values with a reference value and provides the result of this comparison operation. The operator, the reference value and the count mode can be configured individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** specifies whether the input value should be greater than, greater than or equal to, less than or equal to, less than or not equal to the reference value.
- **Reference:** reference value for the comparison operation.
- **Count Mode:** mode of the result counter.
  OnChange: the counter counts every time the result changes to TRUE.
  Cyclic: the counter increments every cycle when the condition is *TRUE*.
- **Use Absolute Values:** if the checkbox is checked, the absolute value of the input signal is used.
- **Tolerance (optional):** tolerance value for the Equal / NotEqual comparisons.

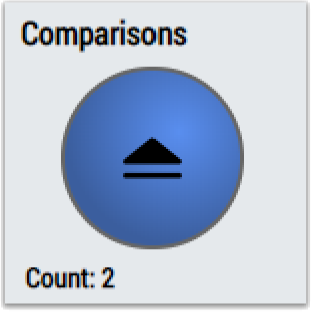
Output values

- **Operation Out:** Result of the comparison operation.
- **Count:** Incremented when the output value is TRUE. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

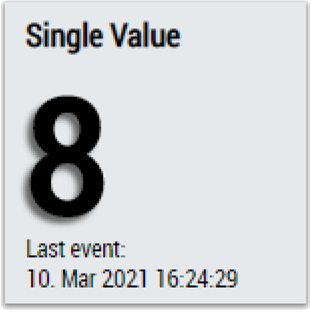
Standard HMI Controls

For the Numerical Compare 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

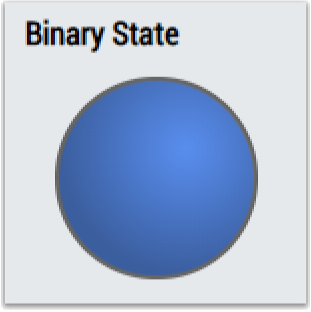
1. The Comparison Control or Multivalue Control visualizes the output values Operation Out and Count as well as the configuration option Operator.



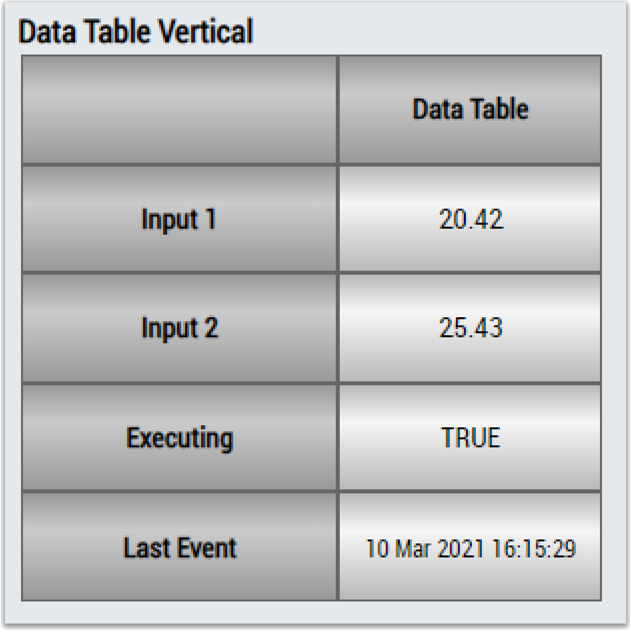
2. The SingleValue control visualizes the output values Count and Last Event.



3. The BinaryState control visualizes the output value Operation Out.

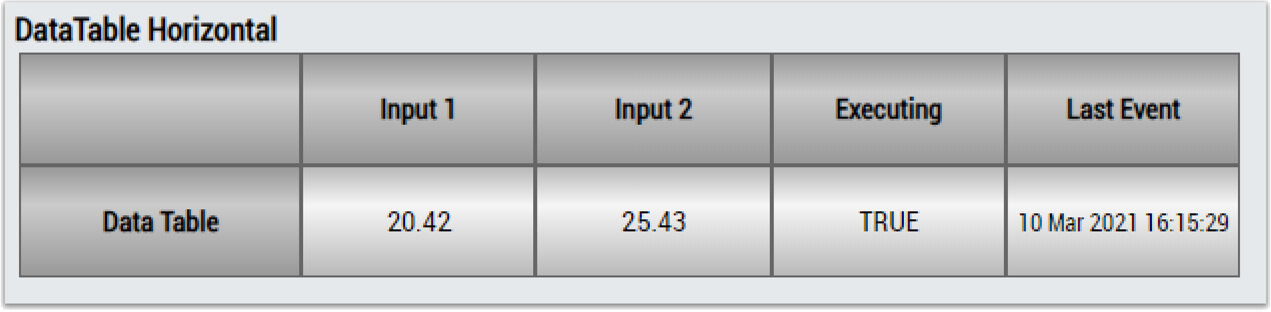


4. The Table Control and Multivalue Control visualize all output values: Operation Out, Count, Last Event.



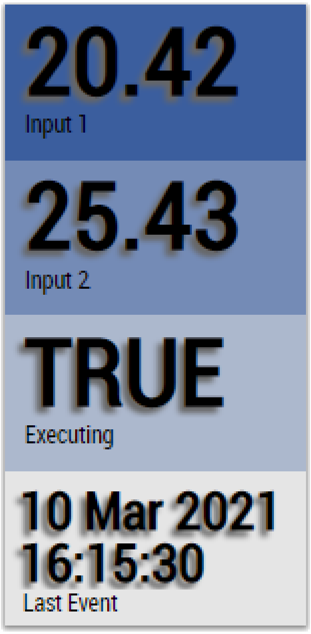
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

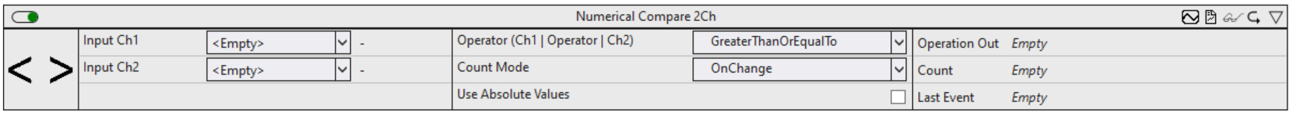
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the *Numerical Compare 1Ch* algorithm using the Mapping Wizard [▶ 432].

6.4.4.9 Numerical Compare 2Ch



Numerical Compare 2Ch

Input Ch1	<Empty> -	Operator (Ch1 | Operator | Ch2)	GreaterThanOrEqualTo	Operation Out	*Empty*
Input Ch2	<Empty> -	Count Mode	OnChange	Count	*Empty*
		Use Absolute Values	□	Last Event	*Empty*

The *Numerical Compare 2Ch* compares the input values of the first channel with the input values of the second channel and provides the result of this comparison operation. The operator and the count mode can be configured individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** specifies whether the input value should be greater than, greater than or equal to, less than or equal to, less than or not equal to the reference value.
- **Count Mode:** mode of the result counter.
  OnChange: the counter counts every time the result changes to TRUE.
  Cyclic: the counter increments every cycle when the condition is *TRUE*.
- **Use absolute values:** if the checkbox is checked, the absolute values of the input signal are used.
- **Tolerance (optional):** tolerance value for the Equal / NotEqual comparisons.

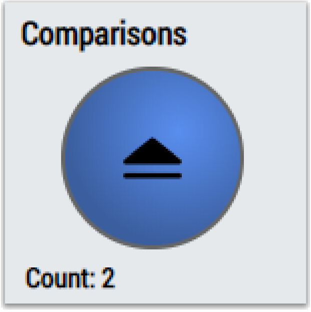
Output values

- **Operation Out:** Result of the comparison operation.
- **Count:** Incremented when the output value is TRUE. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

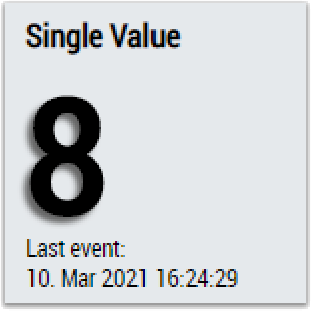
Standard HMI Controls

For the Numerical Compare 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

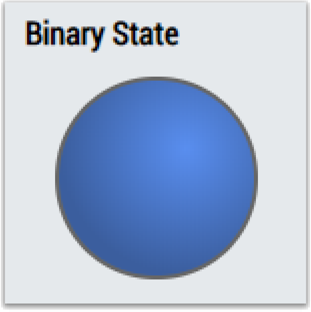
1. The Comparison Control or Multivalue Control visualizes the output values Operation Out and Count as well as the configuration option Operator.



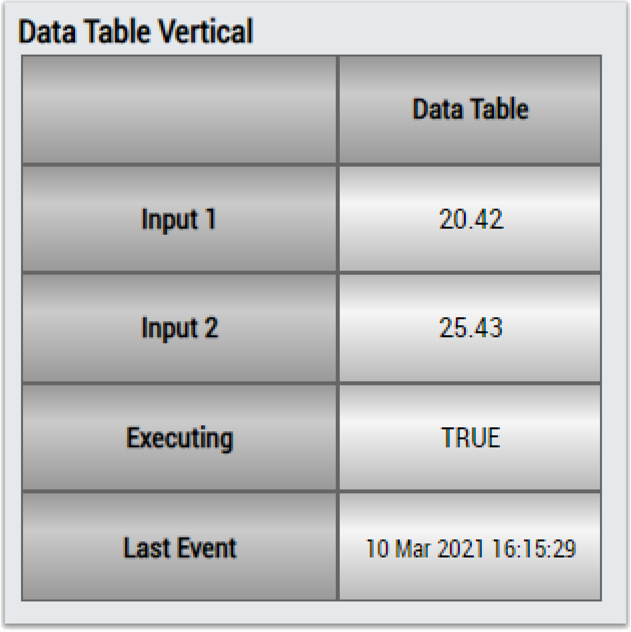
2. The SingleValue control visualizes the output values Count and Last Event.



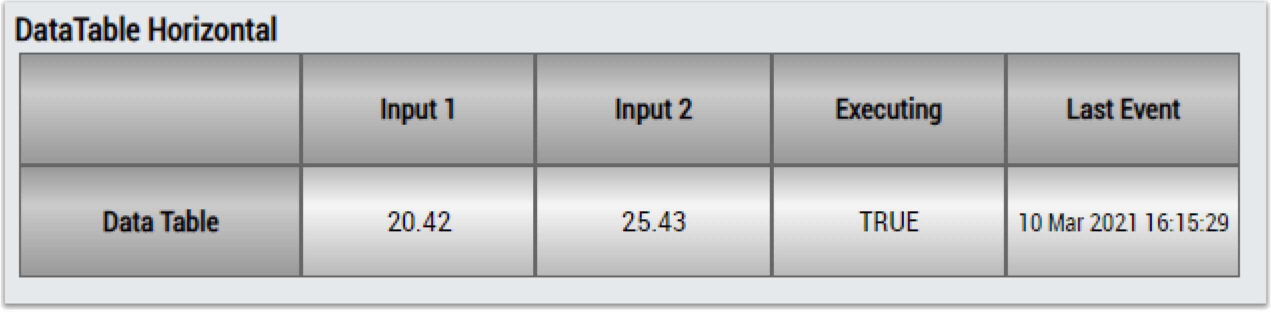
3. The BinaryState control visualizes the output value Operation Out.



4. The Table Control visualizes all output values: Operation Out, Count, Last Event.

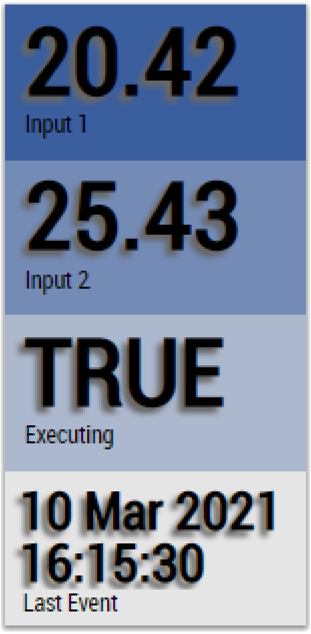

Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

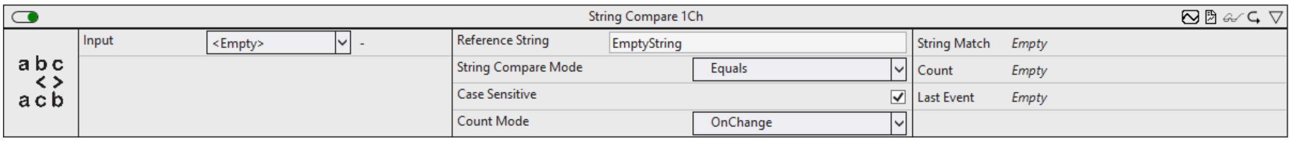
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Numerical Compare 1Ch algorithm using the Mapping Wizard [▸ 432].

6.4.4.10 String Compare 1Ch



String Compare 1Ch

Input <Empty> -

Reference String EmptyString

String Compare Mode Equals

Case Sensitive ☑

Count Mode OnChange

String Match *Empty*

Count *Empty*

Last Event *Empty*

The *String Compare 1Ch* compares the input string with a reference string and counts the string matches. Therefore, case sensitivity can be taken into account or not and the count mode can be changed.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Reference String:** reference string for the comparison operation.
- **String Compare Mode**: enumeration for different string compare modes.
  Equals: input string equals the reference string.
  BeginsWith: input string begins with the reference string.
  Contains: input string contains the reference string.
- **Case Sensitivity:** if the checkbox is checked the input is case-sensitive, otherwise it is not case-sensitive.
- **Count Mode:** mode of the result counter.
  OnChange: the counter counts every time the result changes to TRUE.
  Cyclic: the counter increments every cycle when the condition is *TRUE*.

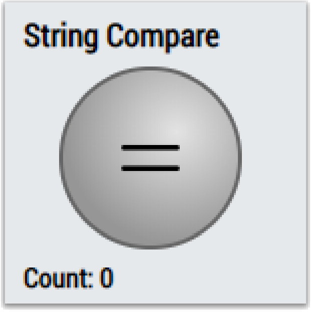
Output values

- **String Match:** *TRUE*, if the input string matches the reference string, otherwise *FALSE*.
- **Count:** Incremented when the output value is TRUE. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

Standard HMI Controls

For the String Compare 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

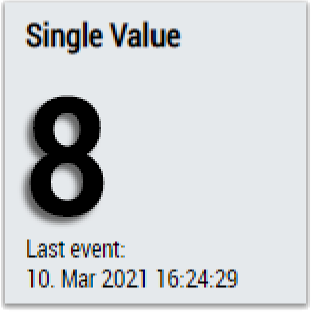
1. The Comparison control visualizes the output values String Match and Count as well as the configuration option String Compare Mode.



String Compare

Count: 0

2. The SingleValue control visualizes the output values Count and Last Event.

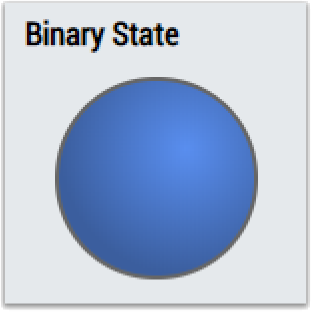


Single Value

8

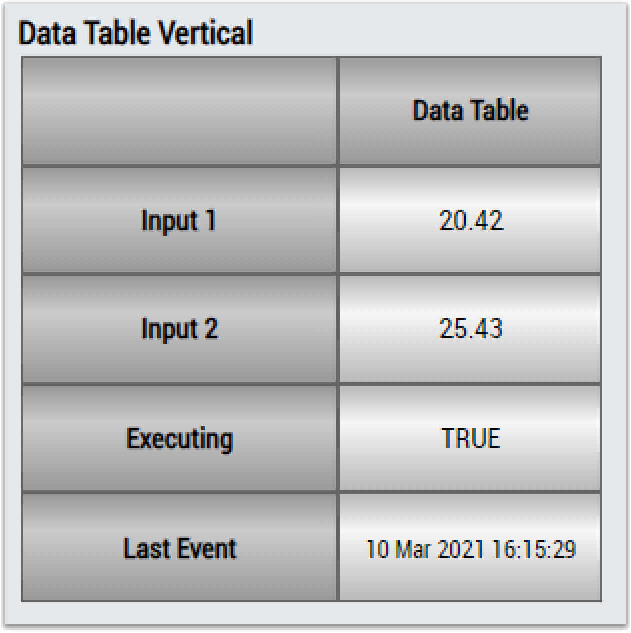
Last event:
10. Mar 2021 16:24:29

3. The BinaryState Control visualizes the output value String Match.



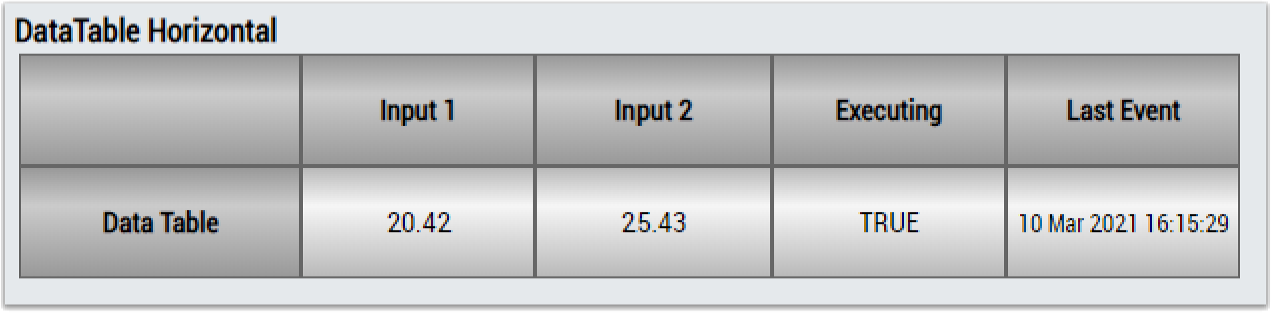
Binary State

4. The Table Control or Multivalue Control visualizes all output values: String Match, Count, Last Event.



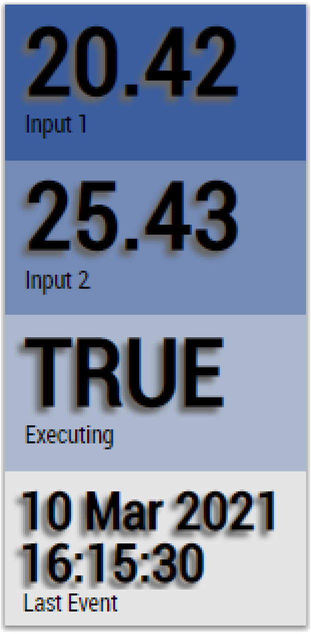
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

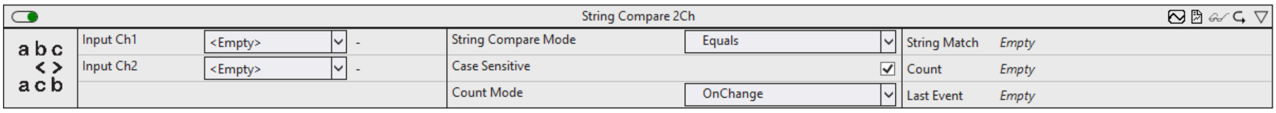
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the String Compare 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.4.11 String Compare 2Ch



String Compare 2Ch

Input Ch1	<Empty>	-	String Compare Mode	Equals	String Match	*Empty*
Input Ch2	<Empty>	-	Case Sensitive	☑	Count	*Empty*
			Count Mode	OnChange	Last Event	*Empty*

The *String Compare 2Ch* compares the values of the first input string with the values of the second string and counts the string matches. Therefore case sensitivity can be taken into account or not and the count mode can be changed.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **String Compare Mode**: enumeration for different string compare modes.
  Equals: input string equals the reference string.
  BeginsWith: input string begins with the reference string.
  Contains: input string contains the reference string.
- **Case Sensitivity:** if the checkbox is checked the input is case-sensitive, otherwise it is not case-sensitive.
- **Count Mode:** mode of the result counter.
  OnChange: the counter counts every time the result changes to TRUE.
  Cyclic: the counter increments every cycle when the condition is *TRUE*.

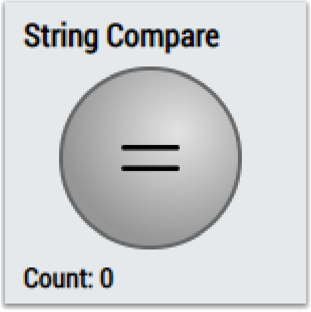
Output values

- **String Match:** *TRUE*, if the value of the first input string matches the value of the second input string, otherwise *FALSE*.
- **Count:** Incremented when the output value is TRUE. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

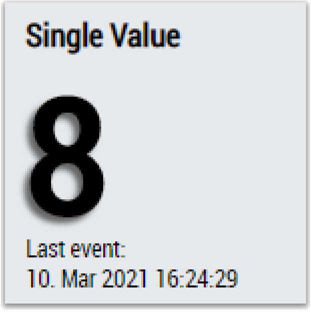
Standard HMI Controls

For the String Compare 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

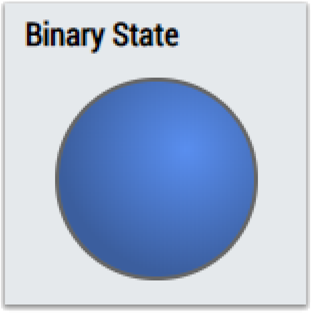
1. The Comparison control visualizes the output values String Match and Count as well as the configuration option String Compare Mode.



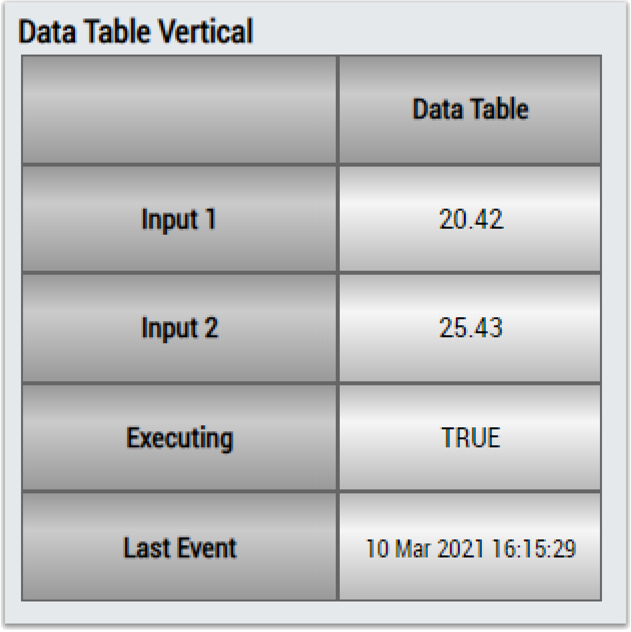
2. The SingleValue control visualizes the output values Count and Last Event.



3. The BinaryState Control visualizes the output value String Match.

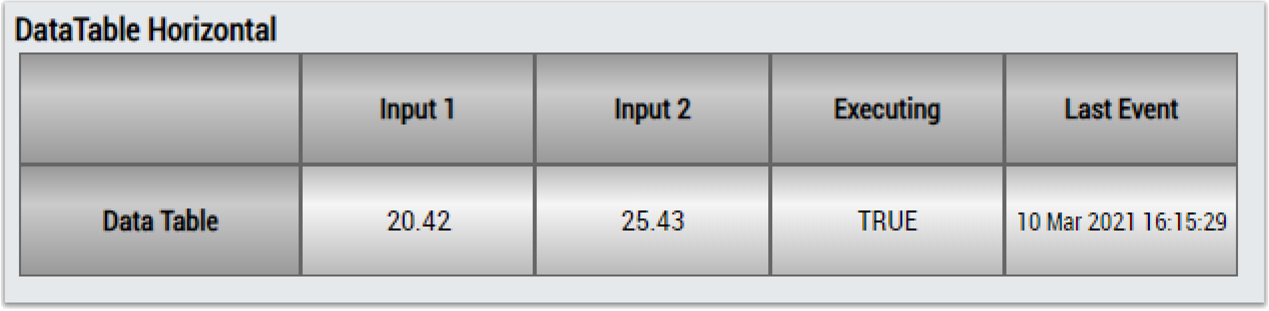


4. The Table Control or Multivalue Control visualizes all output values: String Match, Count, Last Event.



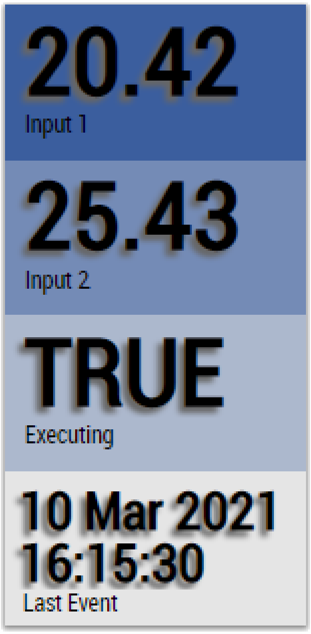
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

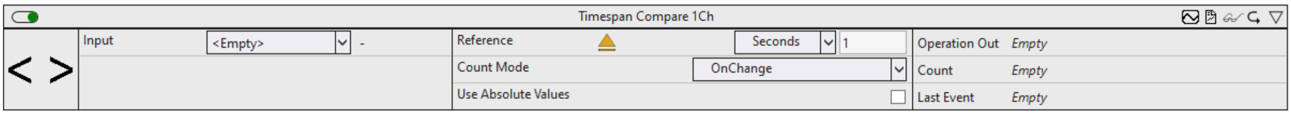
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the String Compare 2Ch algorithm using the Mapping Wizard [▶ 432].

6.4.4.12 Timespan Compare 1Ch



Timespan Compare 1Ch

Input <Empty> -

Reference Seconds 1

Count Mode OnChange

Use Absolute Values □

Operation Out Empty

Count Empty

Last Event Empty

The *Timespan Compare 1Ch* compares the input value with a reference value and returns the result of this comparison operation. A timespan with a resolution of 1 ns is supported as an input. The operator, the reference value and the count mode can be configured individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Operator:** specifies whether the input value should be greater than, greater than or equal to, less than or equal to, less than or not equal to the reference value.
- **Reference:** Reference timespan for the comparison operation.
- **Count Mode:** Mode of the result counter.
  OnChange: The counter counts every time the result changes to *TRUE*.
  Cyclic: The counter increments every cycle when the condition is *TRUE*.
- **Use Absolute Values:** If the checkbox is checked, the absolute value of the input signal is used.

Output values

- **Operation Out:** Result of the comparison operation.
- **Count:** Incremented when the output value is *TRUE*. The behavior depends on the parameter *Count Mode*.
- **Last Event:** Indicates the time of the last triggered event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event

Standard HMI Controls

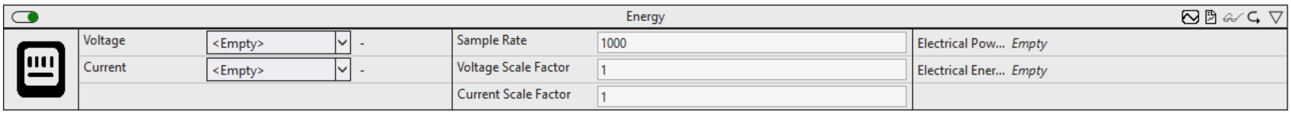
For the Timespan Compare 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control visualizes all output values: Operation Out, Count, Last Event.

Alternatively, customer-specific HMI controls can be mapped in the *Timespan Compare 1Ch* algorithm using the Mapping Wizard [▸ 432].

6.4.5 Analytics - Energy

6.4.5.1 Energy



Energy

Voltage	<Empty>	-	Sample Rate	1000	Electrical Pow... *Empty*
Current	<Empty>	-	Voltage Scale Factor	1	Electrical Ener... *Empty*
			Current Scale Factor	1	

The *Energy module* is suitable for energy monitoring of various electrical components. The input data for current and voltage are averaged over a configurable interval and then calculated.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Voltage Scale Factor:** Scale factor of the input voltage.
- **Current Scale Factor:** Scale factor of the input current.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

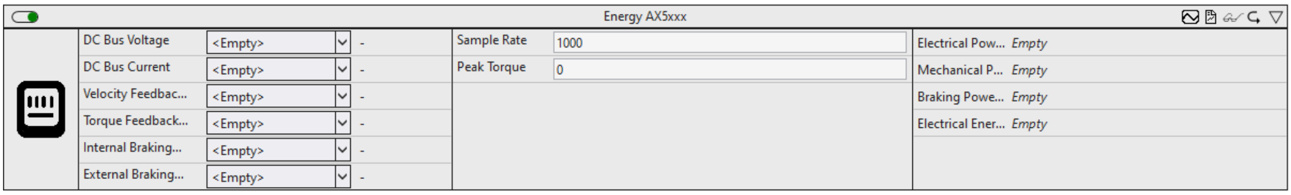
Input values

- **Voltage:** RMS value of the voltage in V.
- **Current:** RMS value of the current in A.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.2 Energy AX5xxx



Energy AX5xxx

DC Bus Voltage	<Empty>	-	Sample Rate	1000	Electrical Pow... *Empty*
DC Bus Current	<Empty>	-	Peak Torque	0	Mechanical P... *Empty*
Velocity Feedbac...	<Empty>	-			Braking Powe... *Empty*
Torque Feedback...	<Empty>	-			Electrical Ener... *Empty*
Internal Braking...	<Empty>	-			
External Braking...	<Empty>	-			

The *Energy AX5xxx* module is suitable for energy monitoring of servo drives from the AX5xxx series. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Peak Torque:** Peak value of the torque in Nm
  Process data: CoE: P-0-0094
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

Input values

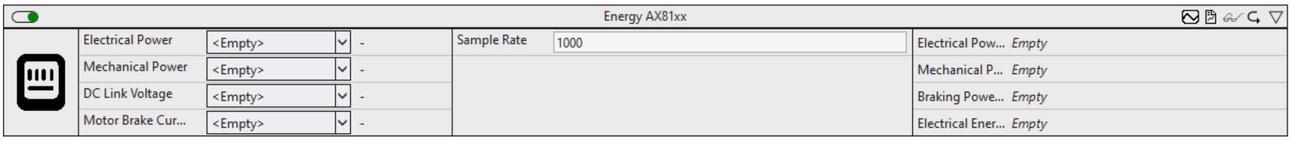
- **DC Bus Voltage:** DC bus voltage in 0.1 V.
  Process data: CoE: S-0-0016. Add that of the CoE S-0-0380 "DC bus voltage" to the PDO content.

- **DC Bus Current:** Input current to the DC link in mA.
  Process data: CoE: S-0-0016. Add that of CoE S-0-0381 "DC bus current" to the PDO content.
- **Velocity Feedback Value:** Motor velocity.
  Process data: CoE: S-0-0016. Add that of the CoE S-0-0040 "Velocity feedback 1 value" to the PDO content.
- **Torque Feedback Value:** Torque-forming current.
  Process data CoE: S-0-0016. Add that of CoE S-0-0084 "Torque/force feedback value" to the PDO content.
- **Internal Braking Resistor Power:** Braking power output via the internal braking resistor in W.
  Process data: CoE: S-0-0016. Add that of the CoE P-0-0209 "Internal braking resistor actual averaged power" to the PDO content.
- **External Braking Resistor Power:** Braking power output via the external braking resistor in W.
  Process data: CoE: S-0-0016. Add the CoE P-0-0210 "External braking resistor actual averaged power" to the PDO content.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Mechanical Power [W]:** Mechanical power in W.
- **Braking Power [W]:** Braking power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.3 Energy AX81xx



Electrical Power	<Empty>	-	Sample Rate	1000	Electrical Pow...	Empty
Mechanical Power	<Empty>	-			Mechanical P...	Empty
DC Link Voltage	<Empty>	-			Braking Powe...	Empty
Motor Brake Cur...	<Empty>	-			Electrical Ener...	Empty

The *Energy AX81xx* module is suitable for energy monitoring of servo drives from the AX81xx series. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

Input values

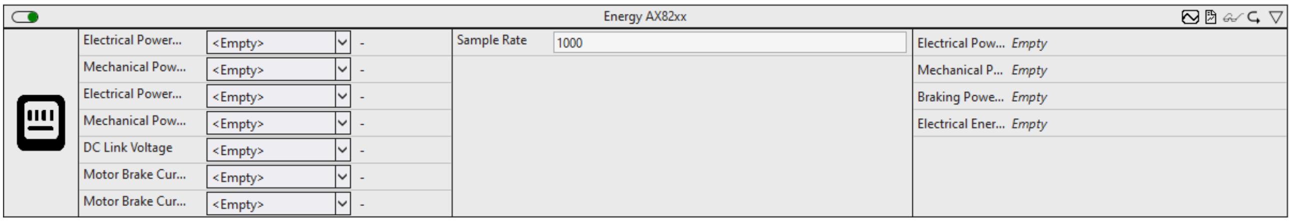
- **Electrical power:** Electrical power in W.
  Process data: Enable PDO input 0x1A32 and add CoE 0x3242:12 "Electrical power" to the PDO content.
- **Mechanical Power:** Mechanical power in W.
  Process data: Enable PDO input 0x1A32 and add CoE 0x3242:13 " Mechanical power" to the PDO content.
- **DC link voltage:** DC link voltage in mV.
  Process data: Enable PDO input 0x1A00 and add CoE 0x6079 "DC link circuit voltage" to the PDO content.
- **Motor Brake Current:** Brake current of the motor in A.
  Process data: Enable PDO input 0x1A20 and add CoE 0x3001:01 "Actual motor brake current" to the PDO content.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Mechanical Power [W]:** Mechanical power in W.

- **Braking Power [W]:** Braking power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.4 Energy AX82xx



Energy AX82xx

Electrical Power... <Empty> -
Mechanical Pow... <Empty> -
Electrical Power... <Empty> -
Mechanical Pow... <Empty> -
DC Link Voltage <Empty> -
Motor Brake Cur... <Empty> -
Motor Brake Cur... <Empty> -

Sample Rate 1000

Electrical Pow... *Empty*
Mechanical P... *Empty*
Braking Powe... *Empty*
Electrical Ener... *Empty*

The *Energy AX82xx* module is suitable for energy monitoring of servo drives from the AX82xx series. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

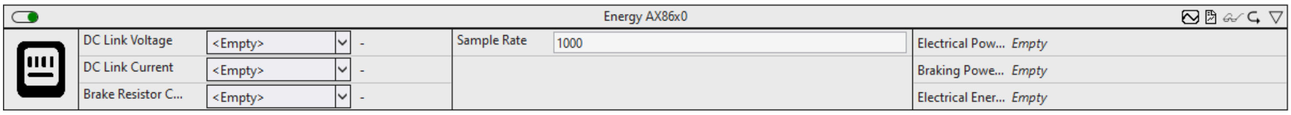
Input values

- **Electrical Power ChA:** Electrical power of the first channel in W.
  Process data: Enable PDO input 0x1A32 and add CoE 0x3242:12 "Electrical power" to the PDO content.
- **Mechanical Power ChA:** Mechanical power of the first channel in W.
  Process data: Enable the PDO input 0x1A32 and add the CoE 0x3242:13 " Mechanical power" to the PDO content.
- **Electrical Power ChB:** Electrical power of the second channel in W.
  Process data: Enable PDO input 0x1A52 and add CoE 0x3642:12 "Electrical power" to the PDO content.
- **Mechanical Power ChB:** Mechanical power of the second channel in W.
  Process data: Enable the PDO input 0x1A52 and add the CoE 0x3642:13 "Mechanical power" to the PDO content.
- **DC link voltage:** DC link voltage in mV.
  Process data: Enable PDO input 0x1A00 and add CoE 0x6079 "DC link circuit voltage" to the PDO content.
- **Motor Brake Current ChA:** Brake current of the motor of the first channel in A.
  Process data: Enable PDO input 0x1A20 and add CoE 0x3001:01 "Actual motor brake current" to the PDO content.
- **Motor Brake Current ChB:** Brake current of the motor of the second channel in A.
  Process data: Enable PDO input 0x1A40 and add CoE 0x3401:01 "Actual motor brake current" to the PDO content.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Mechanical Power [W]:** Mechanical power in W.
- **Braking Power [W]:** Braking power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.5 Energy AX86x0


Energy AX86x0

DC Link Voltage	<Empty>	Sample Rate	1000	Electrical Pow... Empty
DC Link Current	<Empty>			Braking Powe... Empty
Brake Resistor C...	<Empty>			Electrical Ener... Empty

The *Energy AX86xx* module is suitable for energy monitoring of AX86xx series power supply modules. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

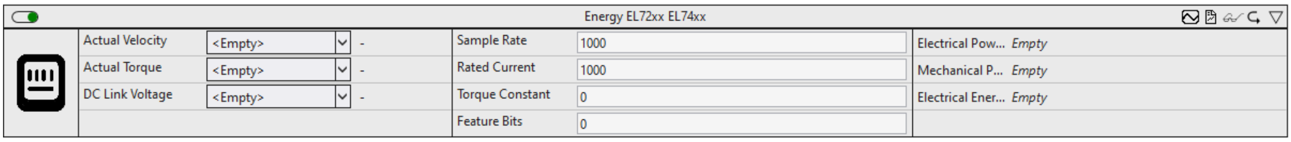
Input values

- **DC Link Voltage:** DC link voltage in mV.
  Process data: Enable the PDO input 0x1A05.
- **DC Link Current:** Input current to the DC link in 0.01 A.
  Process data: Enable the PDO input 0x1A6.
- **Brake Resistor Continuous Power:** Power output via the braking resistor in W.
  Process data: Enable the PDO input 0x1A07.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Braking Power [W]:** Braking power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.6 Energy EL72xx EL74xx


Energy EL72xx EL74xx

Actual Velocity	<Empty>	Sample Rate	1000	Electrical Pow... Empty
Actual Torque	<Empty>	Rated Current	1000	Mechanical P... Empty
DC Link Voltage	<Empty>	Torque Constant	0	Electrical Ener... Empty
		Feature Bits	0	

The *Energy AX72xx EL74xx* module is suitable for energy monitoring of EL72xx and EL74xx EtherCAT Terminals. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Rated Current:** Nominal current in mA.
  Process data: CoE: 0x8011:12.
- **Torque Constant:** Torque constant in mNm / A.
  Process data: CoE: 0x8011:16.
- **Feature bits:** Feature bits for specifying the current type.
  Process data: CoE: 0x8010:53.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

Input values

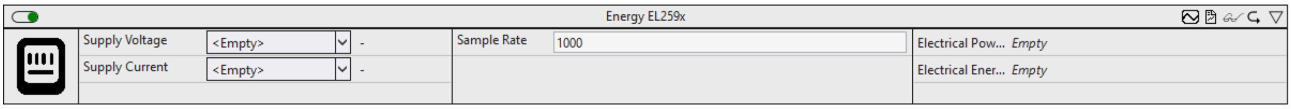
- **Actual velocity:** Actual velocity value.
  Process data: Enable the PDO input 0x1A02.

- **Actual Torque:** Actual value of the torque.
  Process data: Enable the PDO input 0x1A03.
- **DC link voltage:** DC link voltage in mV.
  Process data: Enable PDO input 0x1A04 and select "DC link voltage (mV)" in CoE 0x8010:39.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Mechanical Power [W]:** Mechanical power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.7 Energy EL259x



Energy EL259x

Supply Voltage <Empty> -
Supply Current <Empty> -
Sample Rate 1000
Electrical Pow... *Empty*
Electrical Ener... *Empty*

The *Energy EL259x* module is suitable for energy monitoring of EL259x EtherCAT Terminals. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

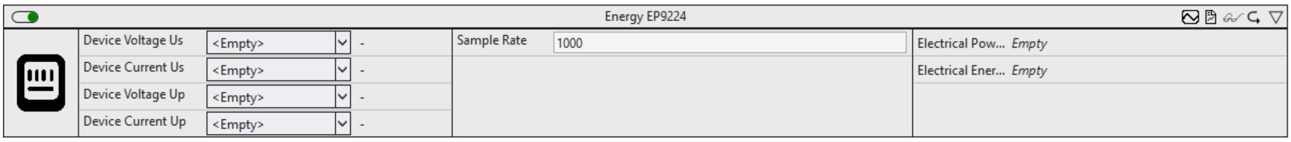
Input values

- **Supply Voltage:** Supply voltage in 0.01 V.
  Process data: Enable PDO input 0x1A01 and select CoE 0xF900:05 "Supply voltage" in CoE 0x8002:11.
- **Supply Current:** Input current in mA.
  Process data: Enable PDO input 0x1A01 and select CoE 0xF900:06 "Supply current" in CoE 0x8002:19.

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.8 Energy EP9224



Energy EP9224

Device Voltage Us <Empty> -
Device Current Us <Empty> -
Device Voltage Up <Empty> -
Device Current Up <Empty> -
Sample Rate 1000
Electrical Pow... *Empty*
Electrical Ener... *Empty*

The *Energy EP9224* module is suitable for the energy monitoring of EP9224 EtherCAT Box modules. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

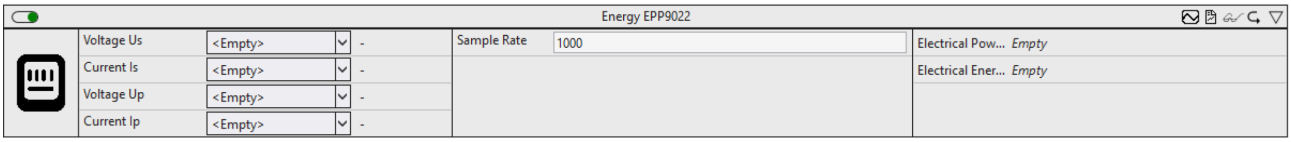
Input values

- **Device Voltage Us:** Control voltage in V.
  Process data: PDO Inputs Device
- **Device Current Us:** Control current in A.
  Process data: PDO Inputs Device
- **Device Voltage Up:** Load voltage in V.
  Process data: PDO Inputs Device
- **Device Current Up:** Load current in A.
  Process data: PDO Inputs Device

Output values

- **Electrical Power [W]:** Electrical power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.5.9 Energy EPP9022



Energy EPP9022

Voltage Us <Empty> -
Current Is <Empty> -
Voltage Up <Empty> -
Current Ip <Empty> -

Sample Rate 1000

Electrical Pow... Empty
Electrical Ener... Empty

The *Energy EPP9022* module is suitable for the energy monitoring of EPP9022 EtherCAT Box modules. The input data is averaged over a configurable interval and then calculated. To ensure that the required input data is available, it must be added to the process image.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Interval (optional):** By default, the input data is averaged over an interval of one second.

Input values

- **Device Voltage Us:** Control voltage in V.
  Process data: PDO Inputs Device
- **Device Current Is:** Control current in A.
  Process data: PDO Inputs Device
- **Device Voltage Up:** Load voltage in V.
  Process data: PDO Inputs Device
- **Device Current Ip:** Load current in A.
  Process data: PDO Inputs Device

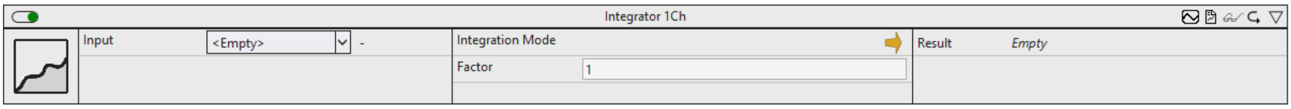
Output values

- **Electrical Power [W]:** Electrical power in W.
- **Electrical Energy [Wh]:** Electrical energy in Wh.

6.4.6 Analytics - Math

The algorithms of the category *Analytics-Math* provide functionalities for mathematical operations such as basic arithmetic operation, integration or slope analysis.

6.4.6.1 Integrator 1Ch



Integrator 1Ch

Input <Empty> -

Integration Mode
Factor 1

Result Empty

The *Integrator 1Ch* integrates the input value over time with a base unit of one second and provides the result of this integration operation. For the approximation of this integral the trapezoidal rule is used. The trapezoidal $T(t_n, t_{n+1})$ between two subsequent timestamps t_n and t_{n+1} with the values y_n and y_{n+1} is calculates as

$$T(t_n, t_{n+1}) = (t_{n+1}[s] - t_n[s]\ ) \cdot \frac{y_n + y_{n+1}}{2} \; .$$

If the integration mode "absolute" ("|x|") is chosen in the configuration, y_n and y_{n+1} are substituted by their absolute values in the above equation.

In each cycle the trapezoidal between the current and the last timestamp is calculated and added to the sum of trapezoids starting from the beginning of the analysis. Additionally, this sum can be scaled by a factor that can be configured individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Integration Mode:** You can select an integration mode. "→": the input value is will be integrated directly. "|x|": The absolute values of the input signal will be integrated.
- **Factor:** With this factor the integral is multiplied.

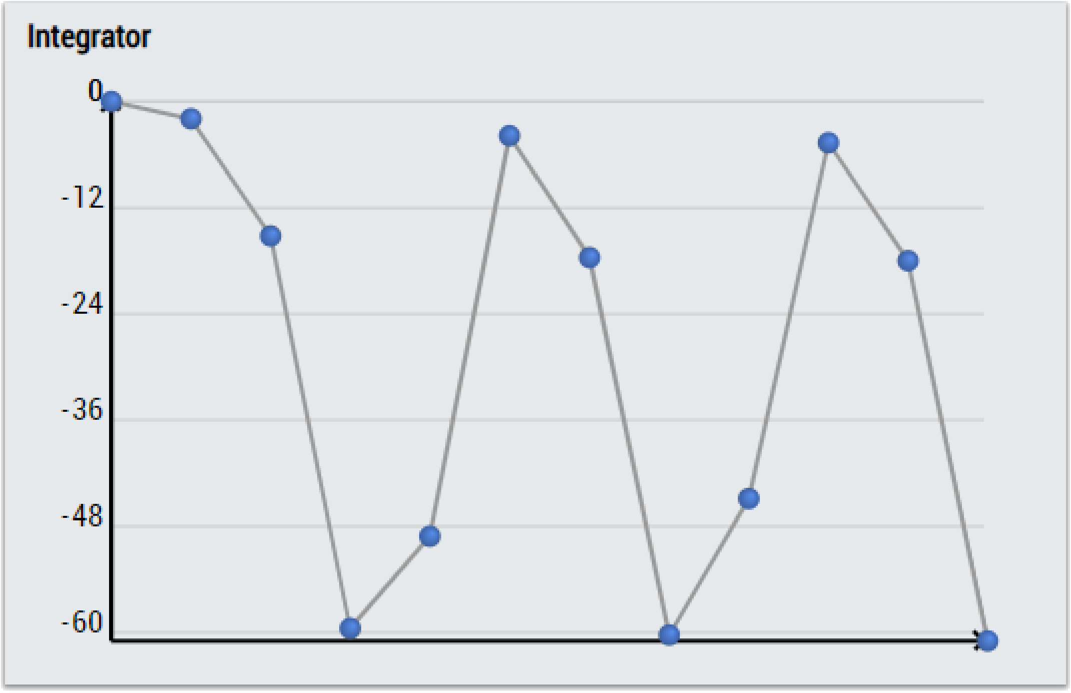
Output Values

- **Result:** Shows the result of the integration operation.

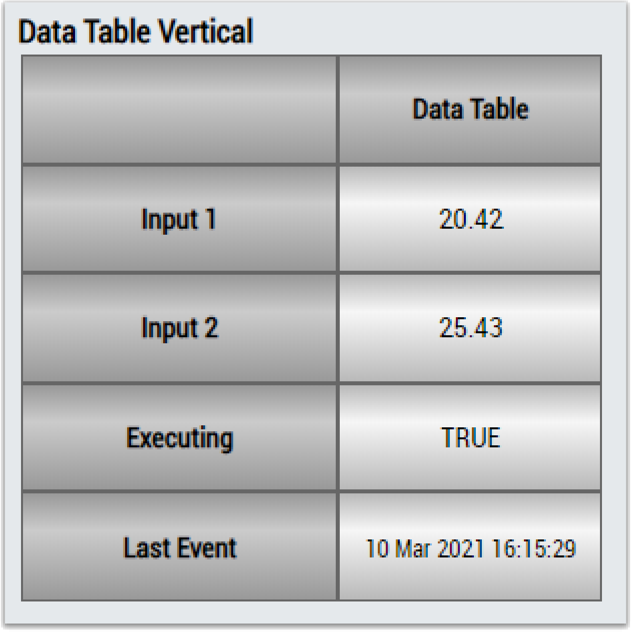
Standard HMI Controls

For the Integrator 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Integrator Control visualizes the last x output values Result.

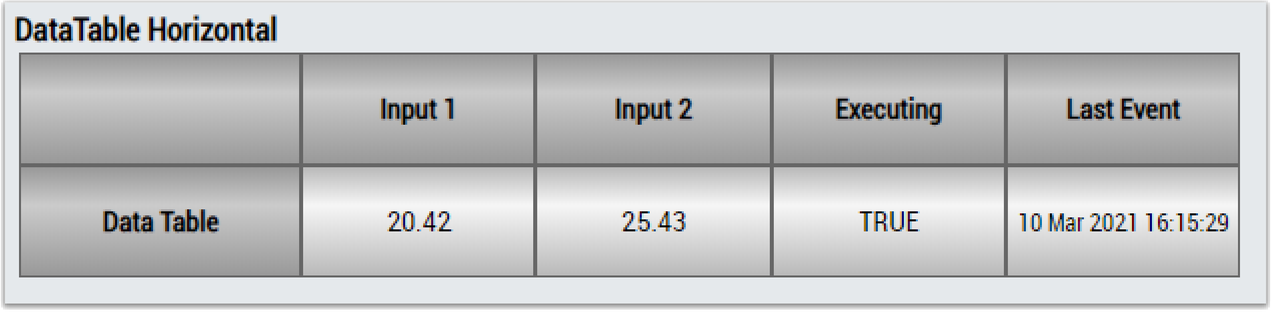


2. The Table Control or Multivalue Control visualizes all output values: Result.



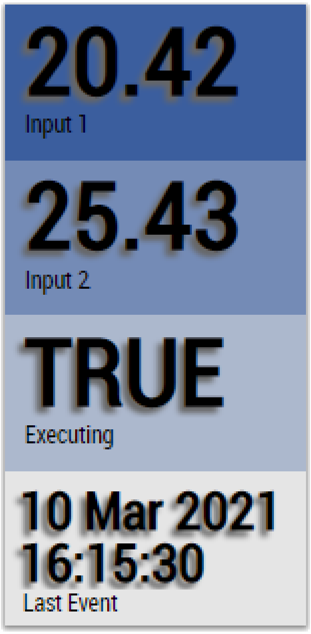
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

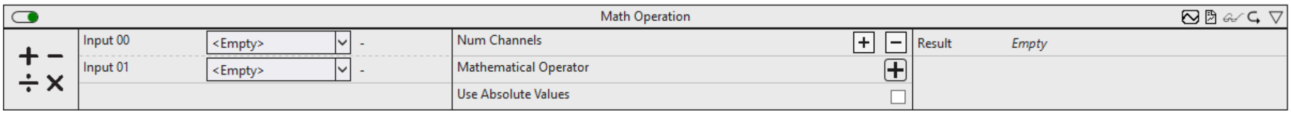
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Integrator 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.6.2 Math Operation



Math Operation

Input 00	<Empty>	-	Num Channels		Result Empty
Input 01	<Empty>	-	Mathematical Operator		
			Use Absolute Values	□	

The *Math Operation* executes a mathematical operation on two or more different input channels and provides the result of the mathematical operation. The operator is the same for all operands.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of operands that are inputs to the function.
- **Mathematical Operator:** Mathematical operator of the operation ("+", "-", "x", "/", "x^y", "%").
- **Use Absolute Values:** If the checkbox is checked, the absolute values of the input signal are used.

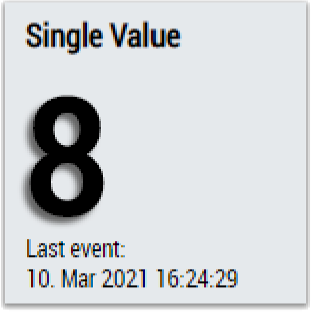
Output Values

- **Result:** Result of the mathematical operation.

Standard HMI Controls

For the Math Operation algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue control visualizes the output value Result.

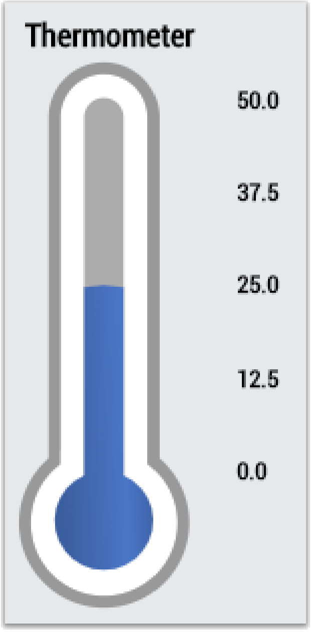


Single Value

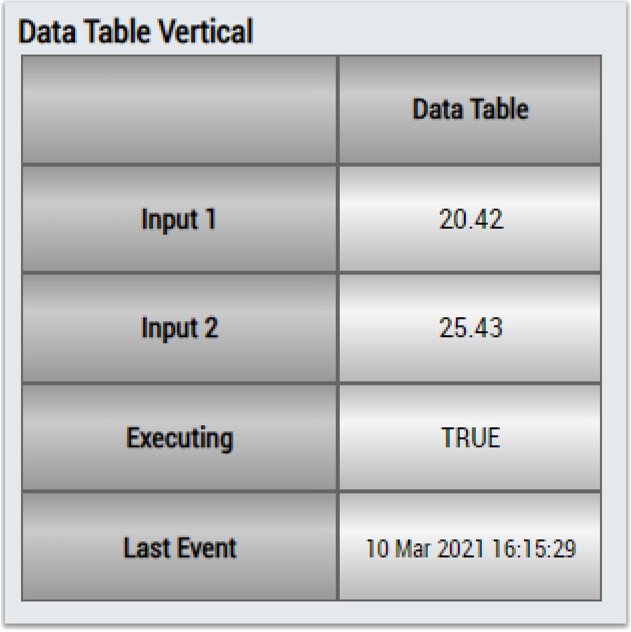
8

Last event:
10. Mar 2021 16:24:29

2. The Thermometer control visualizes the Result output value and can be used for temperature displays.

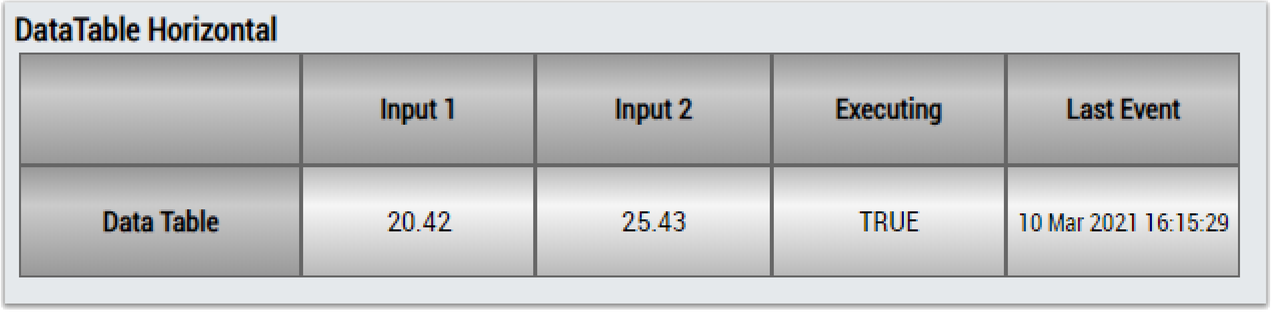


3. The Table Control or Multivalue Control visualizes all output values: Result.



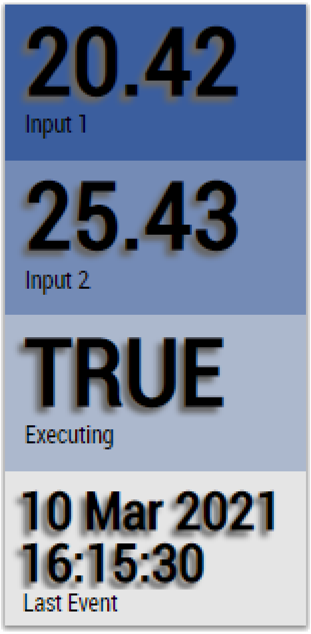
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

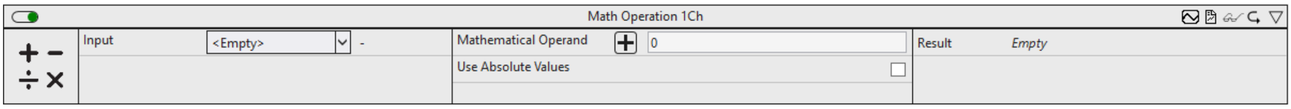
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Math Operation algorithm using the Mapping Wizard [▶ 432].

6.4.6.3 Math Operation 1Ch



Math Operation 1Ch

Input <Empty> -

Mathematical Operand + 0

Use Absolute Values □

Result Empty

The Math Operation 1Ch executes a mathematical operation on the signal of the input channel and a reference value. The algorithm provides the result of the mathematical operation and the operator can be configured individually.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Mathematical Operator:** Mathematical operator of the operation ("+", "-", "x", "/", "x^y", "%").
- **Mathematical Operand:** Mathematical operand for the operation.
- **Use Absolute Values:** If the checkbox is checked, the absolute values of the two operands are used.

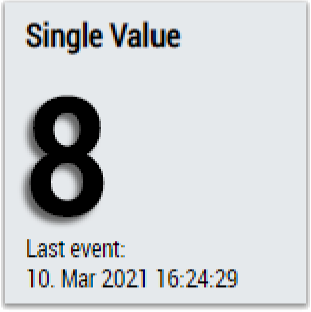
Output Values

- **Result:** Result of the mathematical operation.

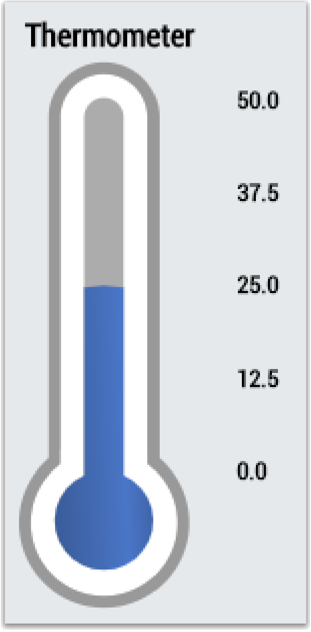
Standard HMI Controls

For the Math Operation algorithm, the following HMI controls are available for generating an Analytics Dashboard:

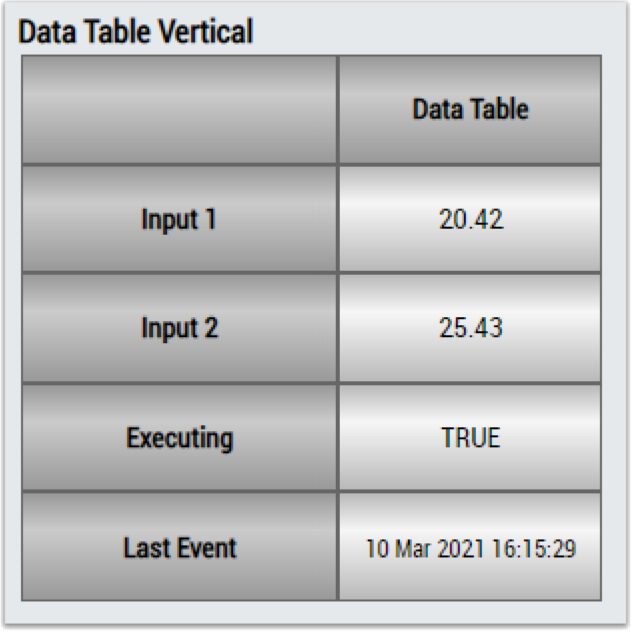
1. The SingleValue control visualizes the output value Result.



2. The Thermometer control visualizes the Result output value and can be used for temperature displays.

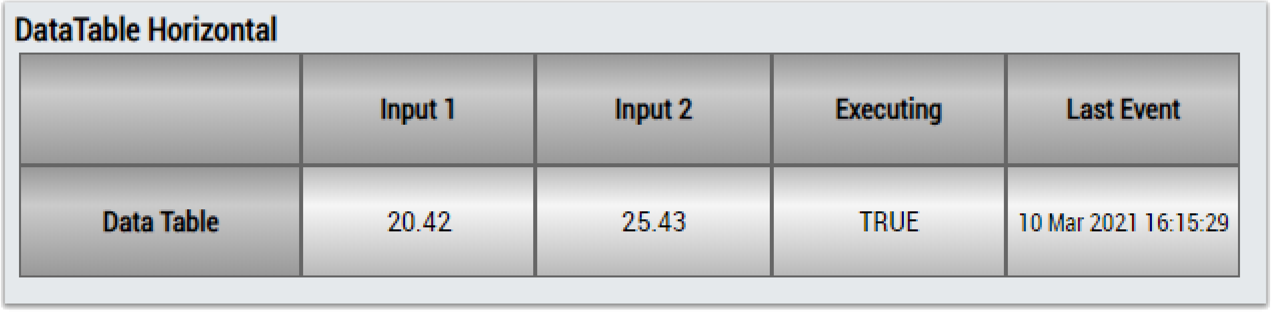


3. The Table Control or Multivalue Control visualizes all output values: Result.



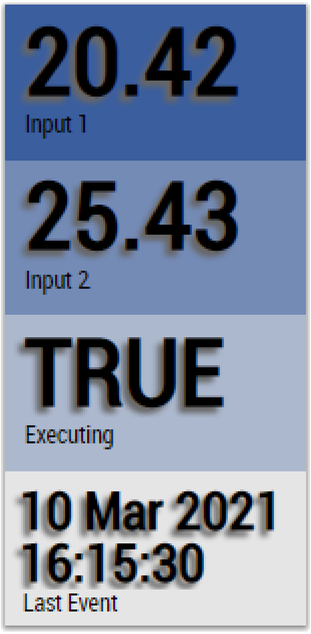
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

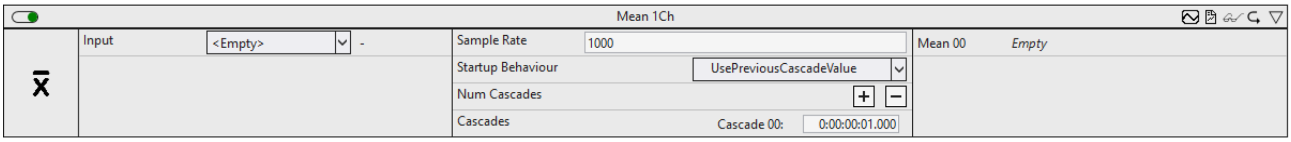
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Math Operation algorithm using the Mapping Wizard [▶ 432].

6.4.6.4 Mean 1Ch



Mean 1Ch

Input <Empty> -

Sample Rate 1000

Startup Behaviour UsePreviousCascadeValue

Num Cascades + −

Cascades Cascade 00: 0:00:00:01.000

Mean 00 *Empty*

The *Mean 1Ch* calculates the mean value over the input values according to the formula

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x$$

The number of samples N that are included in the calculation can be configured by specifying a time interval. A cascaded output can be configured to realize a long-term mean value in a resource-saving way and to pick up intermediate results. The time interval of the configured cascade must correspond to an integer multiple of the time interval of the previous cascade.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Cascades:** Number of output cascades
- **Cascades:** Configuration of the output cascades. The time interval of the configured cascade must correspond to an integer multiple of the time interval of the previous cascade.
- **Sample Rate:** Sample rate of the system to be analyzed in Hz.
- **Startup Behavior:**
  WaitUntilFilled waits until the configured timespan of the cascade has elapsed. The result and the "NewResult" Boolean flag are only set for the first time after the timespan has expired.
  UsePreviousCascadeValue: The cascades whose configured timespan has not yet expired use the next smallest result that has already been set.

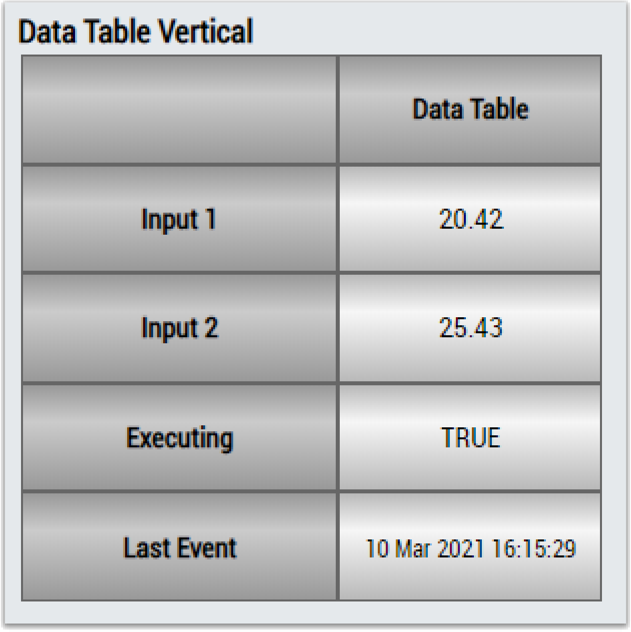
Output values

- **Mean:** Output array in which the results of the mean value calculations are saved. The dimension corresponds to the number of set cascades. The startup behavior can be set via the *Startup Behavior* parameter.

Standard HMI Controls

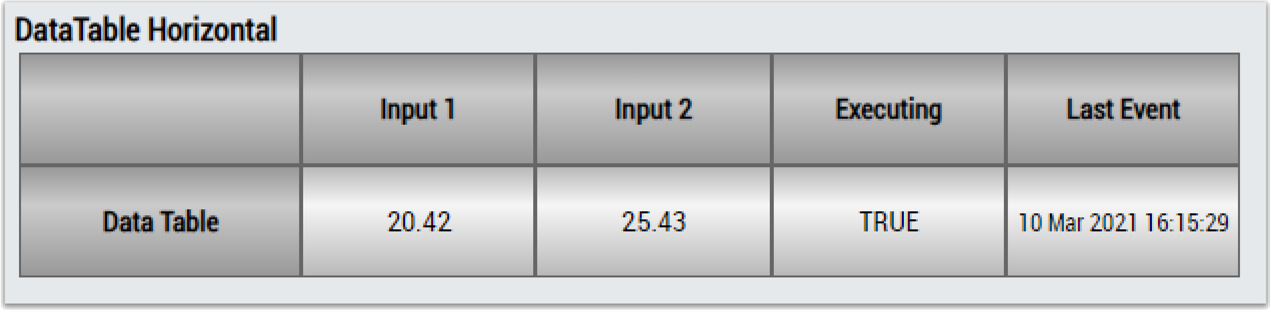
For the Mean algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values: Mean Results.



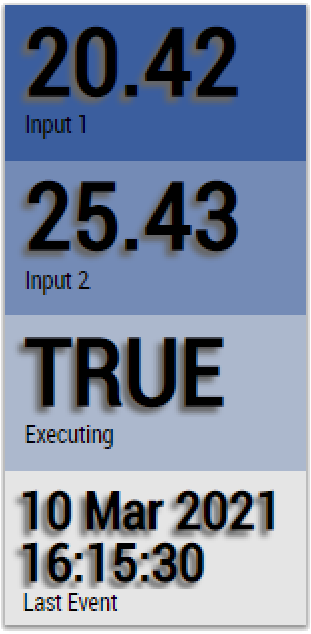
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

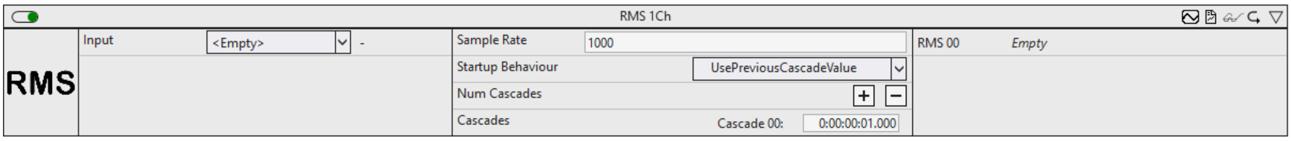
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Mean algorithm using the Mapping Wizard [▶ 432].

6.4.6.5 RMS 1Ch



RMS 1Ch

RMS

Input <Empty> -

Sample Rate 1000

Startup Behaviour UsePreviousCascadeValue

Num Cascades

Cascades Cascade 00: 0:00:00:01.000

RMS 00 *Empty*

RMS 1Ch calculates the root mean square over the input values according to the formula

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x[n]^2}$$

The number of samples N that are included in the calculation can be configured by specifying a time interval. A cascaded output can be configured to realize a long-term RMS in a resource-saving way and to pick up intermediate results. The time interval of the configured cascade must correspond to an integer multiple of the time interval of the previous cascade.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Cascades:** Number of output cascades
- **Cascades:** Configuration of the output cascades. The time interval of the configured cascade must correspond to an integer multiple of the time interval of the previous cascade.

- **Sample Rate:** Sample rate of the system to be analyzed in Hz
- **Startup Behaviour:**
  WaitUntilFilled waits until the configured timespan of the cascade has elapsed. The RMS result and the Boolean flag "NewResult" are only set for the first time after the timespan has elapsed.
  UsePreviousCascadeValue The RMS cascades whose configured timespan has not yet expired use the next smallest RMS result already set.

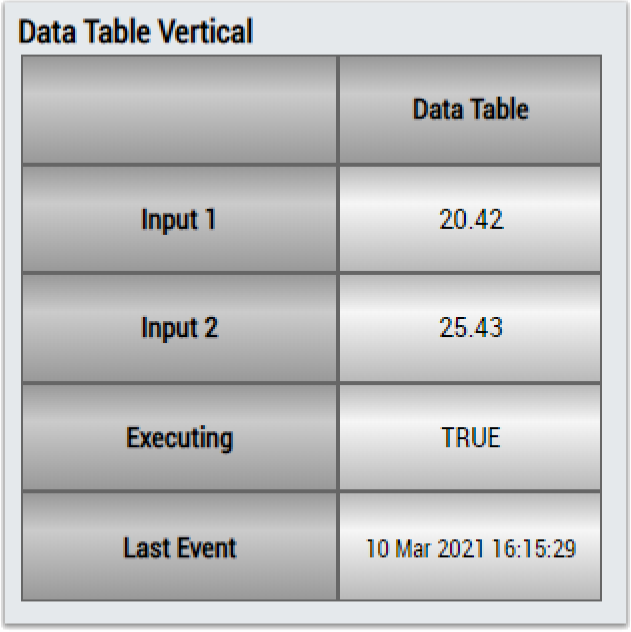
Output values

- **RMS:** Output array in which the results of the RMS calculations are stored. The dimension corresponds to the number of set cascades. The startup behavior can be set via the parameter *Startup Behaviour*.

Standard HMI Controls

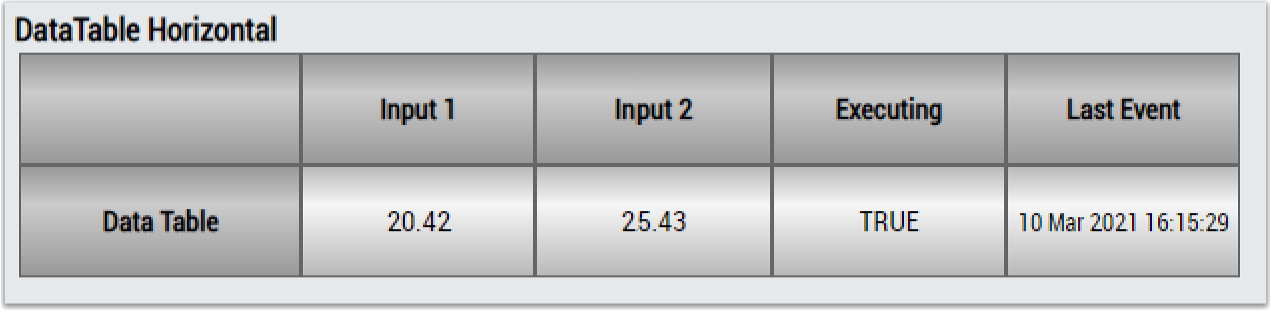
For the RMS algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values: RMS Results.



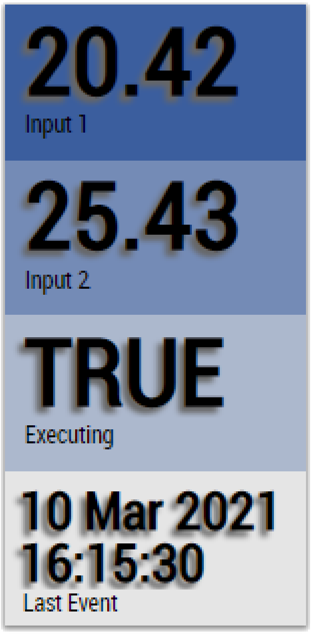
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

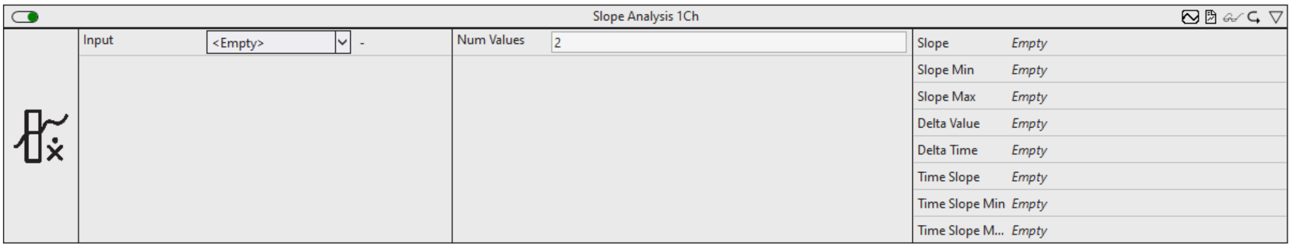
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the RMS algorithm using the Mapping Wizard [▶ 432].

6.4.6.6 Slope Analysis 1Ch



Slope Analysis 1Ch

Input <Empty> -

Num Values 2

Slope	Empty
Slope Min	Empty
Slope Max	Empty
Delta Value	Empty
Delta Time	Empty
Time Slope	Empty
Time Slope Min	Empty
Time Slope M...	Empty

The *Slope Analysis 1Ch* calculates the slope between two values of the input stream. One of those two values is the current input value and the second value is the input value that occurred a defined number (configured by the parameter *Num Values*) of cycles before in the input stream. The difference between these two values is returned as *Delta Value*.

The corresponding distance on the time-coordinate is calculated as the difference of the timestamps of these two values and is provided as the output value *Delta Time.* Note that the value *Delta Time* is displayed in nanoseconds, but for the calculation of the slope it is scaled to a second as base unit.

The *Slope* is then calculated as the fraction of *Delta Value* and *Delta Time* (scaled to seconds) and estimates the gradient for the timestamp in the center of the two timestamps used in the calculation of *Delta Time*. This is the value returned as *Time Slope* if it corresponds to a timestamp of the input stream. For configurations, where *Num Values* is an uneven number there is no input value matching the exact centre timestamp. In this case the timestamp of the value that directly succeeded the calculated centre timestamp is returned as *Time Slope.*

Further, the algorithm provides the minimal slope, the maximal slope and the time values of minimum and maximum.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

Num Values: Number of cycles that are in between the values used for the calculation of the slope.

Output values

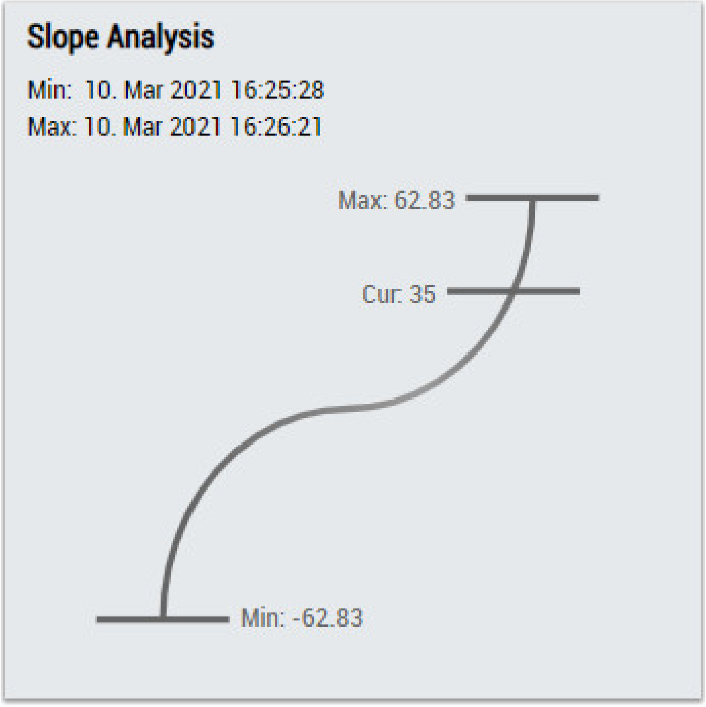
- **Slope:** Indicates the current slope value**.**

- **Slope Min:** Indicates the minimum of the slope values.
- **Slope Max:** Indicates the maximum of the slope values.
- **Delta Value:** Indicates the difference between the two values used to calculate the latest slope.
- **Delta Time:** Indicates the time period used to calculate the latest slope.
- **Time Slope:** Indicates the time value of the latest slope value.
- **Time Slope Min:** Indicates the time value of the minimum of the slope → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **Time Slope Max:** Indicates the time value of the maximum of the slope → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the *Slope Analysis 1Ch* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SlopeAnalysis control visualizes the output values Slope, Slope Min, Slope Max, Time Slope Min and Time Slope Max.

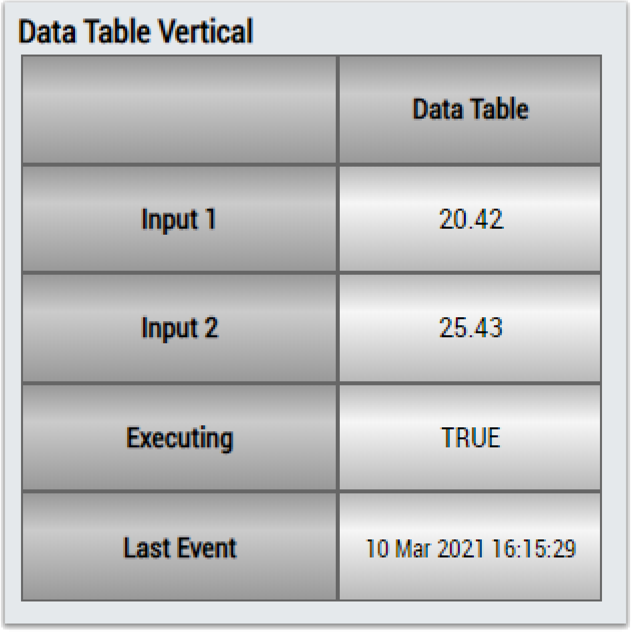


Slope Analysis

Min: 10. Mar 2021 16:25:28

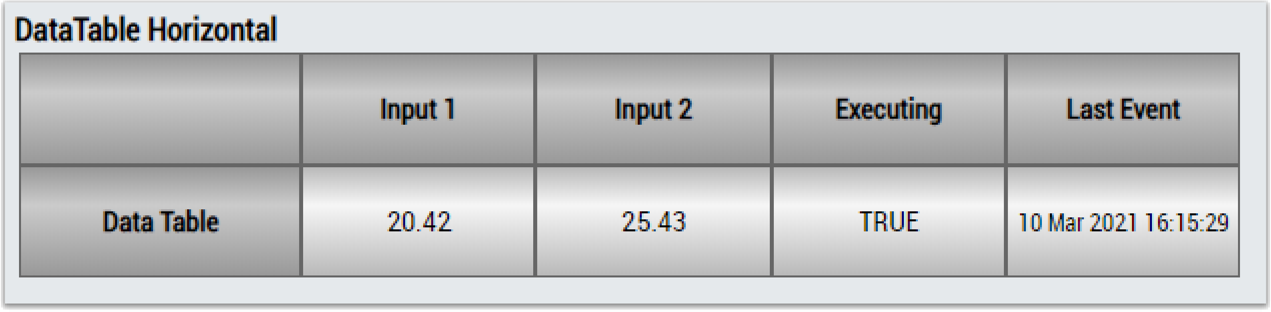
Max: 10. Mar 2021 16:26:21

2. The Table Control or Multivalue Control visualizes all output values: Slope, Slope Min, Slope Max, Delta Value, Delta Time, Time Slope, Time Slope Min, Time Slope Max.



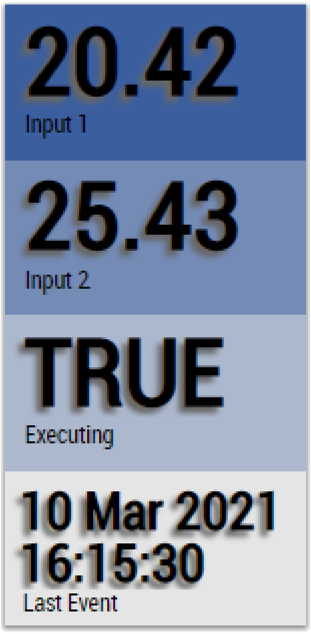
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

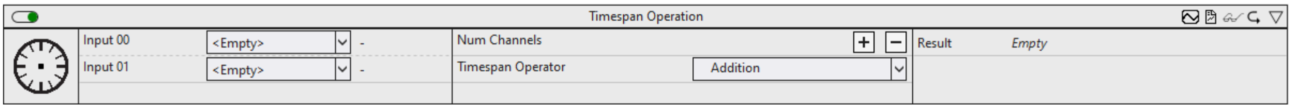
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the algorithm using the Mapping Wizard [▸ 432].

6.4.6.7 Timespan operation



Timespan Operation

Input 00 <Empty> -
Input 01 <Empty> -

Num Channels
Timespan Operator Addition

Result Empty

The *Timespan Operation* performs a mathematical operation on two or more different input channels and returns the result of the mathematical operation. Timespans with a resolution of 1 ns are supported as inputs.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of operands that are inputs to the function.
- **Timespan Operator:** Mathematical operator of the operation (*addition* / *subtraction*).

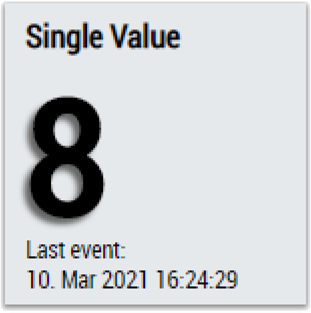
Output values

- **Result:** Result of the mathematical operation as timespan.

Standard HMI Controls

For the Timespan Operation algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The SingleValue control visualizes the output value Result.

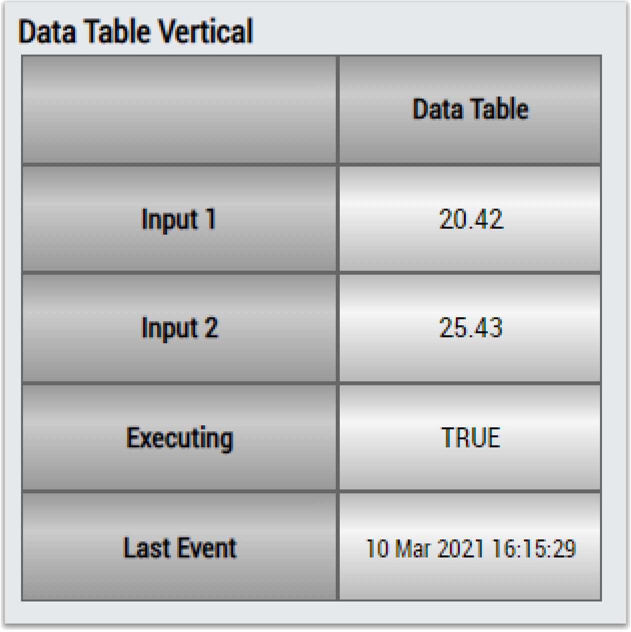


Single Value

8

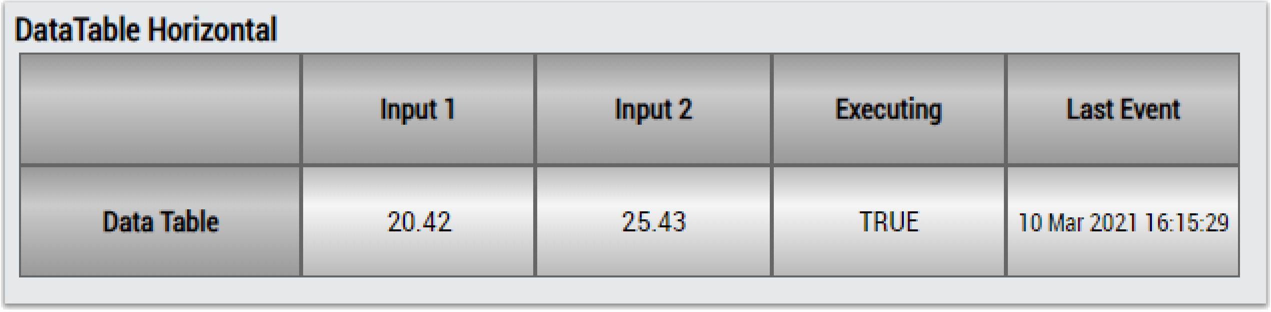
Last event:
10. Mar 2021 16:24:29

2. The Table Control or Multivalue Control visualizes all output values: Result.



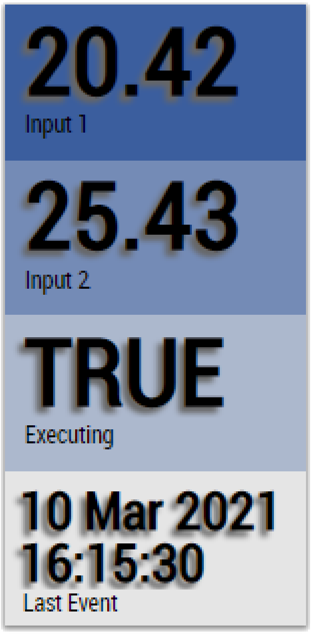
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

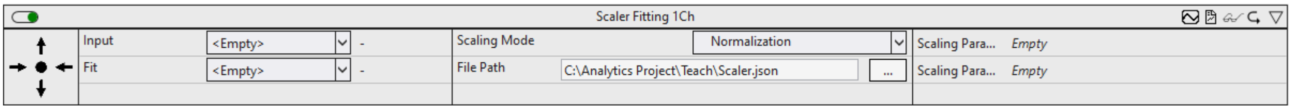
TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Timespan Operation algorithm using the Mapping Wizard [▶ 432].

6.4.7 Analytics - Preprocessing

6.4.7.1 Scaler Fitting



Scaler Fitting 1Ch

Input	<Empty>	-	Scaling Mode	Normalization	Scaling Para...	Empty
Fit	<Empty>	-	File Path	C:\Analytics Project\Teach\Scaler.json ...	Scaling Para...	Empty

The *Scaler Fitting* module can be used to determine the parameters for feature scaling and save them in a file. These parameters can then be used in the *Scaler* module.

The parameters required for scaling are determined depending on the operation mode.

Normalization:

$$x' = \frac{x - low(x)}{high(x) - low(x)}$$

Standardization:

$$x' = \frac{x - \bar{x}}{\sigma}$$

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

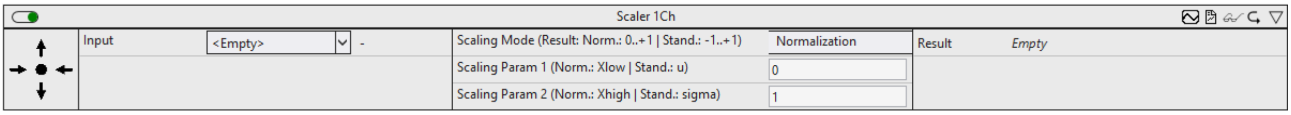
Configuration options

- **Scaling Mode:** Mode for scaling.
  Normalization
  Standardization
- **File Path:** Specifies the file path for the file in which the scaling parameters are saved. File type: *json*.

Output values

- **Scaling Param 1:** The value of the first scaling parameter depends on the scaling mode set.
  Normalization: Smallest value of the input signal
  Standardization: Average value of the input signal
- **Scaling Param 2:** The value of the second scaling parameter depends on the scaling mode set.
  Normalization: Largest value of the input signal
  Standardization: Standard deviation of the input signal

6.4.7.2 Scaler



Scaler 1Ch

Input	<Empty> -	Scaling Mode (Result: Norm.: 0..+1 | Stand.: -1..+1)	Normalization	Result *Empty*
		Scaling Param 1 (Norm.: Xlow | Stand.: u)	0	
		Scaling Param 2 (Norm.: Xhigh | Stand.: sigma)	1	

The *Scaler* module can be used to perform feature scaling.

Depending on the operation mode, the scaling parameters are used in different ways.

Normalization

$$x' = \frac{x - low(x)}{high(x) - low(x)}$$

Standardization:

$$x' = \frac{x - \bar{x}}{\sigma}$$

The scaling parameters can be configured directly. Alternatively, they can be imported from a file.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Scaling Mode:** Mode for scaling.
  Normalization: Result has a value range of 0..+1
  Standardization: Result has a value range of -1..+1
- **Scaling Param 1:** The value of the first scaling parameter depends on the scaling mode set.
  Normalization: Smallest value of the input signal
  Standardization: Average value of the input signal
- **Scaling Param 2:** The value of the second scaling parameter depends on the scaling mode set.
  Normalization: Largest value of the input signal
  Standardization: Standard deviation of the input signal
- **File Path (optional):** File path to a file with previously determined parameters. The file is created by the *Scaler Fitting* module. File type: *json*.

Output values

- **Result:** Outputs the calculated value. This is calculated from the two scaling parameters depending on the scaling mode.
  Normalization: Result has a value range of 0..+1
  Standardization: Result has a value range of -1..+1

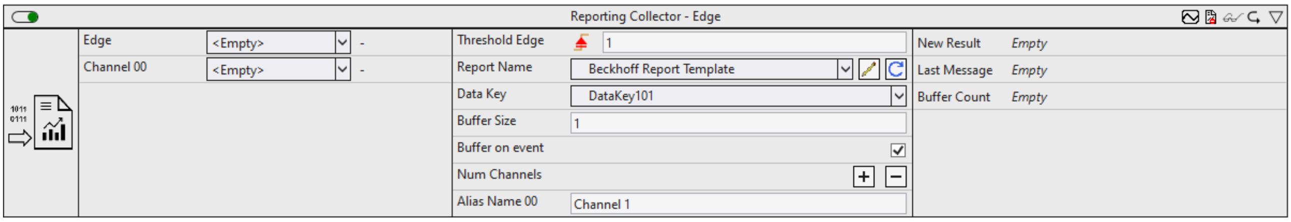
6.4.8 Analytics - Reporting

24/7 reporting can be implemented in TwinCAT Analytics using the algorithms in the *Reporting* category. The reporting collectors collect the data and send it to the reporting server. The reporting triggers trigger the creation of a report.

6.4.8.1 Reporting Collector

The Reporting Collectors collect data and send it to the Reporting Server in a data message after an event.

6.4.8.1.1 Reporting Collector Edge



Reporting Collector - Edge

Edge <Empty> -
Channel 00 <Empty> -

Threshold Edge 1
Report Name Beckhoff Report Template
Data Key DataKey101
Buffer Size 1
Buffer on event ☑
Num Channels
Alias Name 00 Channel 1

New Result *Empty*
Last Message *Empty*
Buffer Count *Empty*

The Reporting Collector Edge collects the data from the input channels and sends the data to the Reporting Server after an event or after the buffer is filled, depending on the configuration. An event is triggered when the signal of the input channel passes the configured edge at a certain threshold.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

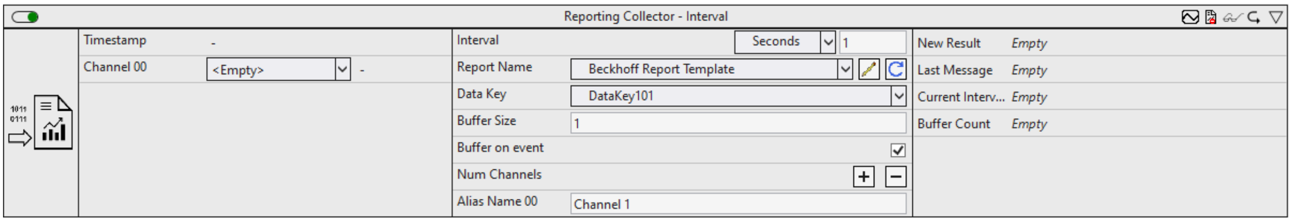
Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.
- **Data Key:** the data key should be unique within a report. The data object can be uniquely assigned to the report via the data key.
- **Buffer Size:** specifies the size of the buffered data. If the buffer size is equal to one, a key-value structure is built. If the buffer size is greater than one, the data is presented in a table.
- **Buffer on event:** specifies how the data is to be collected and buffered.
  If the parameter is True, the data of the inputs is buffered at each edge signal at the input. As soon as the Buffer Count output has the same value as the Buffer Size parameter, the data is sent to the TcReportingServer.
  If the parameter is False, the data of the inputs are buffered in the buffer at each cycle. Once the buffer size is reached, the new data replaces the previous data. If there is an edge signal at the input, the data is sent to the TcReportingServer.
- **Num Channels:** the number of input channels from which the data will be collected. And the number of parameters to allow a description of the input channels.
- **Alias Name:** serves as a description of the input channel. BufferSize = 1: the alias name serves as a key in the key-value structure. BufferSize > 1: the alias name serves as the heading of the table column.
- **Include Timestamps (optional):** inserts a column with the timestamps**.**

Output values

- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Buffer Count:** specifies the number of elements in the buffer.

6.4.8.1.2 Reporting Collector Interval



Reporting Collector - Interval

Timestamp -
Channel 00 <Empty> -

Interval: Seconds 1
Report Name: Beckhoff Report Template
Data Key: DataKey101
Buffer Size: 1
Buffer on event ☑
Num Channels + −
Alias Name 00: Channel 1

New Result *Empty*
Last Message *Empty*
Current Interv... *Empty*
Buffer Count *Empty*

The Reporting Collector Interval collects the data from the input channels and sends the data to the Reporting Server after an event or after the buffer is filled, depending on the configuration. An event is triggered when the timespan of the configured interval has expired.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

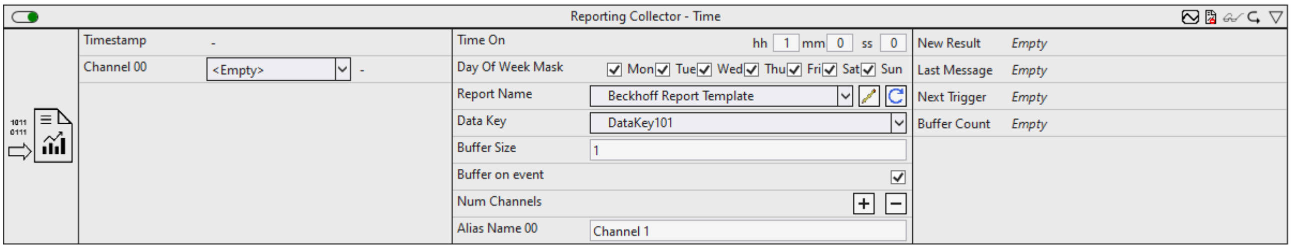
Configuration options

- **Interval:** time interval at which the data should be sent to the Reporting Server.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.
- **Data Key:** the data key should be unique within a report. The data object can be uniquely assigned to the report via the data key.
- **Buffer Size:** specifies the size of the buffered data. If the buffer size is equal to one, a key-value structure is built. If the buffer size is greater than one, the data is presented in a table.
- **Buffer on event:** specifies how the data is to be collected and buffered.
  If the parameter is True, the data of the inputs is buffered at each edge signal at the input. As soon as the Buffer Count output has the same value as the Buffer Size parameter, the data is sent to the TcReportingServer.
  If the parameter is False, the data of the inputs are buffered in the buffer at each cycle. Once the buffer size is reached, the new data replaces the previous data. If there is an edge signal at the input, the data is sent to the TcReportingServer.
- **Num Channels:** the number of input channels from which the data will be collected. And the number of parameters to allow a description of the input channels.
- **Alias Name:** serves as a description of the input channel. BufferSize = 1: the alias name serves as a key in the key-value structure. BufferSize > 1: the alias name serves as the heading of the table column.
- **Include Timestamps (optional):** inserts a column with the timestamps**.**

Output values

- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Current Interval Time:** indicates the amount of time that has already elapsed from the current interval.
- **Buffer Count:** specifies the number of elements in the buffer.

6.4.8.1.3 Reporting Collector Time



Reporting Collector - Time

Timestamp -
Channel 00 <Empty> -

Time On: hh 1 mm 0 ss 0
Day Of Week Mask: ☑ Mon ☑ Tue ☑ Wed ☑ Thu ☑ Fri ☑ Sat ☑ Sun
Report Name: Beckhoff Report Template
Data Key: DataKey101
Buffer Size: 1
Buffer on event ☑
Num Channels + −
Alias Name 00: Channel 1

New Result *Empty*
Last Message *Empty*
Next Trigger *Empty*
Buffer Count *Empty*

The Reporting Collector Time collects the data from the input channels and sends the data to the Reporting Server after an event or after the buffer is filled, depending on the configuration. An event is triggered when the configured switch-on time is reached. The switch-on time and the days of the weeks can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Time On:** switch-on time.
- **Day of Week Mask:** weekdays on which the timer should be active.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.
- **Data Key:** the data key should be unique within a report. The data object can be uniquely assigned to the report via the data key.
- **Buffer Size:** specifies the size of the buffered data. If the buffer size is equal to one, a key-value structure is built. If the buffer size is greater than one, the data is presented in a table.
- **Buffer on event:** specifies how the data is to be collected and buffered.
  If the parameter is True, the data of the inputs is buffered at each edge signal at the input. As soon as the Buffer Count output has the same value as the Buffer Size parameter, the data is sent to the TcReportingServer.
  If the parameter is False, the data of the inputs are buffered in the buffer at each cycle. Once the buffer size is reached, the new data replaces the previous data. If there is an edge signal at the input, the data is sent to the TcReportingServer.
- **Num Channels:** the number of input channels from which the data will be collected. And the number of parameters to allow a description of the input channels.
- **Alias Name:** serves as a description of the input channel. BufferSize = 1: the alias name serves as a key in the key-value structure. BufferSize > 1: the alias name serves as the heading of the table column.
- **Include Timestamps (optional):** inserts a column with the timestamps**.**

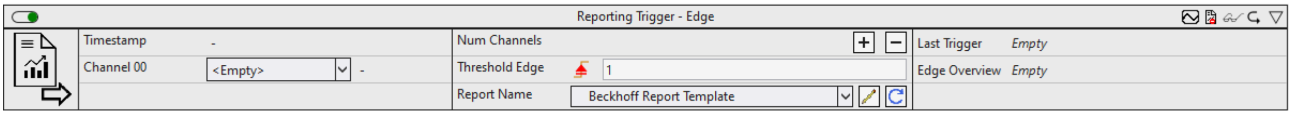
Output values

- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Next Message:** indicates the remaining time until the next message.
- **Buffer Count:** specifies the number of elements in the buffer.

6.4.8.2 Reporting Trigger

The Reporting Triggers send a trigger message to the Reporting Server after an event and thus trigger the creation of a report.

6.4.8.2.1 Reporting Trigger Edge



Reporting Trigger - Edge

Timestamp	-	Num Channels	+ −	Last Trigger	*Empty*
Channel 00	<Empty> -	Threshold Edge	1	Edge Overview	*Empty*
		Report Name	Beckhoff Report Template		

The Reporting Trigger Edge triggers the creation of a report after an event is triggered. An event is triggered when the input channel signal exceeds the configured edge at a specified threshold. Internally, the inputs that were once True remain True. The inputs are only reset to False as soon as all inputs were True at least once. This allows the output bNewResult to be used as one input by multiple Reporting Collectors and once all Reporting Collectors have sent a data message, a trigger message is sent.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

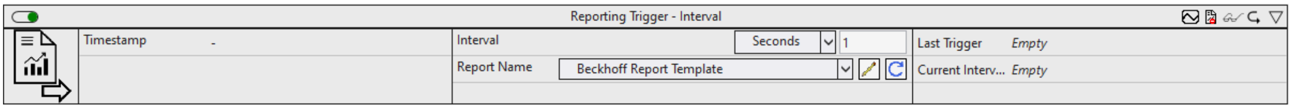
- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.

- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.

Output values

- **Last Trigger:** indicates when the last message was sent to the Reporting Server.
- **Edge Overview:** indicates which input channels were True at least once.

6.4.8.2.2 Reporting Trigger Interval



Reporting Trigger - Interval

Timestamp -

Interval Seconds 1

Report Name Beckhoff Report Template

Last Trigger *Empty*

Current Interv... *Empty*

The Reporting Trigger Interval triggers the creation of a report after an event has been triggered. An event is triggered when the timespan of the configured interval has expired.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

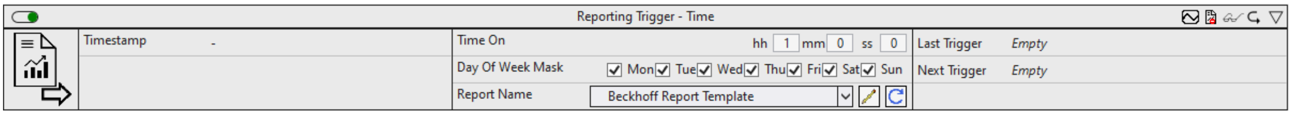
Configuration options

- **Interval:** time interval at which the data should be sent to the Reporting Server.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.

Output values

- **Last Trigger:** indicates when the last message was sent to the Reporting Server.
- **Current Interval Time:** indicates the amount of time that has already elapsed from the current interval.

6.4.8.2.3 Reporting Trigger Time



Reporting Trigger - Time

Timestamp -

Time On hh 1 mm 0 ss 0

Day Of Week Mask ☑ Mon ☑ Tue ☑ Wed ☑ Thu ☑ Fri ☑ Sat ☑ Sun

Report Name Beckhoff Report Template

Last Trigger *Empty*

Next Trigger *Empty*

The Reporting Trigger Time triggers the creation of a report after an event has been triggered. An event is triggered when the configured switch-on time is reached. The switch-on time and the days of the weeks can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Time On:** switch-on time.
- **Day of Week Mask:** weekdays on which the timer should be active.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.

Output values

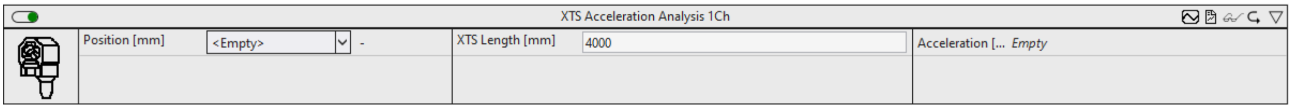
- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Next Trigger:** indicates the remaining time until the next message.

6.4.9 Analytics - Specific

6.4.9.1 XTS

The algorithms of the category *Analytics-XTS* provide special functionalities for the Beckhoff XTS system. For example analysis of distance, velocity and acceleration.

6.4.9.1.1 XTS Acceleration Analysis 1Ch



XTS Acceleration Analysis 1Ch

Position [mm] <Empty> -

XTS Length [mm] 4000

Acceleration [... *Empty*

The *XTS Acceleration Analysis 1Ch* calculates the current acceleration of a XTS mover. For this purpose, the length of the XTS in millimeters must be declared and as input signal the mover position is required.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **XTS Length [mm]:** Length of the given XTS system in millimeters.

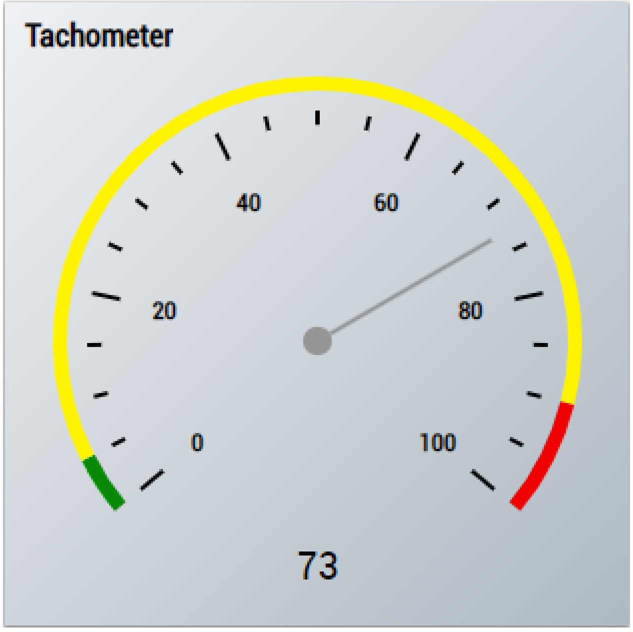
Output Values

- **Acceleration:** Current acceleration of the XTS mover.

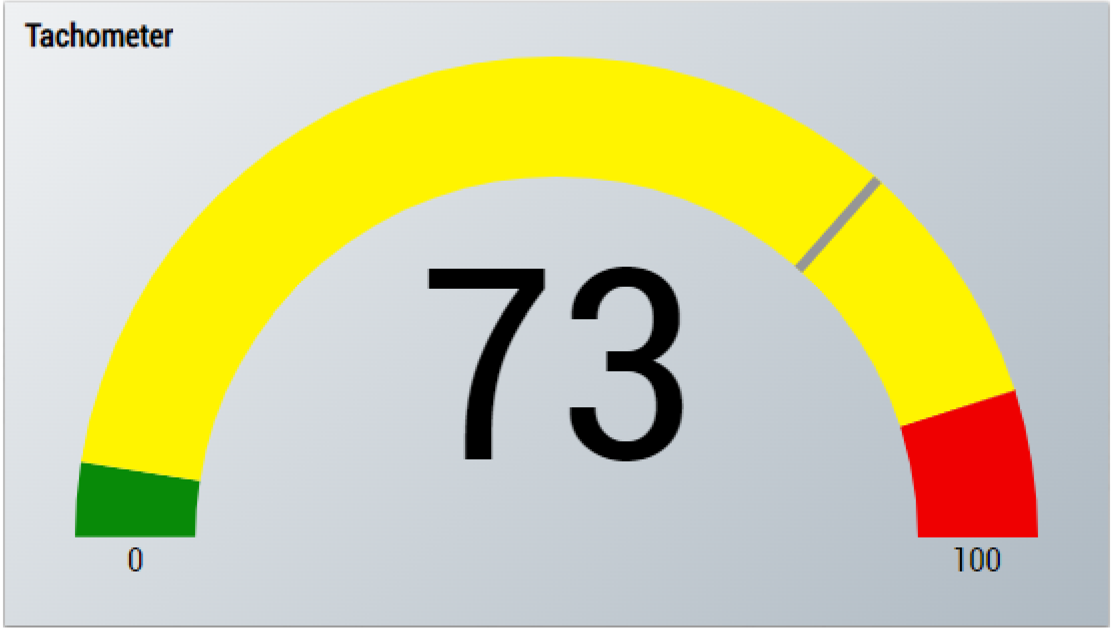
Standard HMI Controls

For the XTS Acceleration Analysis 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

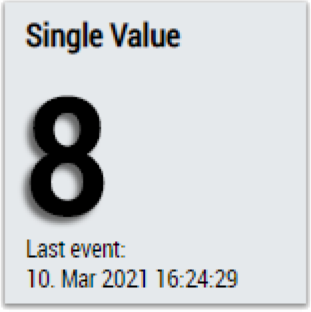
1. The Tachometer control visualizes the output value Acceleration.



2. The Radial Gauge control visualizes the output value Acceleration.



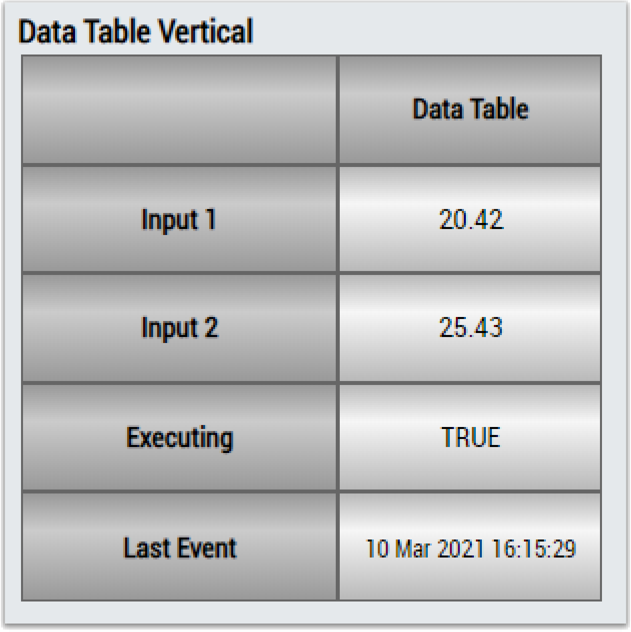
3. The SingleValue control visualizes the output value Acceleration.


Single Value

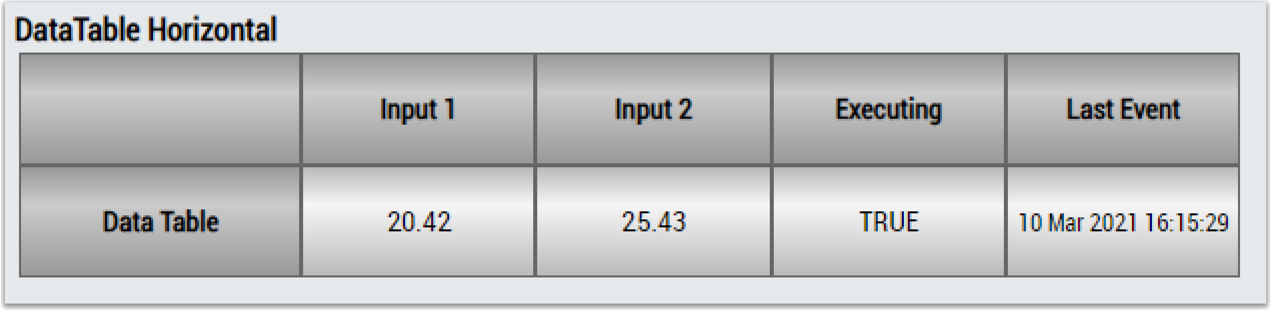
8

Last event:
10. Mar 2021 16:24:29

4. The Table Control or Multivalue Control visualizes all output values: Acceleration.

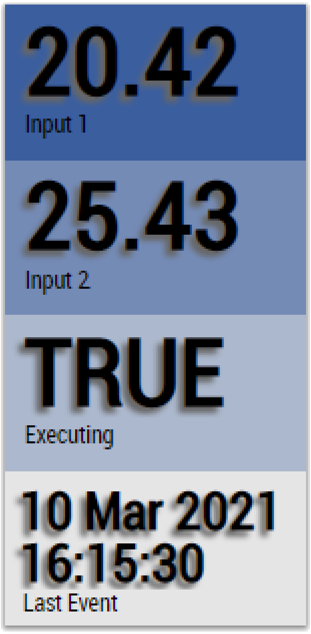

Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

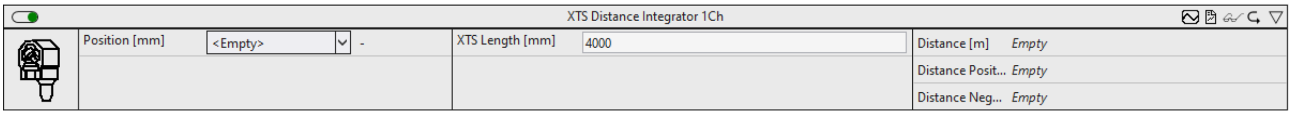
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the XTS Acceleration Analysis 1Ch algorithm using the Mapping Wizard [▸ 432].

6.4.9.1.2 XTS Distance Integrator 1Ch



XTS Distance Integrator 1Ch

Position [mm] <Empty> -

XTS Length [mm] 4000

Distance [m] Empty

Distance Posit... Empty

Distance Neg... Empty

The *XTS Distance Integrator 1Ch* calculates the distance covered by a XTS mover. The algorithm provides the total distance, the positive distance and the negative distance. For this purpose, the length of the XTS in millimeters must be declared and as input signal the mover position is required.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **XTS Length [mm]:** Length of the given XTS system in millimeters.

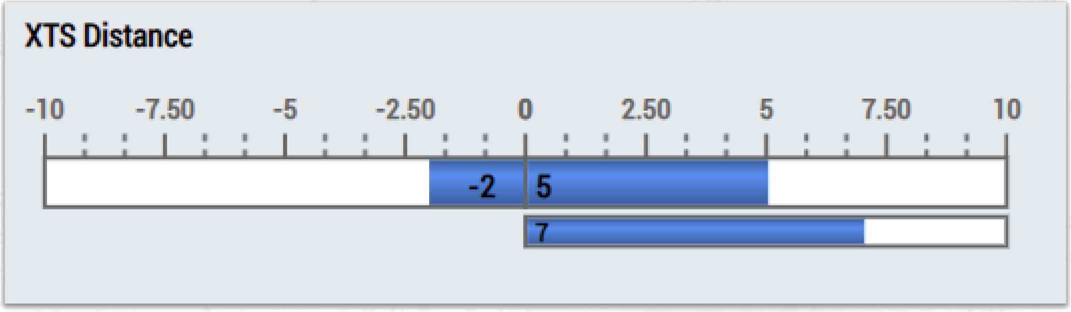
Output Values

- **Distance:** Total distance the XTS mover has covered.
- **Distance Positive:** Positive distance the XTS mover has covered (direction: forward).
- **Distance Negative:** Negative distance the XTS mover has covered (direction: backward).

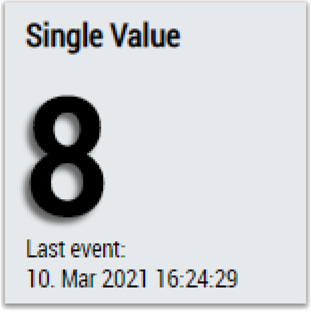
Standard HMI Controls

For the XTS Distance Integrator 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The XTSDistance control visualizes the output values Distance, Distance Positive and Distance Negative.



2. The SingleValue control visualizes the output value Distance.

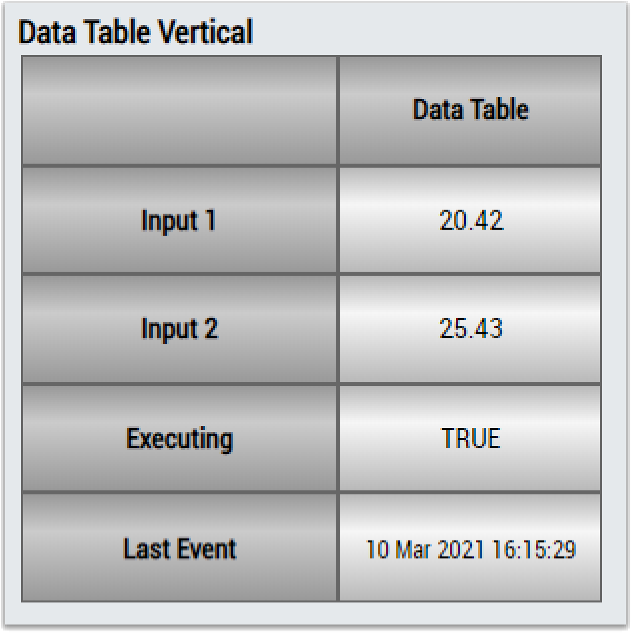


Single Value

8

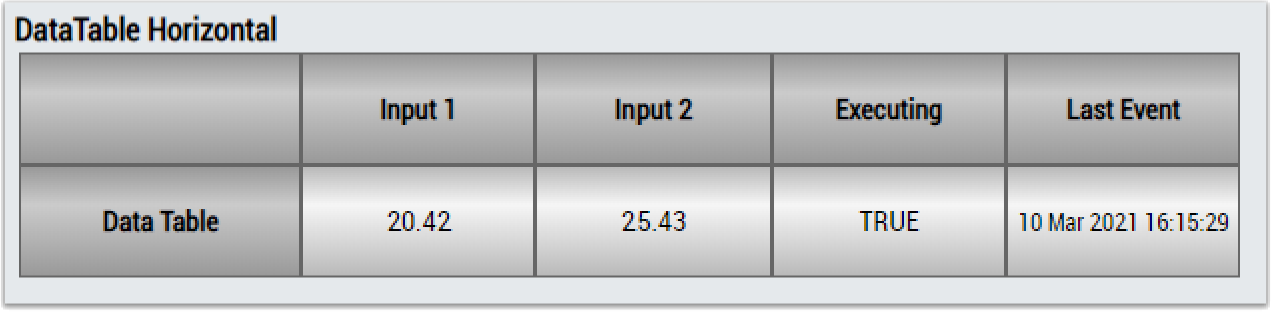
Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Output values Distance, Distance Positive, Distance Negative.



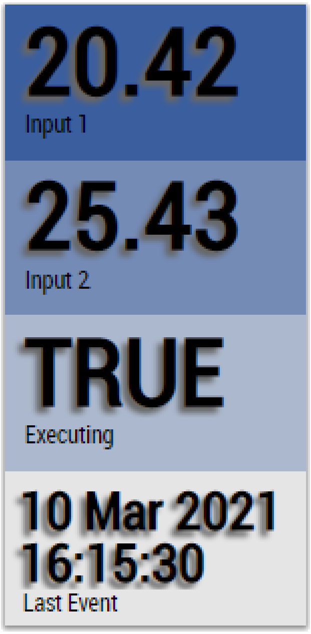
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

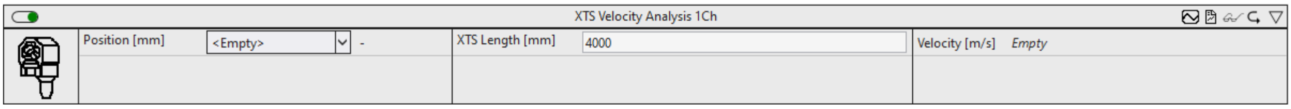
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the XTS Distance Integrator 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.9.1.3 XTS Velocity Analysis 1Ch



XTS Velocity Analysis 1Ch

Position [mm] <Empty> -

XTS Length [mm] 4000

Velocity [m/s] Empty

The *XTS Velocity Analysis 1Ch* calculates the current velocity of a XTS mover. For this purpose, the length of the XTS in millimeters must be declared and as input signal the mover position is required.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **XTS Length [mm]:** Length of the given XTS system in millimeters.

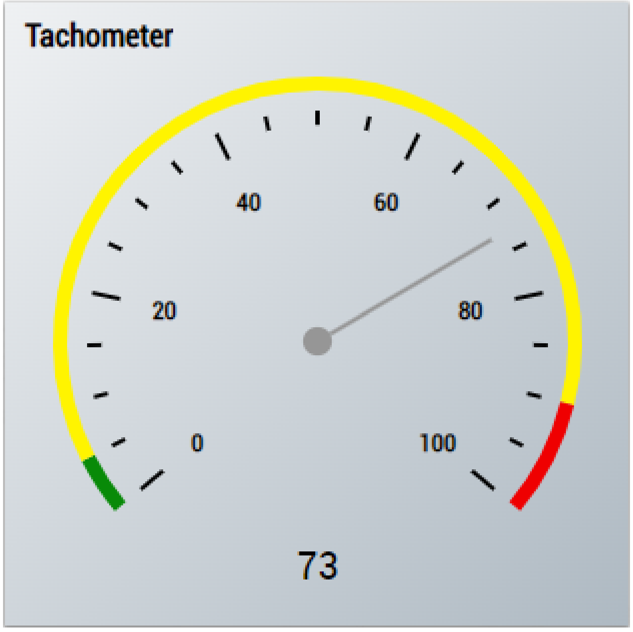
Output Values

- **Velocity:** Current velocity of the XTS mover.

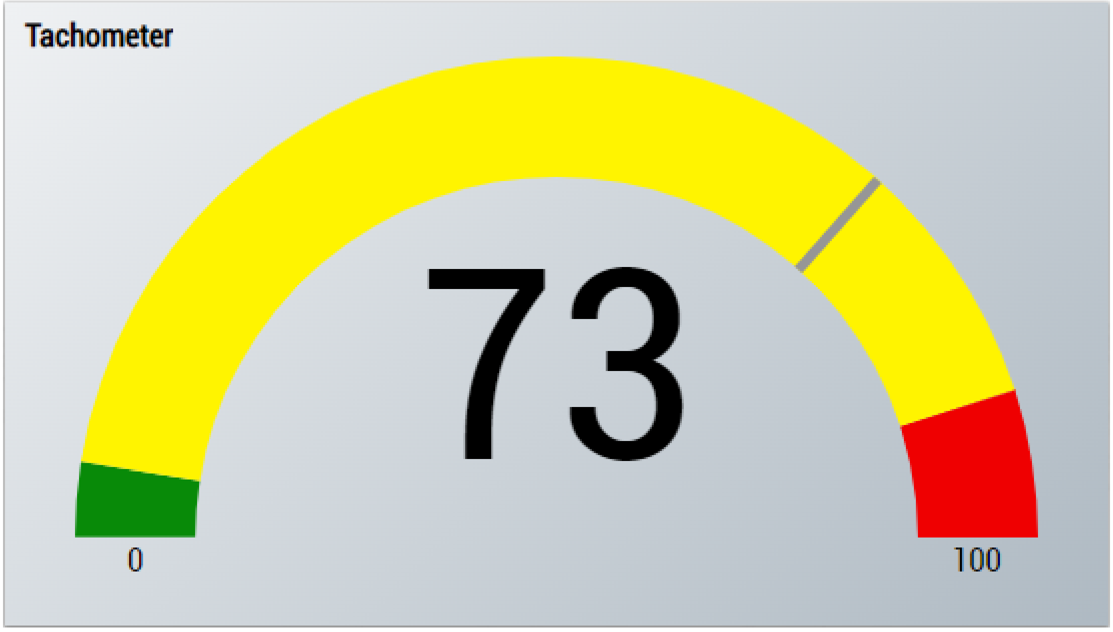
Standard HMI Controls

For the XTS Velocity Analysis 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

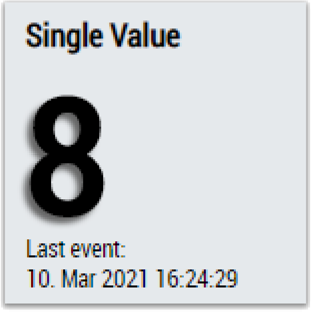
1. The Tachometer control visualizes the output value Velocity.



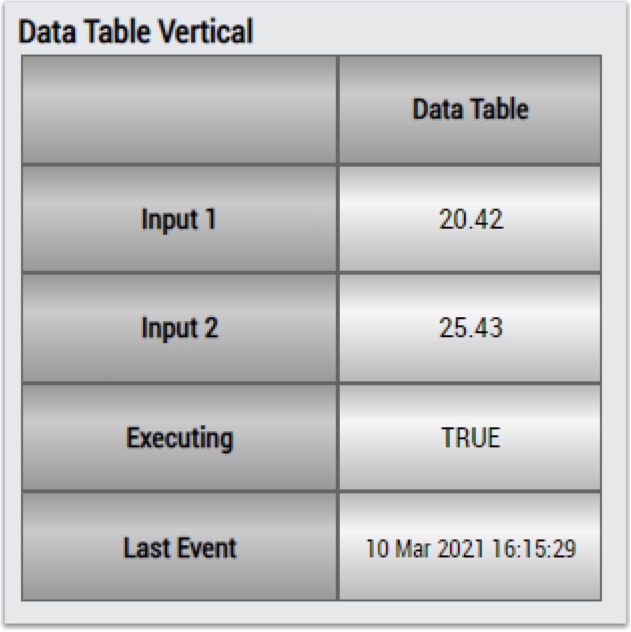
2. The Radial Gauge control visualizes the output value Velocity.



3. The SingleValue control visualizes the output value Velocity.

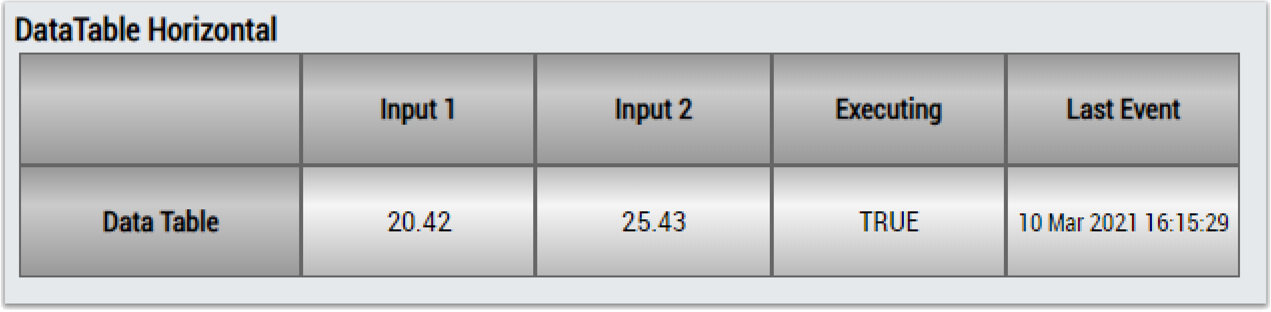


4. The Table Control or Multivalue Control visualizes all output values: Velocity.



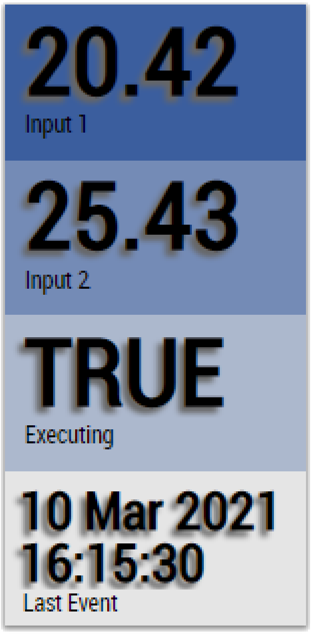
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

TRUE
Executing

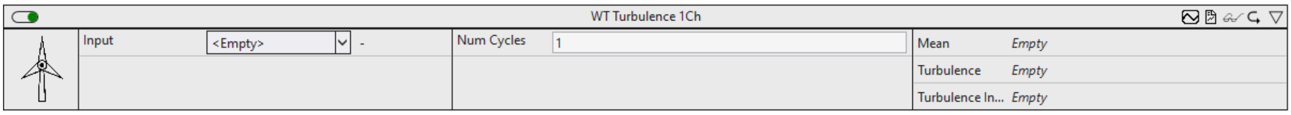
10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the XTS Velocity Analysis 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.9.2 Wind Turbine

The algorithms of the category *Analytics – WT* provide special functionalities for the wind technology industry. For example analysis of mean wind speed, turbulence and turbulence intensity.

6.4.9.2.1 WT Turbulence 1Ch



WT Turbulence 1Ch

Input	<Empty>	Num Cycles	1

Mean	*Empty*
Turbulence	*Empty*
Turbulence In...	*Empty*

The *WT Turbulence 1Ch* calculates the mean of the wind velocity, the turbulence, and the turbulence intensity according to the standard *EN 61400-1*. As input signal, the wind velocity is required. The output values are updated in a cycle of 10 minutes.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Num Cycles 10Min:** Indicates the number of cycles that fit in the time interval for the calculation, according to EN 61400-1 this is a ten minutes interval.

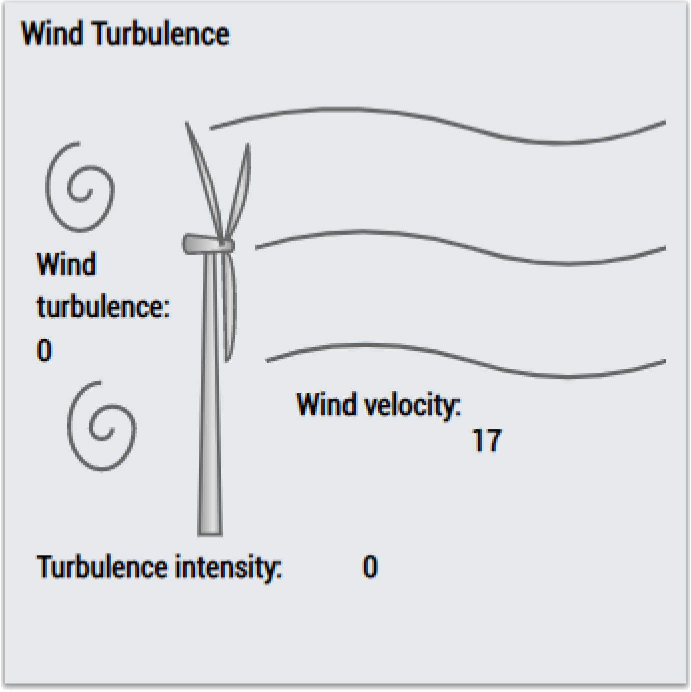
Output Values

- **Mean:** Mean of the wind velocity.
- **Turbulence:** Turbulence of the wind. According to EN-standard this is the standard deviation of the wind velocity over a time interval of 10 minutes.
- **Turbulence Intensity:** Intensity of the wind turbulence.

Standard HMI Controls

For the WT Turbulence 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The WindTurbulence control visualizes the output values Mean, Turbulence and Turbulence Intensity.



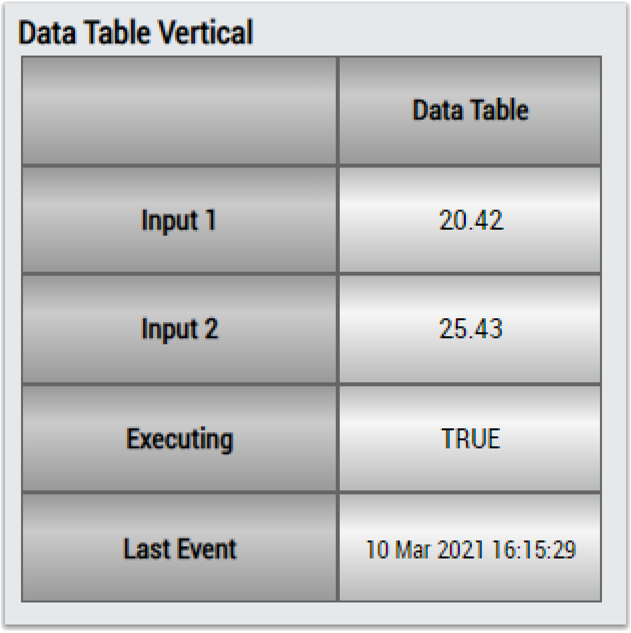
Wind Turbulence

Wind turbulence: 0

Wind velocity: 17

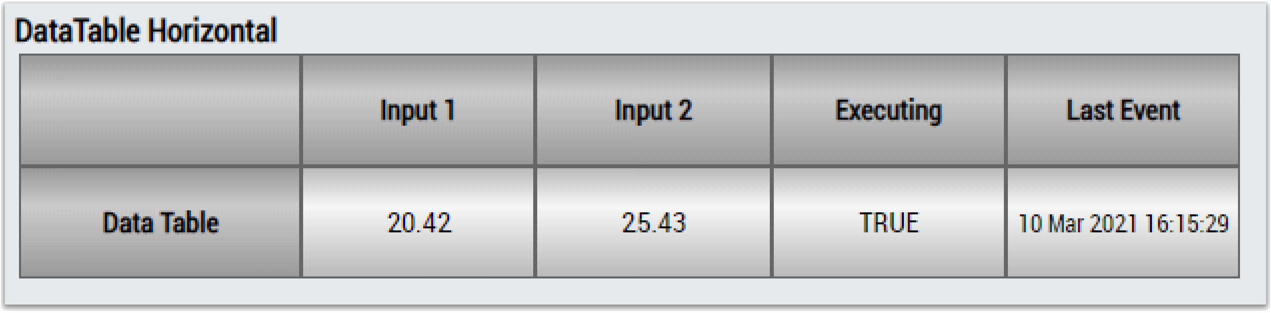
Turbulence intensity: 0

2. The Table Control or Multivalue Control visualizes all output values: Mean, Turbulence, Turbulence Intensity.



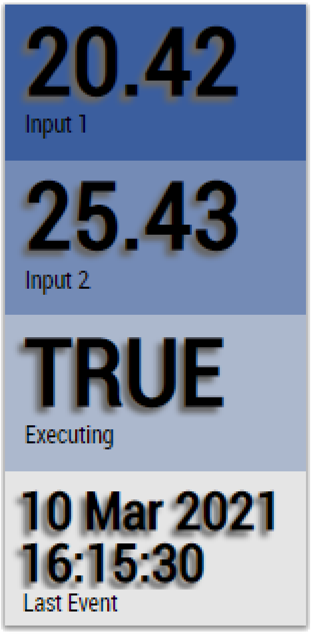
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

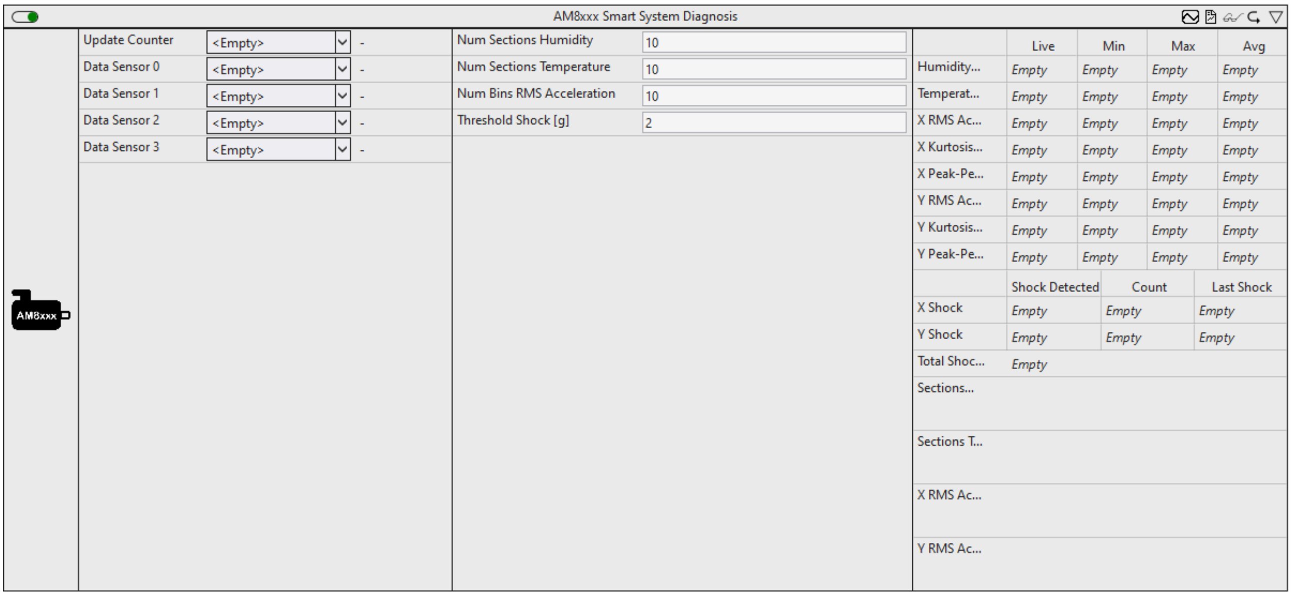
TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the WT Turbulence 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.9.3 AM

6.4.9.3.1 AM8xxx Smart System Diagnosis



AM8xxx Smart System Diagnosis

Update Counter	<Empty>	-
Data Sensor 0	<Empty>	-
Data Sensor 1	<Empty>	-
Data Sensor 2	<Empty>	-
Data Sensor 3	<Empty>	-

Num Sections Humidity	10
Num Sections Temperature	10
Num Bins RMS Acceleration	10
Threshold Shock [g]	2

	Live	Min	Max	Avg
Humidity...	Empty	Empty	Empty	Empty
Temperat...	Empty	Empty	Empty	Empty
X RMS Ac...	Empty	Empty	Empty	Empty
X Kurtosis...	Empty	Empty	Empty	Empty
X Peak-Pe...	Empty	Empty	Empty	Empty
Y RMS Ac...	Empty	Empty	Empty	Empty
Y Kurtosis...	Empty	Empty	Empty	Empty
Y Peak-Pe...	Empty	Empty	Empty	Empty

	Shock Detected	Count	Last Shock
X Shock	Empty	Empty	Empty
Y Shock	Empty	Empty	Empty
Total Shoc...	Empty		

Sections...

Sections T...

X RMS Ac...

Y RMS Ac...

The *AM8xxxSSD* enables the analysis of a motor equipped with the Smart System Diagnosis functionality. The built-in encoder records measured values from various internally installed sensors. These include the temperature and humidity in the encoder, as well as the acceleration in the X and Y directions. Various statistical parameters are calculated on the basis of these measured values and threshold monitoring is carried out to detect shocks.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Sections Humidity:** Configure the number of sections for analyzing the humidity in the encoder.
- **Num Sections Temperature:** Configure the number of sections for analyzing the temperature in the encoder.
- **Num Bins RMS Acceleration:** Configure the number of histogram bins for analyzing the acceleration.
- **Threshold Shock:** Threshold value for the RMS of the acceleration for detecting shocks.

Output values

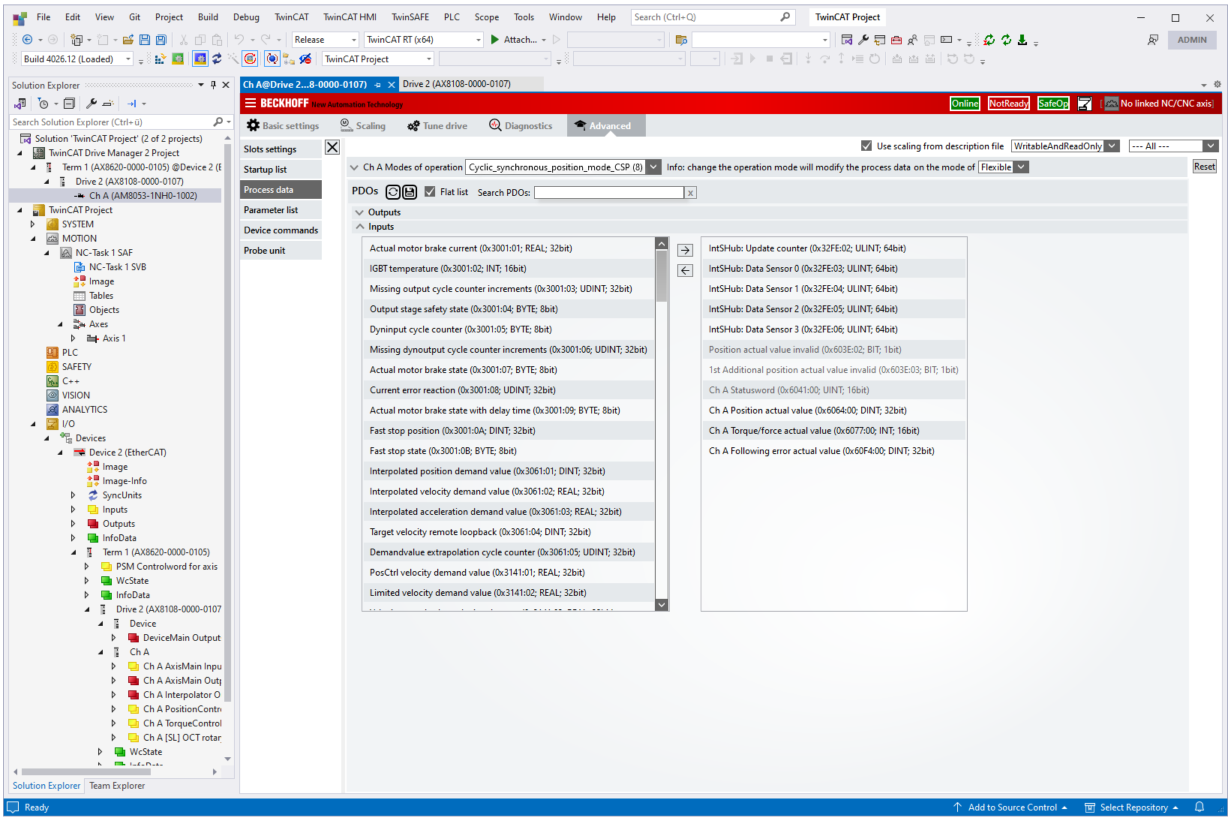
- **Humidity:** Minimum, maximum and average value of the humidity within the encoder in %.
- **Temperature:** Minimum, maximum and average value of the temperature within the encoder in °C.
- **X RMS Acceleration:** Minimum, maximum and average value of the RMS acceleration in the X direction in g.
- **X Kurtosis Acceleration:** Minimum, maximum and average value of the kurtosis of the acceleration in the X direction.
- **X Peak-Peak Acceleration:** Minimum, maximum and average value of the peak-peak value of the acceleration in the X direction in g.
- **YRms Acceleration:** Minimum, maximum and average value of the RMS acceleration in the Y direction in g.
- **Y Kurtosis Acceleration:** Minimum, maximum and average value of the kurtosis of the acceleration in the Y direction.
- **Y Peak-Peak Acceleration:** Minimum, maximum and average value of the peak-peak value of the acceleration in the Y direction in g.
- **X Shock | Shock Detected:** `TRUE`, if a shock is detected in the X direction. The threshold value of the RMS acceleration is monitored for this purpose.

- **X Shock | Count:** Number of shocks in the X direction.
- **X Shock | Last Shock:** Timestamp of the last detected shock in the X direction.
- **Y Shock | Shock Detected:** `TRUE`, if a shock is detected in the Y direction. The threshold value of the RMS acceleration is monitored for this purpose.
- **Y Shock | Count:** Number of shocks in the Y direction.
- **Y Shock | Last Shock:** Timestamp of the last detected shock in the Y direction.
- **Total Shocks:** Number of shocks in the X and Y direction.
- **Sections Humidity:** Temporal distribution of humidity within the encoder. (Section Timer)
- **Sections Temperature:** Temporal distribution of the temperature within the encoder. (Section Timer)
- **X Rms Acceleration:** Distribution of the acceleration in the X direction. (Histogram)
- **Y Rms Acceleration:** Distribution of the acceleration in the Y direction. (Histogram)

Preparation

In order for the sensor values to be available for analysis, they must be added to the process image of the drive. This process is described below.

1. Create a TwinCAT project.
2. Scan the I/Os of the TwinCAT project.
3. Add a Drive Manager project to the solution.
4. Select the corresponding channel of the drive to which the motor with the B/SSD-capable encoder is connected.
5. Select the **Advanced** tab.
6. Further process data can be added in the **Process data** subitem.
7. Add the following process data:
   IntSHub: Update counter
   IntSHub: Data Sensor 0
   IntSHub: Data Sensor 1
   IntSHub: Data Sensor 2
   IntSHub: Data Sensor 3



Solution 'TwinCAT Project' (2 of 2 projects)
TwinCAT Drive Manager 2 Project
Term 1 (AX8620-0000-0105) @Device 2 (E
Drive 2 (AX8108-0000-0107)
Ch A (AM8053-1NH0-1002)
TwinCAT Project
SYSTEM
MOTION
NC-Task 1 SAF
NC-Task 1 SVB
Image
Tables
Objects
Axes
Axis 1
PLC
SAFETY
C++
VISION
ANALYTICS
I/O
Devices
Device 2 (EtherCAT)
Image
Image-Info
SyncUnits
Inputs
Outputs
InfoData
Term 1 (AX8620-0000-0105)
PSM Controlword for axis
WcState
InfoData
Drive 2 (AX8108-0000-0107
Device
DeviceMain Output
Ch A
Ch A AxisMain Inpu
Ch A AxisMain Outp
Ch A Interpolator O
Ch A PositionContr
Ch A TorqueControl
Ch A [SL] OCT rotar
WcState

☑ Use scaling from description file WritableAndReadOnly --- All ---

Ch A Modes of operation Cyclic_synchronous_position_mode_CSP (8) Info: change the operation mode will modify the process data on the mode of Flexible

PDOs ☑ Flat list Search PDOs:

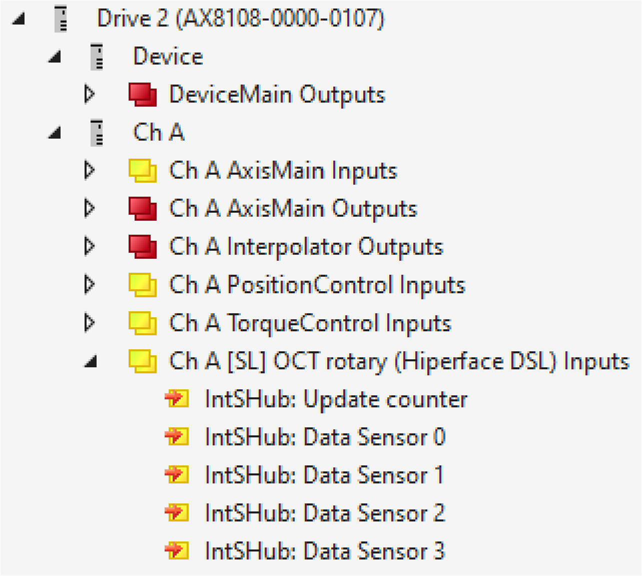
Outputs

Inputs

Actual motor brake current (0x3001:01; REAL; 32bit)
IGBT temperature (0x3001:02; INT; 16bit)
Missing output cycle counter increments (0x3001:03; UDINT; 32bit)
Output stage safety state (0x3001:04; BYTE; 8bit)
Dyninput cycle counter (0x3001:05; BYTE; 8bit)
Missing dynoutput cycle counter increments (0x3001:06; UDINT; 32bit)
Actual motor brake state (0x3001:07; BYTE; 8bit)
Current error reaction (0x3001:08; UDINT; 32bit)
Actual motor brake state with delay time (0x3001:09; BYTE; 8bit)
Fast stop position (0x3001:0A; DINT; 32bit)
Fast stop state (0x3001:0B; BYTE; 8bit)
Interpolated position demand value (0x3061:01; DINT; 32bit)
Interpolated velocity demand value (0x3061:02; REAL; 32bit)
Interpolated acceleration demand value (0x3061:03; REAL; 32bit)
Target velocity remote loopback (0x3061:04; DINT; 32bit)
Demandvalue extrapolation cycle counter (0x3061:05; UDINT; 32bit)
PosCtrl velocity demand value (0x3141:01; REAL; 32bit)
Limited velocity demand value (0x3141:02; REAL; 32bit)

IntSHub: Update counter (0x32FE:02; ULINT; 64bit)
IntSHub: Data Sensor 0 (0x32FE:03; ULINT; 64bit)
IntSHub: Data Sensor 1 (0x32FE:04; ULINT; 64bit)
IntSHub: Data Sensor 2 (0x32FE:05; ULINT; 64bit)
IntSHub: Data Sensor 3 (0x32FE:06; ULINT; 64bit)
Position actual value invalid (0x603E:02; BIT; 1bit)
1st Additional position actual value invalid (0x603E:03; BIT; 1bit)
Ch A Statusword (0x6041:00; UINT; 16bit)
Ch A Position actual value (0x6064:00; DINT; 32bit)
Ch A Torque/force actual value (0x6077:00; INT; 16bit)
Ch A Following error actual value (0x60F4:00; DINT; 32bit)

⇨ Once the process data has been added, it is displayed in the I/O node and is available for analysis.

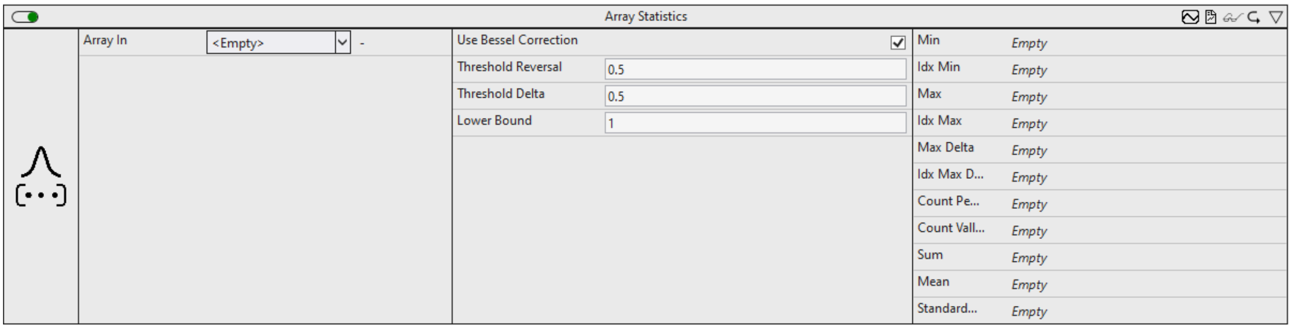


- Drive 2 (AX8108-0000-0107)
  - Device
    - DeviceMain Outputs
  - Ch A
    - Ch A AxisMain Inputs
    - Ch A AxisMain Outputs
    - Ch A Interpolator Outputs
    - Ch A PositionControl Inputs
    - Ch A TorqueControl Inputs
    - Ch A [SL] OCT rotary (Hiperface DSL) Inputs
      - IntSHub: Update counter
      - IntSHub: Data Sensor 0
      - IntSHub: Data Sensor 1
      - IntSHub: Data Sensor 2
      - IntSHub: Data Sensor 3

6.4.10 Analytics - Statistics

The algorithms of the category *Statistics* offer functions for data analysis based on statistical methods. This includes, for example, the calculation of signal correlations and regression analyses.

6.4.10.1 Array Statistics



Array Statistics

Array In	<Empty>	-

Use Bessel Correction	☑
Threshold Reversal	0.5
Threshold Delta	0.5
Lower Bound	1

Min	Empty
Idx Min	Empty
Max	Empty
Idx Max	Empty
Max Delta	Empty
Idx Max D...	Empty
Count Pe...	Empty
Count Vall...	Empty
Sum	Empty
Mean	Empty
Standard...	Empty

The *Array Statistics* algorithm calculates various statistical quantities based on the input array.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Use Bessel Correction:** if the checkbox is activated, Bessel correction will be applied. In order to obtain an expectation-true result for random samples, this parameter must be activated. The parameter is only relevant for the calculation of the standard deviation.

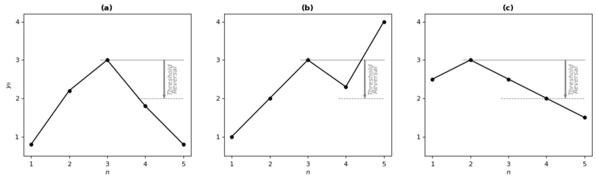
The empirical standard deviation, without Bessel's correction

$$s' = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$$

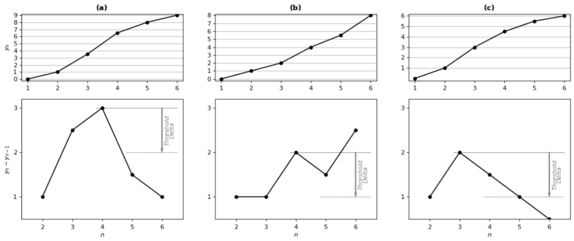
The empirical standard deviation, with Bessel's correction

$$s = \sqrt{\frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$$

- **Threshold Reversal:** threshold for identifying reversals. Reversals are only detected if their difference from the next reversal exceeds the value of Threshold Reversal.
  Below are three examples of peak identification using the parameter *Threshold Reversal*.
  (a) The value y_3 is identified as a peak immediately after processing the value y_4 because the difference between y_3 and y_4 is greater than *Threshold Reversal*.
  (b) The value y_3 is not identified as a peak because the difference between y_3 and y_4 is smaller than *Threshold Reversal* and the curve starts rising again after y_4.
  (c) The value y_2 is identified as a peak after processing the value y_5 because the difference between y_2 and y_5 *exceeds Threshold Reversal*. The value y_2 cannot be identified as a peak beforehand because the difference between y_2 and y_3 (y_4) is less than/equal to *Threshold Reversal* and it is not known whether the values will continue to decrease.



- **Threshold Delta:** threshold for identifying Delta maxima. Maxima of the absolute difference of two successive values (delta) are detected only if the difference between successive deltas exceeds *Threshold Delta*.
  Below are three examples of identifying the Delta maxima with the parameter *Threshold Delta* . The upper diagrams show the original input signals, the lower ones the corresponding delta.
  (a) The value y_4 is identified as a maximum after processing the value y_5 because the difference between the two deltas exceeds *Threshold Delta*.
  (b) No maximum is identified because the difference between the deltas is less than *Threshold Delta*.
  (c) The value y_3 is identified as a maximum after processing the value y_6.



Regardless of *Threshold Delta*, at least one maximum of the Delta between two reversals is detected.

- **Lower Bound:** Lower bound of the array

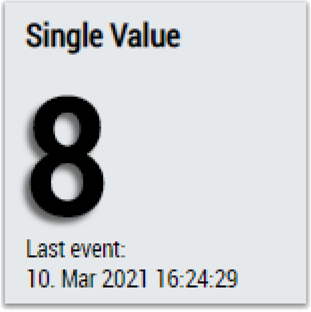
Output values

- **Min:** smallest value in the input array.
- **Idx Min:** array index of *Min*. Indexing starts at 1.
- **Max:** largest value in the input array.
- **Idx Max:** array index of *Max*. The indexing starts at 1.
- **Max Delta:** maximum of the absolute difference between two consecutive values in the input array.
- **Idx Max Delta:** array index of *Max Delta*. The indexing starts at 1.
- **Count Peaks:** total number of peaks identified.
- **Count Valleys:** total number of valleys identified.
- **Sum:** sum over the entire input array.
- **Mean:** mean value over the entire input array.
- **Standard Deviation:** standard deviation over the entire input array.

Standard HMI Controls

For the Array Statistics algorithm, the following HMI controls are available for generating an Analytics Dashboard:

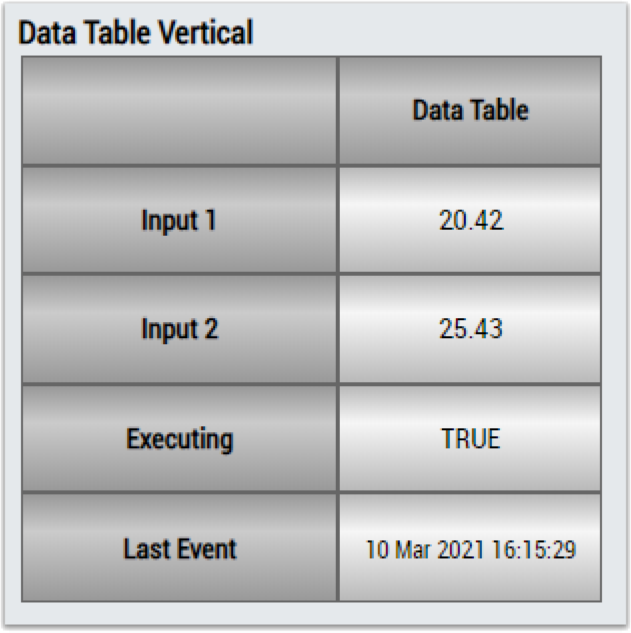
1. The SingleValue control visualizes the Sum output.



2. The SingleValue control visualizes the Mean output.

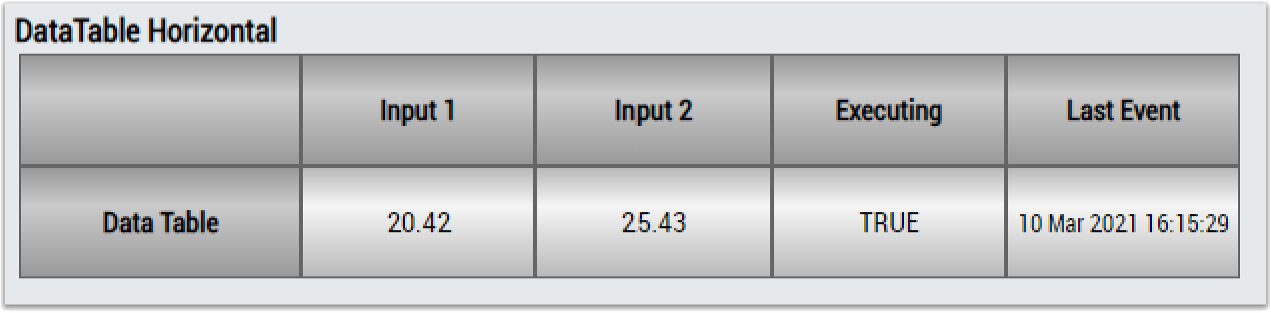
3. The SingleValue control visualizes the Standard Deviation output.

4. The Table Control or Multivalue Control visualizes the output values: Min, Max, Max Delta, CountPeaks Count Valleys, Sum, Mean, Standard Deviation.



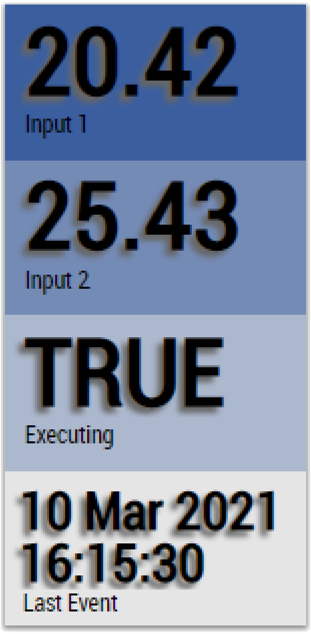
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

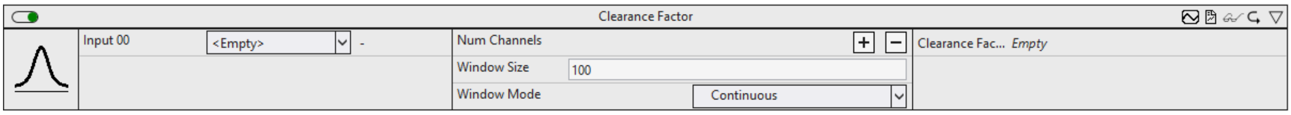
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Standard Deviation algorithm using the Mapping Wizard [▶ 432].

6.4.10.2 Clearance Factor



Clearance Factor

Input 00 <Empty> -

Num Channels

Window Size 100

Window Mode Continuous

Clearance Fac... *Empty*

The algorithm *Clearance Factor* calculates the signal feature of the same name from the input values. The output value is calculated from the ratio of the peak value of the input signal to the squared mean of the square roots of the absolute input signal.

$$\text{Clearance Factor} = \frac{\max(|x|)}{\left(\frac{1}{N}\sum_{n=1}^{N}\sqrt{|x[n]|}\right)^2}$$

The number of input data to be included in the calculation and the type of calculation can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** Configuration of the number of independent input and output channels.
- **Window Mode:** Used window mode. Influences the number of input data that are included in the calculation as well as the time of the calculations.
  Continuous: All input values since the start of the algorithm are included in the calculation. The calculation is cyclic.
  Fix Window: Only the last *N* input values are included in the calculation. The calculation is performed every *N* calls.
  Sliding Window: Only the last *N* input values are included in the calculation. The calculation is performed cyclically.
- **Window Size:** Depending on the window mode used, you can configure the number of values *N* that will be included in the calculation. In the window mode *Continuous* this parameter is ignored.

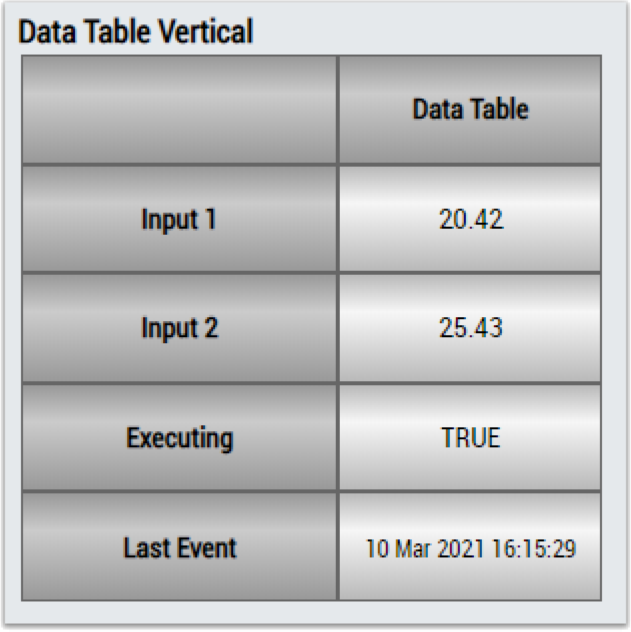
Output values

- **Clearance Factor 00..n:** Clearance factor of the input channels.

Standard HMI Controls

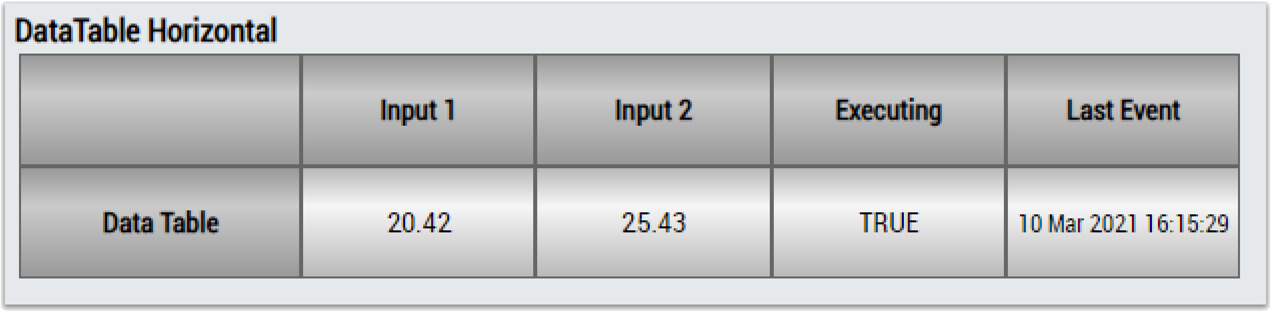
For the *Clearance Factor* algorithm the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values.



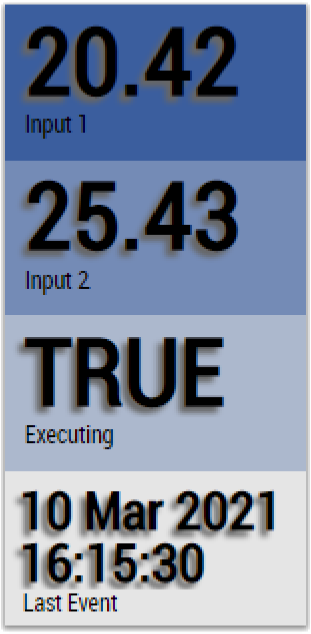
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

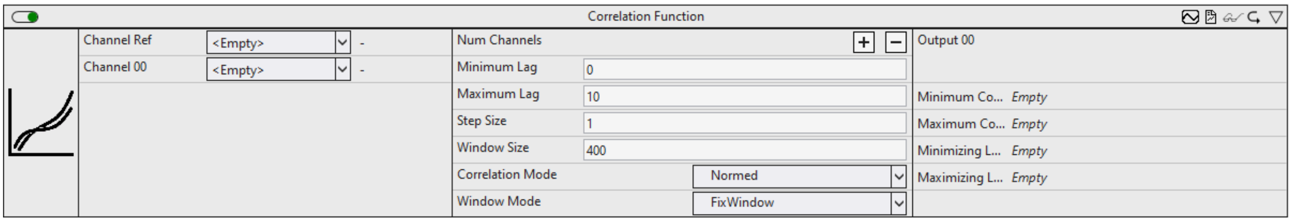
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped using the Mapping Wizard [▸ 432].

6.4.10.3 Correlation Function



Correlation Function

Channel Ref	<Empty>	-
Channel 00	<Empty>	-

Num Channels	
Minimum Lag	0
Maximum Lag	10
Step Size	1
Window Size	400
Correlation Mode	Normed
Window Mode	FixWindow

Output 00

Minimum Co...	Empty
Maximum Co...	Empty
Minimizing L...	Empty
Maximizing L...	Empty

The *Correlation Function* function block calculates the discrete correlation function between a reference signal (Channel Ref) and one or more other signals (Channel 00, ..., Channel 0n). The correlation coefficients are calculated for time shifts of *m* cycles between the two signals, with the maximum and minimum values for *m* being limited by the parameters *Minimum Lag* (negative integer) and *Maximum Lag* (positive integer).

The *Step Size* parameter determines the number of cycles by which the signals are moved to calculate two consecutive correlation coefficients. I.e. *m* is always a multiple of *StepSize*. Accordingly, for *Minimum Lag* and *Maximum Lag* only multiples of *StepSize* are allowed. If the *StepSize* is set to one and *Minimum Lag* is set to -6 and *Maximum Lag* is set to +4 -for example-, correlation coefficients are calculated for shifts of -6, -5, -4, -3, -2, -1, 0, +1, +2, +3 and +4 cycles. If the *StepSize* is set to two, coefficients are calculated for shifts by -6, -4, -2, 0, +2 and +4 cycles.

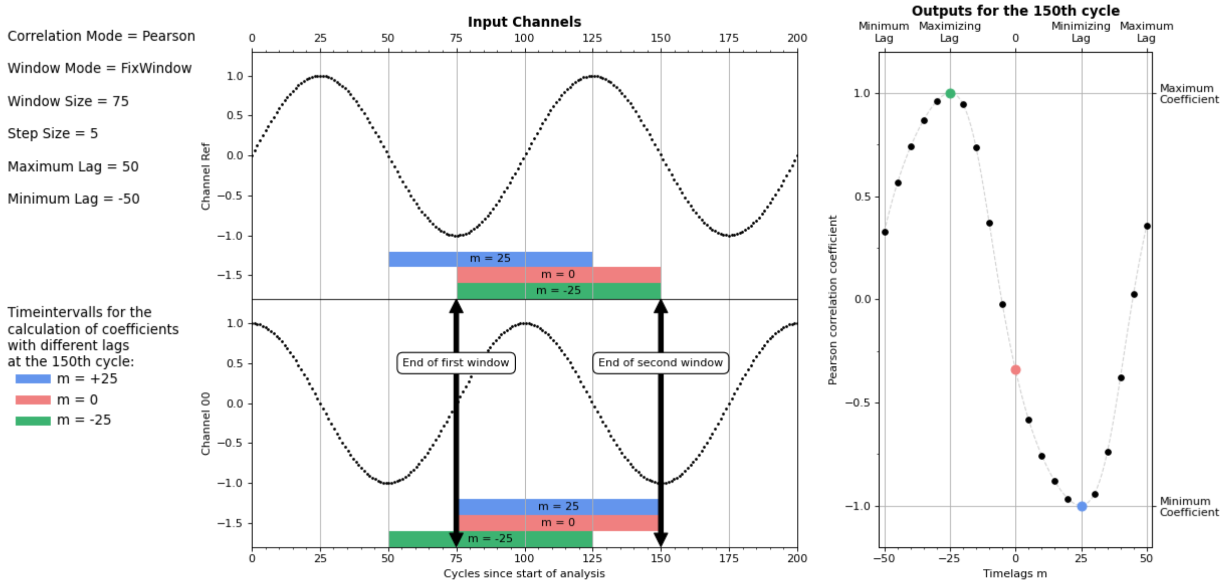
The coefficients can be calculated over different timeframes, which are set by the *Window Mode* parameter. In *Continuous* mode, all values since the beginning of the analysis are included in the calculations. In *SlidingWindow* mode, the calculation runs continuously for the last number of cycles set by the *Window Size*. In *FixWindow* mode, the calculation is also done for the number of cycles set by the *Window Size*. However, the calculation restarts after each *WindowSize* cycle and the output values are updated only when the last cycle of a window is run through.

If *m* is not zero, the window for the corresponding signal shifts by *m* cycles. The number of values included in the calculation of the coefficients is the same for all values of *m*. The only exceptions to this are the results from the first cycles after the start of the analysis. If the number of elapsed cycles is less than |*m*|, correspondingly fewer values are included in the calculation.

For positive values of *m*, the values of the reference signal (Channel Ref) are stored in a ring memory, so that the respective value of the reference signal received before *m* cycles can be compared with the current values from Channel 00 to Channel 0n. This corresponds to a shift of the reference signal into the past. For negative values of *m*, the reference signal would accordingly have to be moved into the future. Since this is obviously not possible, the second signal (Channel 00, .. Channel0n) is saved and moved backward instead.

The correlation coefficients can be calculated according to different calculation rules. This is determined by the *Correlation Mode*. In the *Base* and *Normed* modes, the coefficients are calculated analogously to the definition from signal processing, which calculates the correlation over the convolution. In *Normed* mode, the coefficients are also divided by the number of summands. *Covariance* and *CovarianceBessel* calculate the covariance without and with Bessel correction. *Pearson* mode uses the definition of the Pearson correlation coefficient commonly used in statistics. The exact calculation rules are mathematically listed in the configuration options. Here, x_n denotes the value of the reference signal and y_n denotes the value of the second signal (in each case Channel00, ..., Channel0n) at the timestamp t_n (corresponding to the nth cycle since the start of the analysis or since reset, except for the cycles in which *Enable Execution* = FALSE). The value of *N* depends on the *WindowMode* you select. For *SlidingWindow* mode and *FixWindow* mode, *N* is equal to the *WindowSize*, provided a corresponding number of cycles have already elapsed since the start of the analysis, so that x_{n-N-m} (or y_{n-N+m}) has been recorded, otherwise *N* will be reduced to *n-m+1* or *n+m+1* respectively. Note that in *FixWindow* mode the output values are only updated every *WindowSize* cycle. In *Continuous* mode, *N* = *n+1* always applies.

In the illustration, the output values of the function block for two signals (Channel Ref and Channel 00) for a given cycle (n = 150) are shown as an example of a configuration (*Correlation Mode* = *Pearson*, *Window Mode* = *FixWindow*, *Window Size* = 75, *Step Size* = 5, *Maximum Lag* = 50, *Minimum Lag* = -50). In the two left plots, the input signals Channel Ref and Channel 00 are shown over time. The right plot shows the discrete correlation function (the Pearson correlation coefficients in relation to m). Coefficients are shown for the shifts m = -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, +5, +10, +15, +20, +25, +30, +35, +40, +45, +50. These are output as an array in the function block. In addition to the two parameters Minimum Lag and Maximum Lag, the output values Minimizing Lag and Maximizing Lag are marked on the abscissa. The corresponding coefficients Minimum Coefficient and Maximum Coefficient, which also represent outputs of the function block, are marked on the ordinate. For the shifts m = -25, m = 0 and m =+25 in the plots of the input channels (left), the time ranges included in the calculation of the respective coefficient are highlighted in color. In the right plot, the corresponding points are colored.



Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of channels that are correlated with the reference signal. This can be set via Add/Remove Channel
- **Maximum Lag:** Specifies the maximum number of cycles by which the two signals are shifted to calculate the correlation to each other. This is a positive integer.
- **Minimum Lag:** Specifies the minimum number of cycles by which the two signals are shifted to calculate the correlation to each other. This is a negative integer.
- **Step Size:** Specifies by how many cycles the signals are shifted to calculate two consecutive correlation coefficients.
- **Window Size:** For the SlidingWindow and FixWindow window modes, specifies the number of cycles over which the coefficients are calculated. The windowing has no effect for the *Continuous* Window Mode and cannot be set. In *SlidingWindow* mode, Window Size values for all channels are buffered in addition to Maximum Lag values from the Reference Channel and Minimum Lag values for Channel 00 to Channel 0n. The size of the router memory should therefore be taken into account when setting these parameters.
- **Correlation Mode:** The correlation coefficients are calculated according to one of the following definitions:
- *Base*:

$$C_{xy}[m,t_n] = \begin{cases} \sum_{i=0}^{N-1} x_{n-i-m} \cdot y_{n-i}, & m \geq 0 \\ \sum_{i=0}^{N-1} x_{n-i} \cdot y_{n-i+m}, & m < 0 \end{cases}$$

- *Normed*:

$$\hat{C}_{xy}[m,t_n] = \begin{cases} \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m} \cdot y_{n-i}, & m \geq 0 \\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i} \cdot y_{n-i+m}, & m < 0 \end{cases}$$

- *Covariance*:

$$cov_{xy}[m,t_n] = \begin{cases} \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m} \cdot y_{n-i} - \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m} \cdot \frac{1}{N}\sum_{i=0}^{N-1} y_{n-i}, & m \geq 0 \\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i} \cdot y_{n-i+m} - \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i} \cdot \frac{1}{N}\sum_{i=0}^{N-1} y_{n-i+m}, & m < 0 \end{cases}$$

- *CovarianceBessel*:

$$\widetilde{cov}_{xy}[m,t_n] = \frac{N}{N-1} cov_{xy}[m,t_n]$$

- *Pearson:*

$$\rho_{xy}[m,t_n] = \frac{cov(x,y)[m,t_n]}{\sigma_x[m,t_n]\sigma_y[m,t_n]}, where\ \ \sigma_x^2[m,t_n] = cov_{xx}[m,t_n]$$

- **Window Mode:** Specifies the type of window used to calculate the coefficients:

  Continuous: All values since the start of the analysis are included in the analysis with equal weighting.

SlidingWindow: The calculation is done via a window of the size Window Size. The current values are always included in the analysis and outputs are updated with each cycle.

FixWindow: The outputs are updated every Window Size cycles and calculated via a window with the length Window Size.

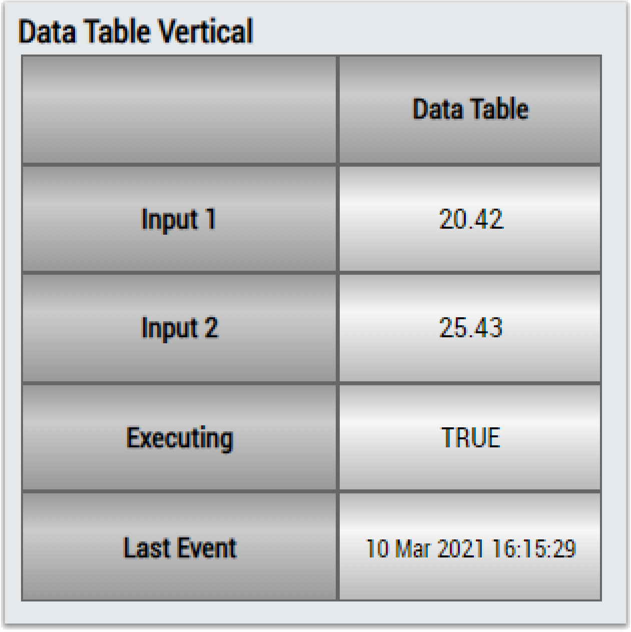
Output values

- **Output00 .. Output0n:** Shows for each input (Channel00, ..., Channel0n) the coefficients for the shifts *m* = *Minimum Lag*, *m* = *Minimum Lag* + *Step Size*, ..., *m* = - *Step Size*, *m* = 0, *m* = +*StepSize*, ..., m = *Maximum Lag* as an array.
- **Minimizing Lag00..Minimizing Lag0n:** For each channel, specifies the shift for which the coefficient becomes minimum.
- **Minimum Coefficient00..Minimum Coefficient0n:** Specifies the minimum coefficient for each channel.
- **Maximizing Lag00..Maximizing Lag0n:** for each channel, specifies the shift for which the coefficient becomes maximum.
- **Maximum Coefficient 00..00.. Maximum Coefficient0n:** Specifies the maximum coefficient for each channel.

Standard HMI Controls

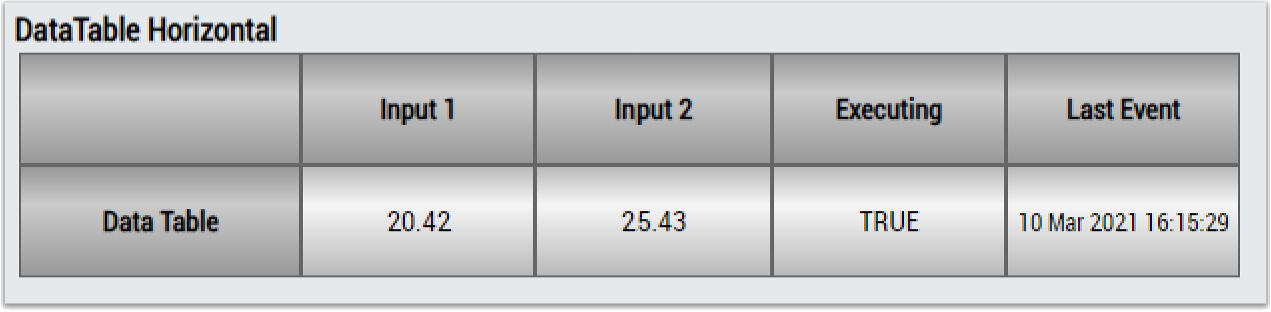
For the Correlation Function algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values: minimum coefficient, maximum coefficient, minimum lag, maximum lag and the coefficient array.



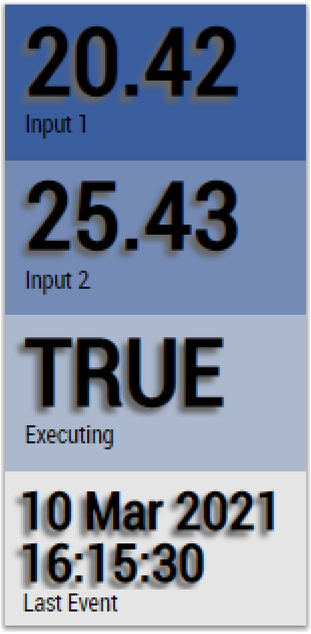
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

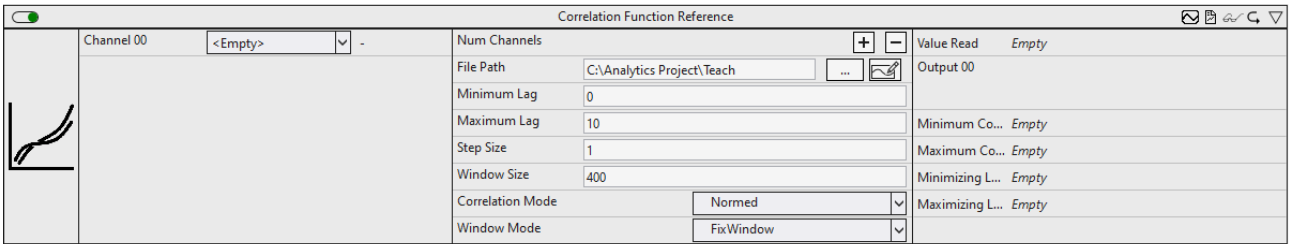
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Correlation Function algorithm using the Mapping Wizard [▸ 432].

6.4.10.4 Correlation Function Reference



Correlation Function Reference

Channel 00	<Empty> -	Num Channels	
		File Path	C:\Analytics Project\Teach
		Minimum Lag	0
		Maximum Lag	10
		Step Size	1
		Window Size	400
		Correlation Mode	Normed
		Window Mode	FixWindow

Value Read	Empty
Output 00	
Minimum Co...	Empty
Maximum Co...	Empty
Minimizing L...	Empty
Maximizing L...	Empty

The *Correlation Function Reference* function block calculates the discrete correlation function between a recorded signal (referred to below as reference signal), which is read from a tcab file, and one or more input signals (Channel 00, ..., Channel 0n).

The correlation coefficients are calculated for time shifts of *m* cycles between the two signals, with the maximum and minimum values for *m* being limited by the parameters *Minimum Lag* (negative integer) and *Maximum Lag* (positive integer).

The *Step Size* parameter determines the number of cycles by which the signals are moved to calculate two consecutive correlation coefficients. I.e. *m* is always a multiple of the *Step Size*. Accordingly, only multiples of the *Step Size* are also allowed for *Minimum Lag* and *Maximum Lag*. If the *Step Size* is set to one, *Minimum Lag* to -6 and *Maximum Lag* to +4, for example, coefficients are calculated for shifts by -6, -5, -4, -3, -2, -1, 0, +1, +2, +3, and +4 cycles. If the *Step Size* is set to two, coefficients are calculated for shifts by -6, -4, -2, 0, +2, and +4 cycles.

From the start of the analysis, a value is processed from the read-in signal per cycle. When the end of the file is reached, the process starts again with the first value of the file. The read-in signal is therefore assumed to be periodic. If you only want to correlate certain periods of the input signal with the reference signal, you can control this via the *Enable Execution* and *Reset* inputs as well as via the *New Result* output.

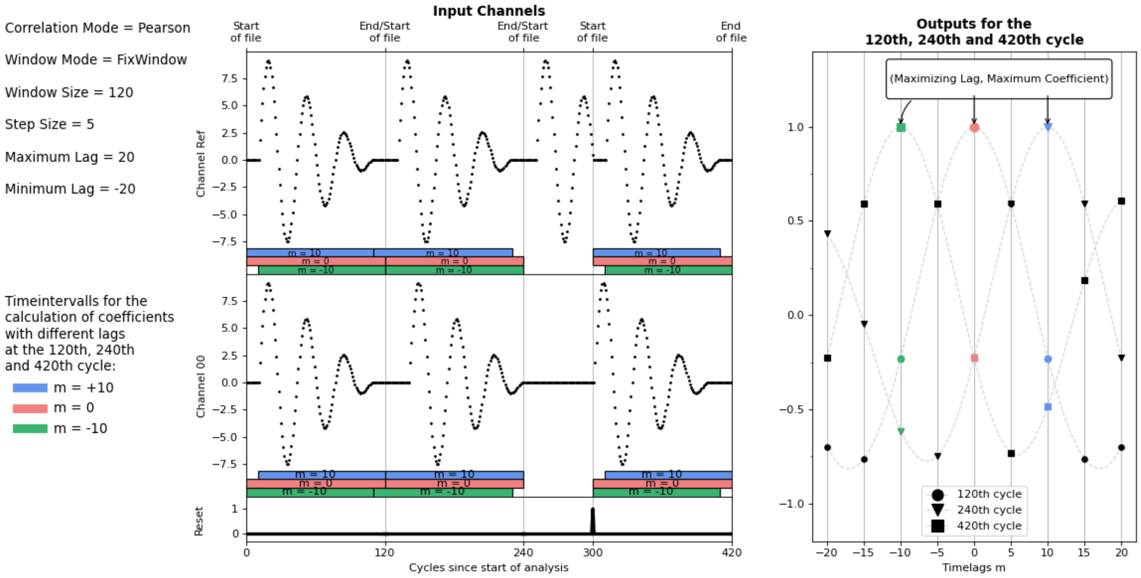
The correlation coefficients can be calculated over different timeframes. which are set by the *Window Mode* parameter. In *Continuous* mode, all values since the beginning of the analysis are included in the calculations. In *SlidingWindow* mode, the calculation runs continuously for the last number of cycles set by the *Window Size*. In *FixWindow* mode, the calculation is done over the length of the reference signal and the output values are always updated at the end of the recorded sequence.

If *m* is not zero, the window for the corresponding signal shifts by *m* cycles. The number of values included in the calculation of the coefficients is the same for all values of *m*. The only exceptions to this are the results from the first cycles after the start of the analysis. If the number of elapsed cycles is less than |*m*|, correspondingly fewer values are included in the calculation.

For positive values of *m,* the values of the reference signal (Channel Ref) are stored in a ring memory, so that the respective value of the reference signal received before *m* cycles can be compared with the current values from Channel 00 to Channel 0n. This corresponds to a shift of the reference signal into the past. For negative values of *m,* the reference signal would accordingly have to be moved into the future. Since this is obviously not possible, the second signal (Channel 00, .. Channel0n) is saved and moved backward instead.

The correlation coefficients can be calculated according to different calculation rules. This is determined by the *Correlation Mode*. In the *Base* and *Normed* modes, the coefficients are calculated analogously to the definition from signal processing, which calculates the correlation over the convolution. In *Normed* mode, the coefficients are also divided by the number of summands. *Covariance* and *CovarianceBessel* calculate the covariance without and with Bessel correction. *Pearson* mode uses the definition of the Pearson correlation coefficient commonly used in statistics. The exact calculation rules are mathematically listed in the configuration options. Here, x_n denotes the value of the reference signal and y_n denotes the value of the input signal (in each case Channel00, ..., Channel0n) at the timestamp t_n (corresponding to the nth cycle since the start of the analysis or since reset, except for the cycles in which *Enable Execution* = FALSE). The value of *N* depends on the *WindowMode* you select. For *SlidingWindow* mode and *FixWindow* mode, *N* is equal to the *WindowSize,* provided a corresponding number of cycles have already elapsed since the start of the analysis, so that x_{n-N-m} (or y_{n-N+m}) has been recorded, otherwise *N* will be reduced to *n-m+1* or *n+m+1* respectively. In FixWindow mode, the window size is not to be set manually, but corresponds to the length of the read-in signal section (reference signal) and the output values are updated at the end of the signal section. In *Continuous* mode, *N* = *n+1* always applies.

The illustration shows the different input and output values of the function block for a configuration (*Correlation Mode* = *Pearson*, *Window Mode* = *FixWindow*, *Window Size* = 120 (= number of values in the file), *Step Size* = 5, *Maximum Lag* = 20, *Minimum Lag* = -20) and an input channel. On the left side, the input signals Channel 00 and *Reset* are shown in the two lower plots, above which the read-in sequence is shown on the same timeline, according to its processing. The signal starts to be read in at the beginning of the analysis. If the last value from the file is processed in the 120th (or 240th) cycle, the process starts again with the first one. In the 300th cycle, a *reset* is performed and the process starts again with the first value of the file. In addition, all values are deleted from the internal memories and the calculation of the coefficients begins again. For example, it was detected here that Channel 00 did not contain valid values in the previous cycles and the vibration has now been re-energized. In this area the analysis could also be interrupted by *Enable Execution* = FALSE. Since *WindowMode* = *FixWindow*, the outputs are updated only in the 120th, 240th and 420th cycle. The corresponding coefficients (from Output00) are shown in the right plot. From the value pairs (*Maximizing Lag*, *Maximum Coefficient*) you can read how much the input signal is shifted with respect to the reference signal. For example, in the 420th cycle (*Maximizing Lag*, *Maximum Coefficient*) = (-10.1). This means that if the *Reset* had taken place 10 cycles earlier, the reference signal and Channel 00 would have matched each other exactly in this window.



Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of channels that are correlated with the reference signal. This can be set via Add/Remove Channel
- **File Path:** Path to the data file.
- **Maximum Lag:** Specifies the maximum number of cycles by which the two signals are shifted to calculate the correlation to each other. This is a positive integer.
- **Minimum Lag:** Specifies the minimum number of cycles by which the two signals are shifted to calculate the correlation to each other. This is a negative integer.
- **Step Size:** Specifies by how many cycles the signals are shifted to calculate two consecutive correlation coefficients.
- **Window Size:** For the *SlidingWindow* and *FixWindow* window modes, specifies the number of cycles over which the coefficients are calculated. The windowing has no effect for the *Continuous* Window Mode and cannot be set. In *SlidingWindow* mode, Window Size values for all channels are buffered in addition to Maximum Lag values from the Reference Channel and Minimum Lag values for Channel 00 to Channel 0n. The size of the router memory should therefore be taken into account when setting these parameters.
- **Correlation Mode:** The correlation coefficients are calculated according to one of the following definitions:
- *Base*:

$$C_{xy}[m,t_n] = \begin{cases} \sum_{i=0}^{N-1} x_{n-i-m} \cdot y_{n-i}, & m \geq 0 \\ \sum_{i=0}^{N-1} x_{n-i} \cdot y_{n-i+m}, & m < 0 \end{cases}$$

- *Normed:*

$$\hat{C}_{xy}[m,t_n] = \begin{cases} \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m} \cdot y_{n-i}, & m \geq 0 \\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i} \cdot y_{n-i+m}, & m < 0 \end{cases}$$

- *Covariance:*

$$cov_{xy}[m,t_n] = \begin{cases} \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m} \cdot y_{n-i} - \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m} \cdot \frac{1}{N}\sum_{i=0}^{N-1} y_{n-i}, & m \geq 0 \\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i} \cdot y_{n-i+m} - \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i} \cdot \frac{1}{N}\sum_{i=0}^{N-1} y_{n-i+m}, & m < 0 \end{cases}$$

- *CovarianceBessel:*

$$\widetilde{cov}_{xy}[m,t_n] = \frac{N}{N-1} cov_{xy}[m,t_n]$$

- *Pearson:*

$$\rho_{xy}[m,t_n] = \frac{cov(x,y)[m,t_n]}{\sigma_x[m,t_n]\sigma_y[m,t_n]}, where\ \ \sigma_x^2[m,t_n] = cov_{xx}[m,t_n]$$

- **Window Mode:** Specifies the type of window used to calculate the coefficients:

  Continuous: All values since the start of the analysis are included in the analysis with equal weighting.

  SlidingWindow: The calculation is done via a window of the size Window Size. The current values are always included in the analysis and outputs are updated with each cycle.

  FixWindow: The coefficients are calculated over the entire length of the read-in signal section and the outputs are updated after processing the last value.

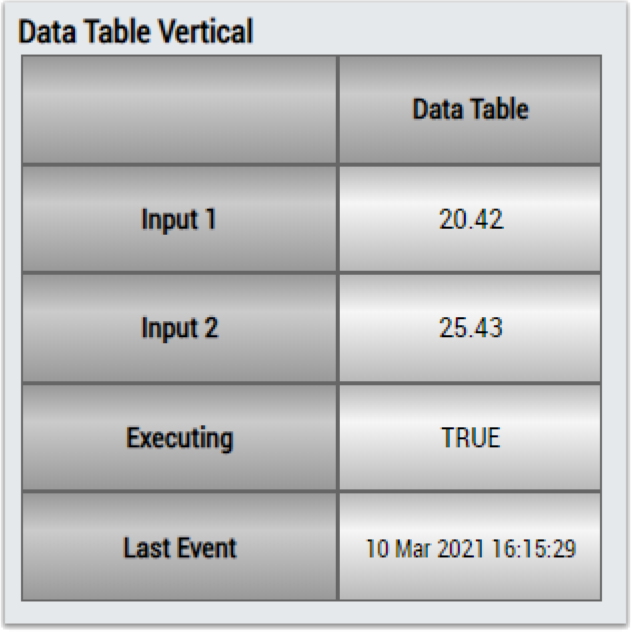
Output values

- **Output00 .. Output0n:** Shows for each input (Channel00, ..., Channel0n) the coefficients for the shifts *m* = *Minimum Lag*, *m* = *Minimum Lag* + *Step Size*, ..., *m* = - *Step Size*, *m* = 0, *m* = +*StepSize*,..., m = *Maximum Lag* as an array.
- **Minimizing Lag00..Minimizing Lag0n:** For each channel, specifies the shift for which the coefficient becomes minimum.
- **Minimum Coefficient00..Minimum Coefficient0n:** Specifies the minimum coefficient for each channel.
- **Maximizing Lag00..Maximizing Lag0n:** for each channel, specifies the shift for which the coefficient becomes maximum.
- **Maximum Coefficient 00..00.. Maximum Coefficient0n:** Specifies the maximum coefficient for each channel.

Standard HMI Controls

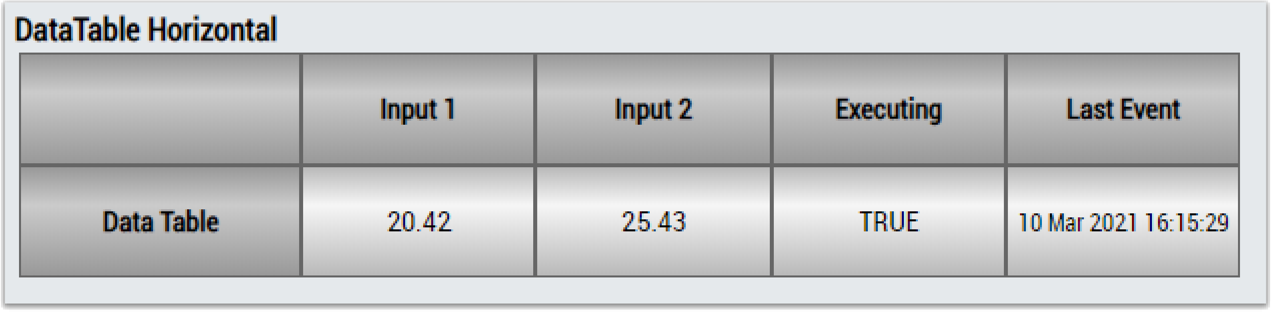
For the Correlation Function Reference algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values: read value, minimum coefficient, maximum coefficient, minimum lag, maximum lag and the coefficient array.



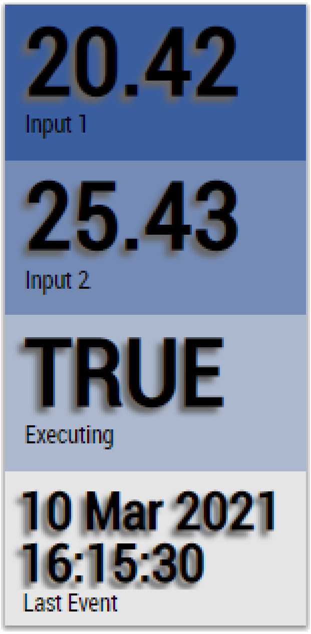
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

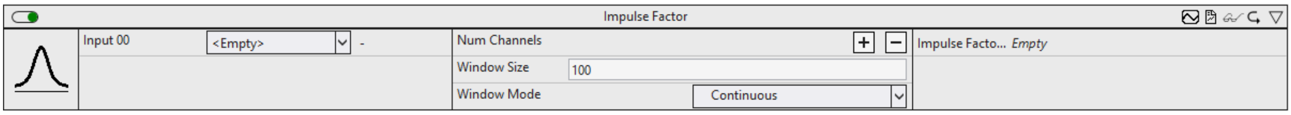
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Correlation Function Reference algorithm using the Mapping Wizard [▸ 432].

6.4.10.5 Impulse Factor



Impulse Factor

Input 00 <Empty> -

Num Channels

Window Size 100

Window Mode Continuous

Impulse Facto... *Empty*

The algorithm *Impulse Factor* calculates the signal feature of the same name from the input values. The output value is calculated from the ratio of the peak value of the input signal to the mean value of the input signal.

$$\text{Impulse Factor} = \frac{\max(|x|)}{\frac{1}{N}\sum_{n=1}^{N}|x[n]|}$$

The number of input data to be included in the calculation and the type of calculation can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** Configuration of the number of independent input and output channels.
- **Window Mode:** Used window mode. Influences the number of input data that are included in the calculation as well as the time of the calculations.
  Continuous: All input values since the start of the algorithm are included in the calculation. The calculation is cyclic.
  Fix Window: Only the last *N* input values are included in the calculation. The calculation is performed every *N* calls.
  Sliding Window: Only the last *N* input values are included in the calculation. The calculation is performed cyclically.
- **Window Size:** Depending on the window mode used, you can configure the number of values *N* that will be included in the calculation. In the window mode *Continuous* this parameter is ignored.

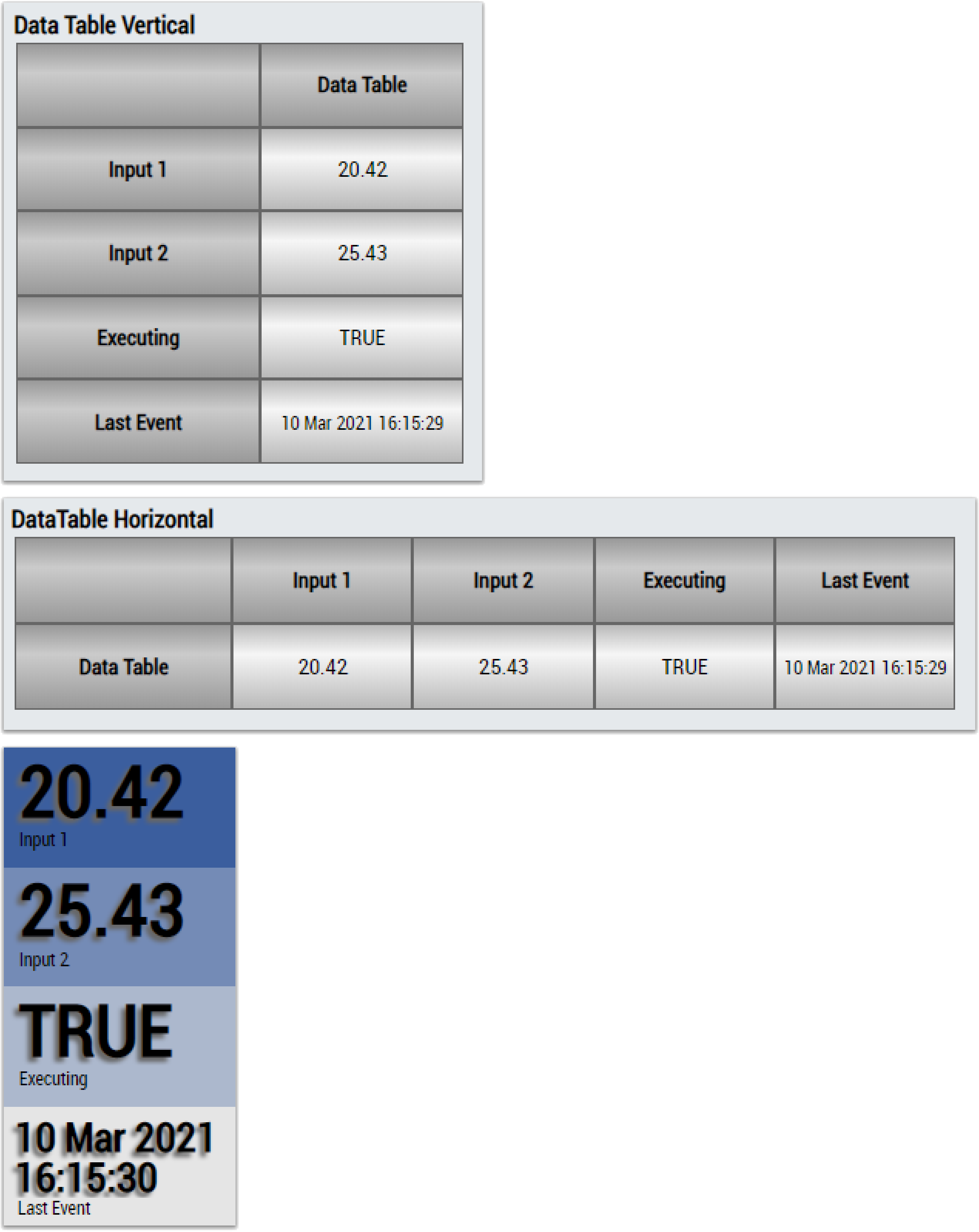
Output values

- **Impulse Factor 00..n:** Impulse factor of the input channels.

Standard HMI Controls

For the *Impulse Factor* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control and Multivalue Control visualize all output values.



Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29

20.42
Input 1

25.43
Input 2

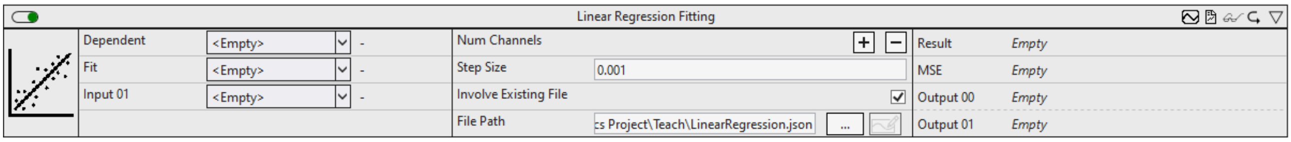
TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Fig. 3:

Alternatively, customer-specific HMI controls can be mapped using the Mapping Wizard [▸ 432].

6.4.10.6 Linear Regression Fitting



Linear Regression Fitting

Dependent <Empty> -
Fit <Empty> -
Input 01 <Empty> -

Num Channels
Step Size 0.001
Involve Existing File ☑
File Path cs Project\Teach\LinearRegression.json ...

Result Empty
MSE Empty
Output 00 Empty
Output 01 Empty

The Linear Regression Fitting function block approximates one variable (the Dependent input) by linear combination of several other variables (Input 01 ... Input 0n). This is done by the incremental stochastic gradient method. At the end of the analysis, the calculated coefficients are written to a file.

The linear combination is given by the following equation:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i \times \text{Input } 0i$$

In each cycle, the values for β_0 to β_n are recalculated using the following rule:

$$\beta_i = \begin{cases} \beta_i - \gamma \times \text{Input } 0i \times (y - \text{Dependent}), & i = 1, 2, \ldots, n \\ \beta_i - \gamma \times (y - \text{Dependent}), & i = 0 \end{cases}$$

This corresponds to the minimization of the squared deviation of the calculated values y (output by the function block as result) from the corresponding input value Dependent. The parameter γ corresponds to the step size and specifies how strongly the parameters are adjusted. The larger the value, the faster the coefficients approach a local optimum. However, if the value is too large, the algorithm may not converge.

Typically, the Linear Regression Fitting function block is first used to fit the weights for the regression of a target variable. Then, using the Linear Regression Inference function block and the fitted weights, the target variable can be predicted based on the input variables.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of channels that are correlated with the reference signal. This can be set via Add/Remove Channel
- **File Path:** Specifies the file path for the file in which the coefficients are saved at the end of the approximation process. File type: *json* or *tas*.
- **Involve Existing File:** If TRUE, the values for the coefficients are read from the file at the start of the analysis and then adjusted further. If FALSE, all coefficients are set to zero at the start.
- **Step Size:** Specifies how much the coefficients are adjusted after each new calculation.
- **Bias (optional):** If FALSE, the Bias Output 00 is set to zero and is not approximated further.
- **Mini Batch Size (optional):** Specifies over how many cycles the MSE is to be calculated before the coefficients are adjusted based on it.

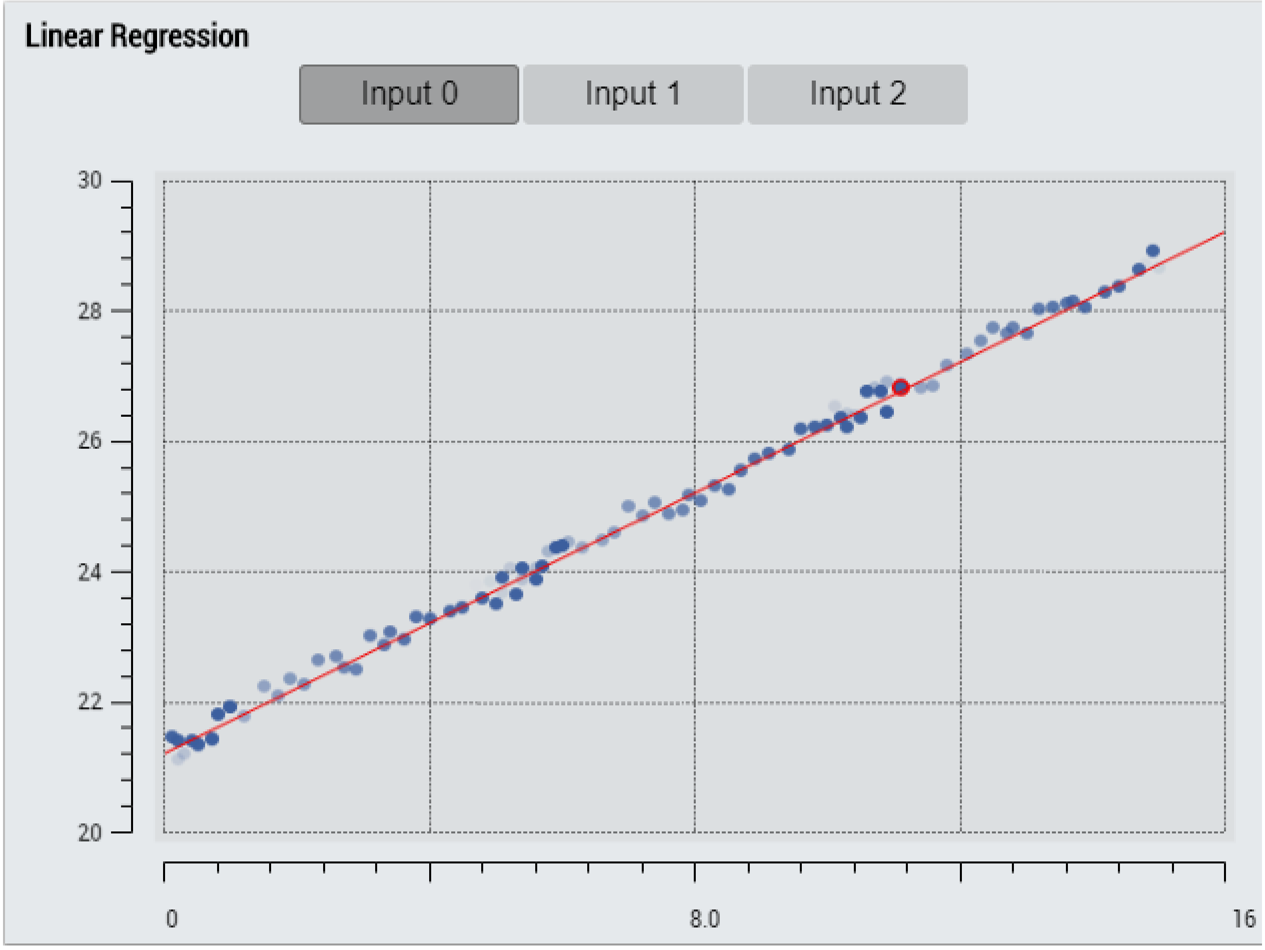
Output values

- **MSE:** Specifies the MSE (mean squared error) between the calculated *Result* value and the *Dependent* input value.
- **Result:** Outputs the approximated value for the inputs of the current cycle with the coefficients updated from them.
- **Output00 .. Output0n:** Shows the calculated coefficients.

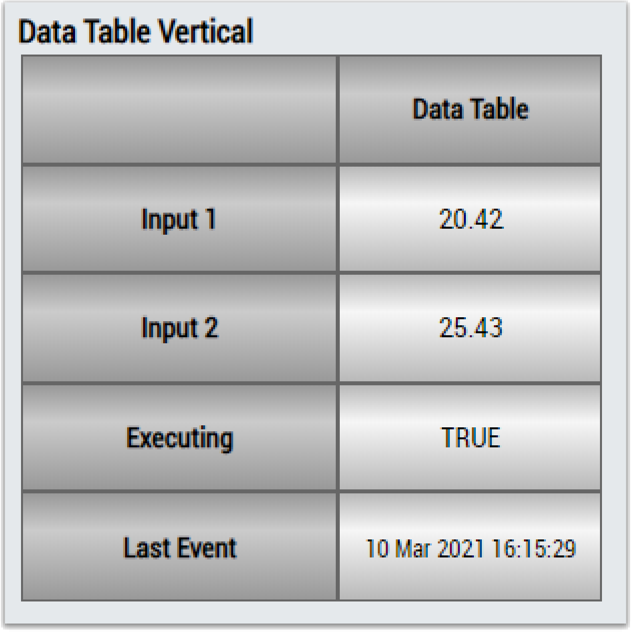
Standard HMI Controls

For the Linear Regression Fitting algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Linear Regression Control visualizes the inputs and the calculated regression line. The buttons can be used to select the input channel to be displayed on the x axis. The y axis shows the target values of the regression. A new point is outlined in red, old points gradually fade.

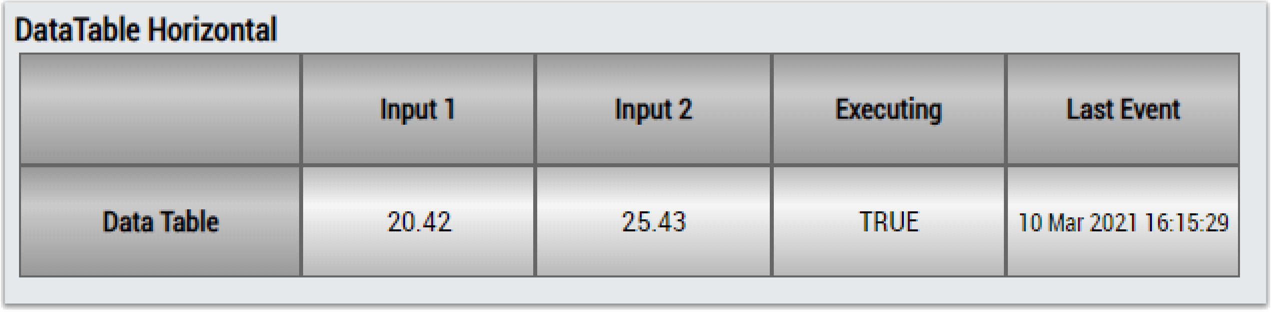


2. The Table Control or Multivalue Control visualizes all output values: MSE, result, output coefficients.



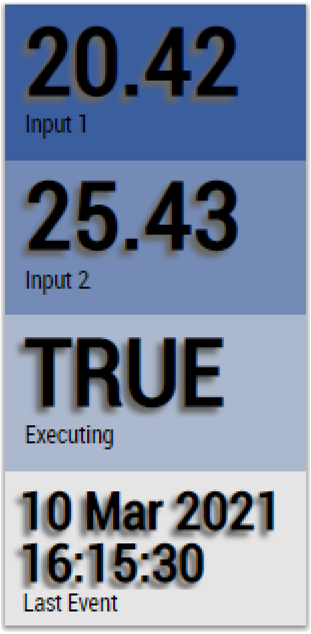
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

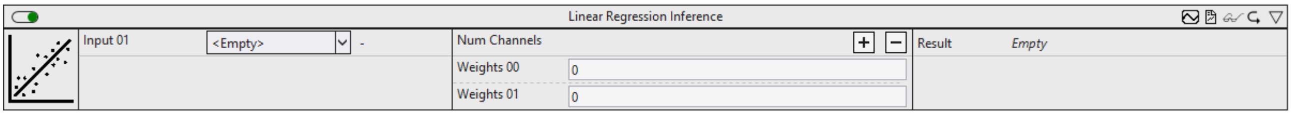
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Linear Regression Curve Fitting algorithm using the Mapping Wizard [▸ 432].

6.4.10.7 Linear Regression Inference



Linear Regression Inference

Input 01 <Empty> -

Num Channels

Weights 00: 0

Weights 01: 0

Result: *Empty*

The Linear Regression Inference function block calculates the linear combination of the inputs (Input 01 .. Input 0n) with the coefficients (Weights 00.. Weights 0n).

$$\text{Result} = \text{Weights } 00 + \sum_{i=1}^{n} \text{Weights } 0i \times \text{Input } 0i$$

The parameters Weights 00 to Weights 0n can either be set manually or automatically via a file generated by the Linear Regression Fitting function block by dragging / dropping onto the parameter field. Typically, the Linear Regression Fitting function block is first used to fit the weights for the regression of a target variable. Then, using the Linear Regression Inference function block and the fitted weights, the target variable can be predicted based on the input variables.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** The number of channels that are correlated with the reference signal. This can be set via Add/Remove Channel

- **Weights00 .. Weights 0n:** Specifies the coefficients that determine the linear combination to be calculated.
- **File Path (optional):** File path to a file with previously determined parameters. The file is created by the *Linear Regression Fitting* module. File type: *json or tas*.

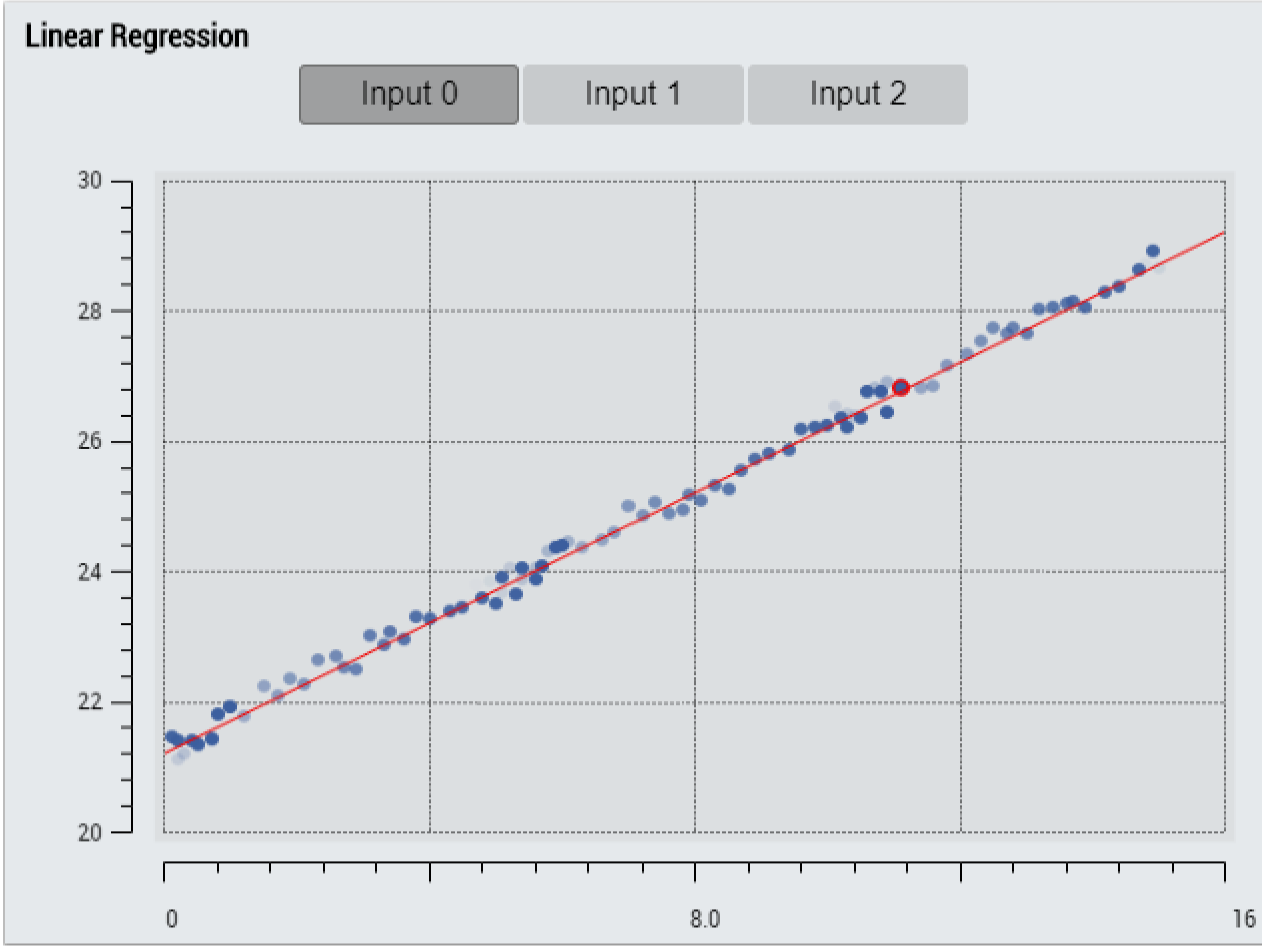
Output values

- **Result:** Specifies the value calculated from the linear combination.

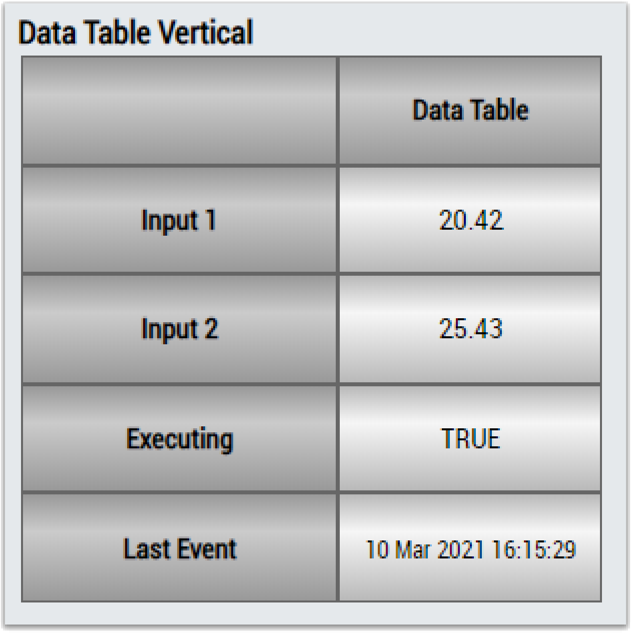
Standard HMI Controls

For the Linear Regression Inference algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Linear Regression Control visualizes the inputs and the calculated regression line. The buttons can be used to select the input channel to be displayed on the x axis. The y axis shows the target values of the regression. A new point is outlined in red, old points gradually fade.

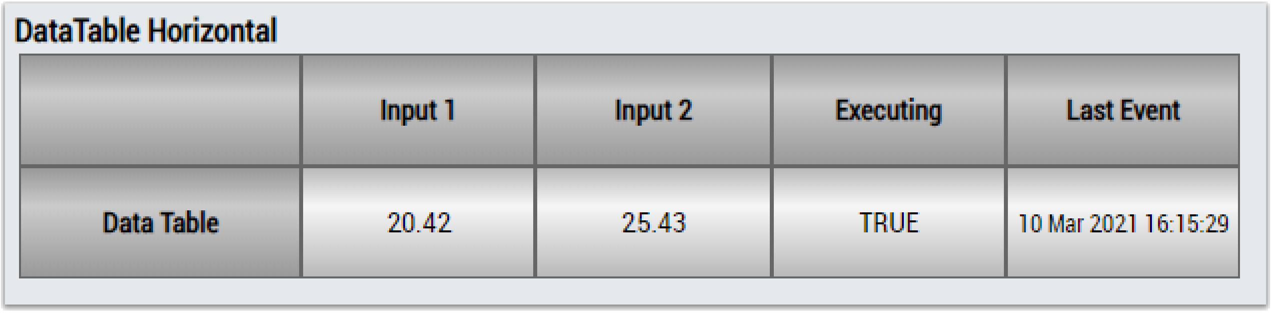


2. The Table Control or Multivalue Control visualizes all output values: MSE, result, output coefficients.



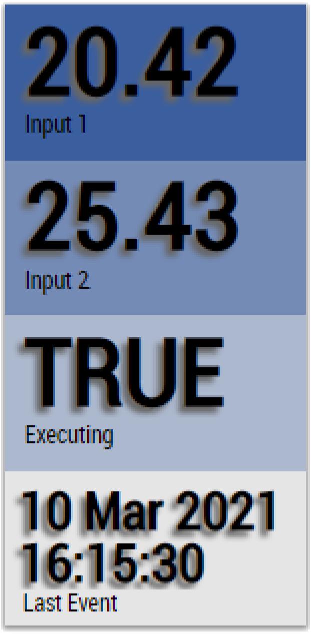
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

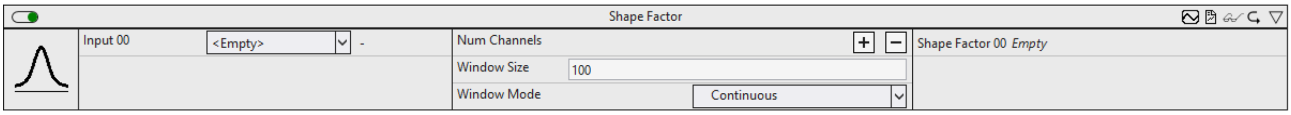
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Linear Regression Inference algorithm using the Mapping Wizard [▶ 432].

6.4.10.8 Shape Factor



Shape Factor

Input 00 <Empty> -

Num Channels

Window Size 100

Window Mode Continuous

Shape Factor 00 *Empty*

The algorithm *Shape Factor* calculates the signal feature of the same name from the input values. The output value is calculated from the ratio of the root mean square (RMS) of the input signal to the mean of the absolute values of the input signal.

$$\text{Shape Factor} = \frac{x_{\text{rms}}}{\frac{1}{N}\sum_{n=1}^{N}|x[n]|}$$

The number of input data to be included in the calculation and the type of calculation can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** Configuration of the number of independent input and output channels.
- **Window Mode:** Used window mode. Influences the number of input data that are included in the calculation as well as the time of the calculations.
  Continuous: All input values since the start of the algorithm are included in the calculation. The calculation is cyclic.
  Fix Window: Only the last *N* input values are included in the calculation. The calculation is performed every *N* calls.
  Sliding Window: Only the last *N* input values are included in the calculation. The calculation is performed cyclically.
- **Window Size:** Depending on the window mode used, you can configure the number of values *N* that will be included in the calculation. In the window mode *Continuous* this parameter is ignored.

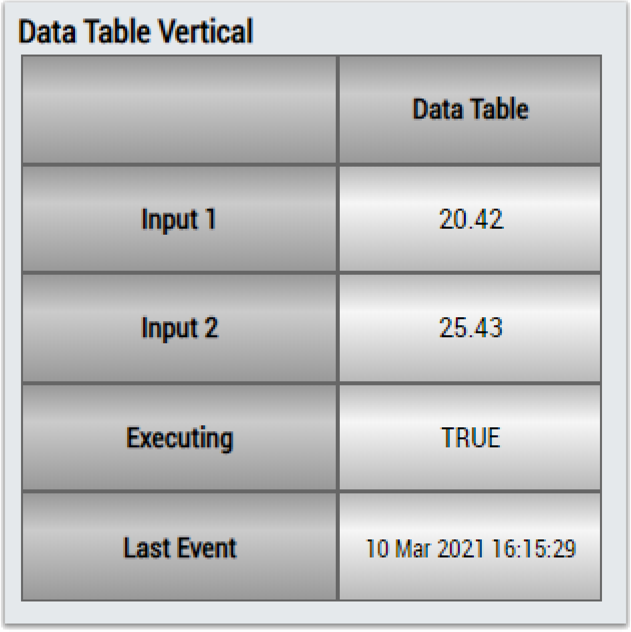
Output values

- **Shape Factor 00..n:** Shape factor of the input channels.

Standard HMI Controls

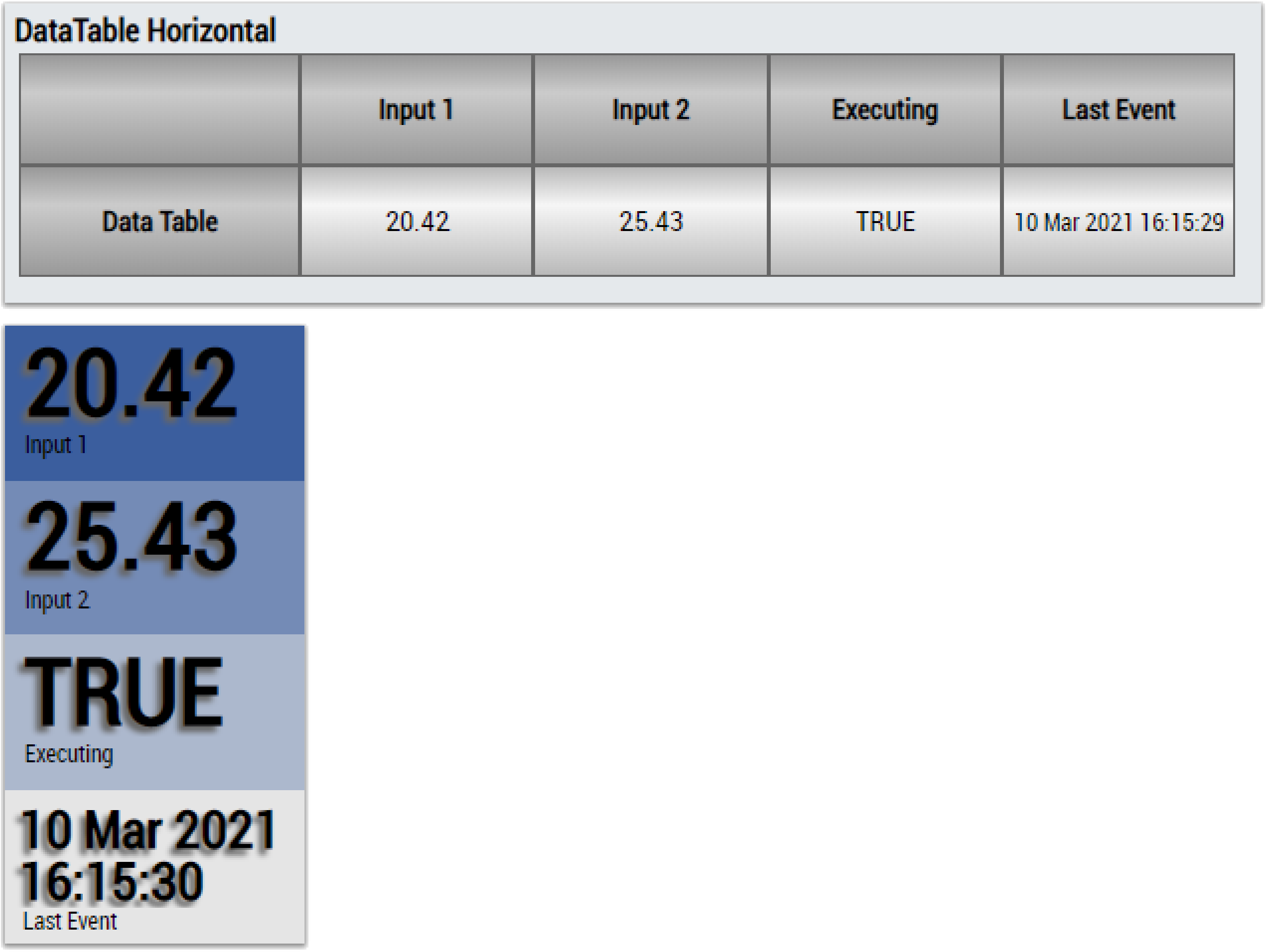
For the *Shape Factor* algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control and Multivalue Control visualize all output values.



Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29

20.42
Input 1

25.43
Input 2

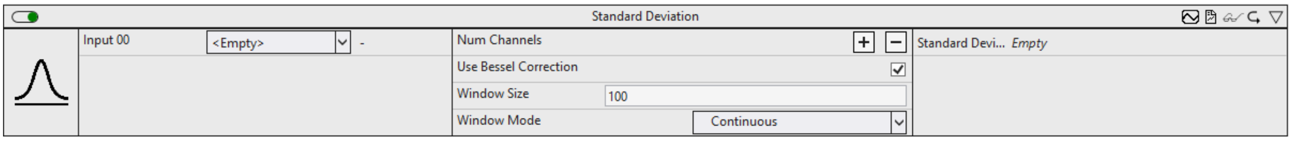
TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Fig. 4:

Alternatively, customer-specific HMI controls can be mapped using the Mapping Wizard [▶ 432].

6.4.10.9 Standard Deviation



Standard Deviation

Input 00 <Empty> -

Num Channels

Use Bessel Correction ☑

Window Size 100

Window Mode Continuous

Standard Devi... *Empty*

The *Standard Deviation* algorithm calculates the empirical standard deviation for a configurable number of input channels. The number of input data to be included in the calculation and the type of calculation can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Num Channels:** Configuration of the number of independent input and output channels.
- **Use Bessel Correction:** If the checkbox is checked, Bessel's correction is applied. This parameter must be enabled in order to obtain an expected result for random samples.

The empirical standard deviation, without Bessel's correction

$$s' = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$$

The empirical standard deviation, with Bessel's correction

$$s = \sqrt{\frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$$

- **Window Mode:** Window mode used. Influences the amount of input data included in the calculation and the timing of the calculations.
  Continuous: All input values since the start of the algorithm are included in the calculation. The calculation is performed cyclically.
  Fix Window: Only the last *N* input values are included in the calculation. The calculation is performed every *N* calls.
  Sliding Window: Only the last *N* input values are included in the calculation. The calculation is performed cyclically.
- **Window Size:** The number of values *N* that are included in the calculation can be configured depending on the window mode used. With the window mode *Continuous* this parameter is ignored.

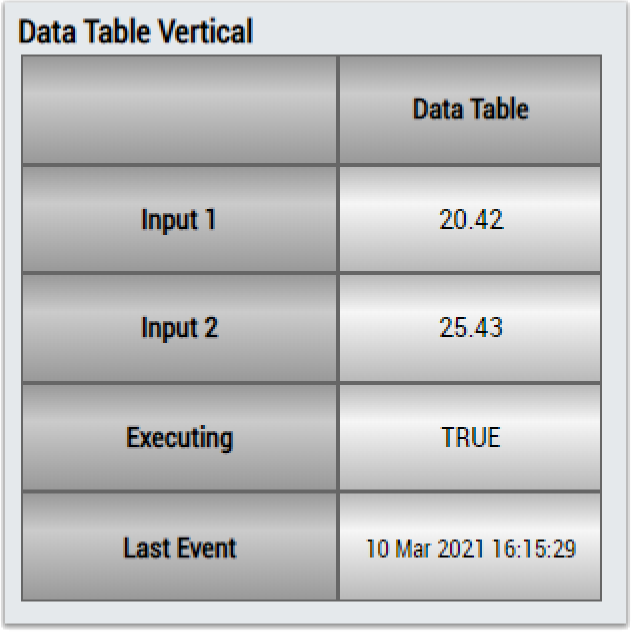
Output values

- **Standard Deviation 00..n:** Empirical standard deviation of the input channels.

Standard HMI Controls

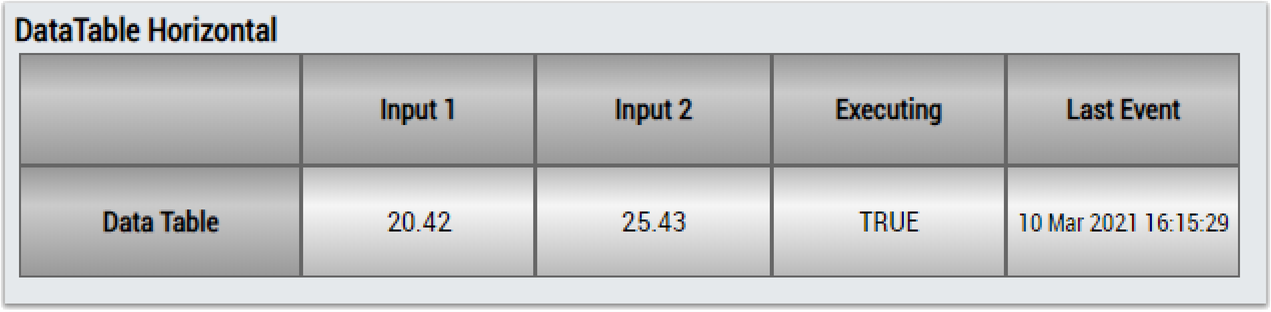
For the Standard Deviation algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The Table Control or Multivalue Control visualizes all output values: Standard Deviation Results.



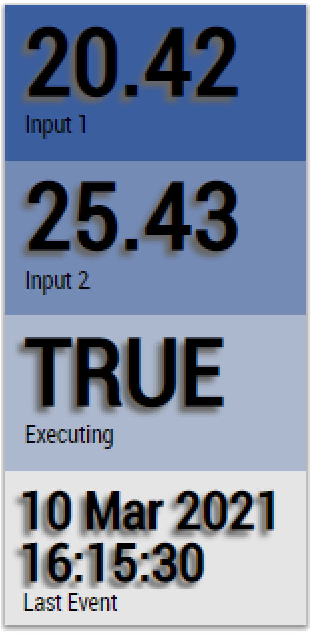
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

TRUE
Executing

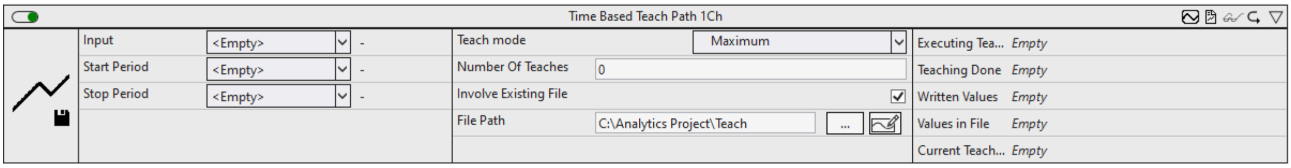
10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the Standard Deviation algorithm using the Mapping Wizard [▶ 432].

6.4.11 Analytics - Training Base

The algorithms of the category *Analytics – Training Base* provide functionalities for teaching periodic signals and write this data to a file. So that it is possible to compare it later on to another input signal, to analyze differences from the optimal behavior.

6.4.11.1 Time Based Teach Path 1Ch



Time Based Teach Path 1Ch

Input	<Empty> -	Teach mode	Maximum	Executing Tea...	Empty
Start Period	<Empty> -	Number Of Teaches	0	Teaching Done	Empty
Stop Period	<Empty> -	Involve Existing File	☑	Written Values	Empty
		File Path	C:\Analytics Project\Teach	Values in File	Empty
				Current Teach...	Empty

Time Based Teach Path 1Ch periodically writes the input data to a file according to the configured number of teach operations. This means that the values are not written sequentially for each period, but the values of a new period are compared with the existing values. The period can be defined by the input values *Start Period* and *Stop Period* (boolean signals are required). According to the teach mode, each value is overwritten or retained, so that the result is a taught input signal that can later be used as a reference signal for the Time Based Envelope 1Ch [▶ 154] algorithm, for example.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

It is recommended that you do not use Time Based Teach Path 1Ch at the same time as Time Based Envelope 1Ch [▶ 154] due to competing file access. Instead, the reference signal should be learned first and only then should an evaluation with Time Based Envelope 1Ch [▶ 154] be performed.

Configuration Options

- **Teach mode:** Mode for teaching. Defines according to which criteria the values will be compared (*Minimum*, *Maximum* or *Mean*). In case of *Mean* a weighted average is calculated, in order to ensure that the values of a later period do not have an increasing weight regarding the total result.
- **Number of Teaches:** Amount of cycles the teach process should be stopped after automatically. If set to 0 the teaching is processed continuously.

- **Involve Existing File:** If the checkbox is checked and a file with data already exists, the values of the existing file will be included in the teach process. Otherwise the existing file will be ignored and overwritten.
- **File Path:** Path to the data file.
- **Negate Start Period:** If the checkbox is checked the Boolean input signal of the *Start Period* is negated.
- **Negate Stop Period:** If the checkbox is checked the Boolean input signal of the *Stop Period* is negated.

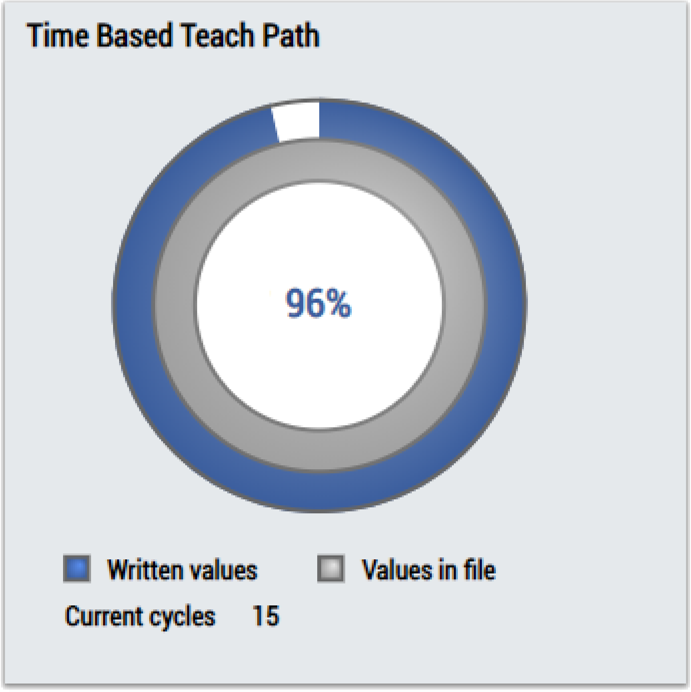
Output Values

- **Executing Teach:** Shows if the teaching is active (time range between start and stop flag).
- **Written Values:** Shows the total amount of written values during the teach process. Not to be confused with the amount of values in File, which are overwritten each teach cycle.
- **Values in File:** Shows the amount of values which are written currently into the file (after one cycle the value will be constant).
- **Current Teach Cycles:** Shows the amount of teach cycles within the file.

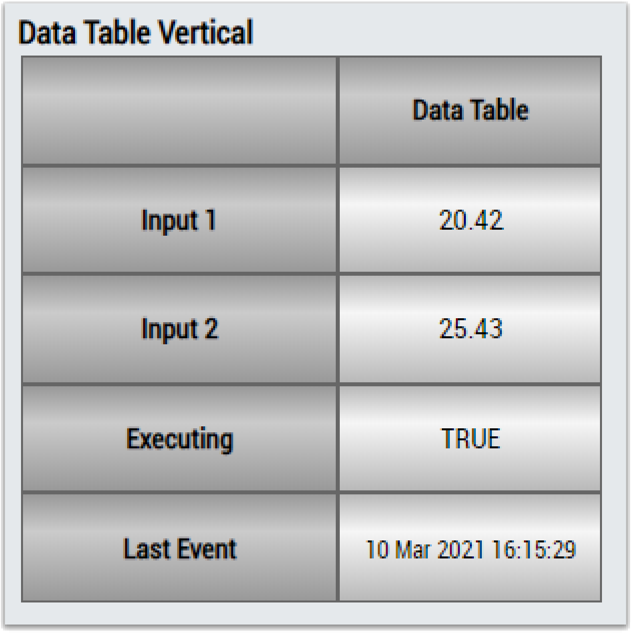
Standard HMI Controls

For the Time Based Teach Path 1Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The TimeBasedTeachPath Control visualizes the output values Written Values, Values in File and Current Teach Cycles.

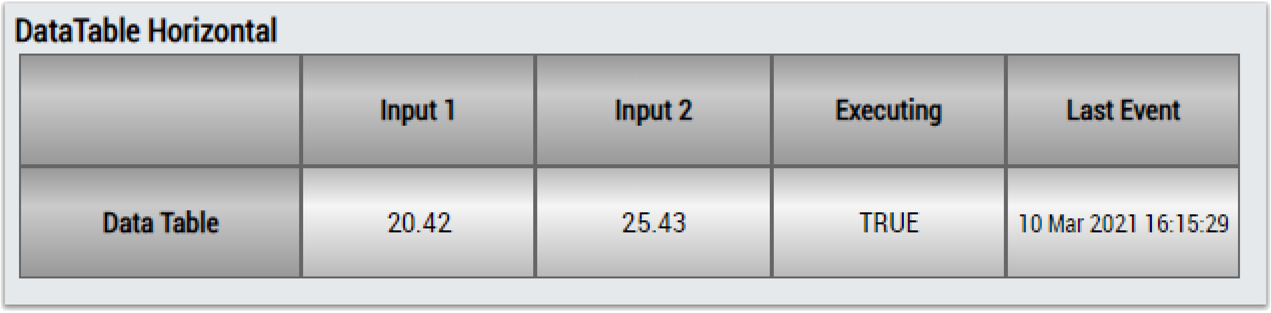


2. The Table Control or Multivalue Control visualizes all output values: Written Values, Values in File, Current Teach Cycles.



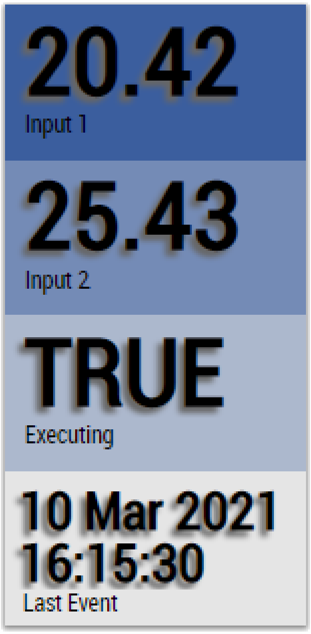
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

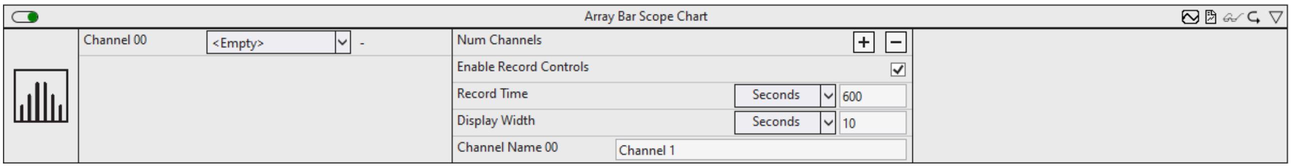
Alternatively, customer-specific HMI controls can be mapped in the Time Based Teach Path 1Ch algorithm using the Mapping Wizard [▶ 432].

6.4.12 Analytics - Visualization Only

The algorithms in the *Visualization Only* category represent auxiliary function blocks that provide functions for subsequent visualization in the context of the HMI dashboard.

6.4.12.1 Array Bar Scope Chart

The generation of the HMI Scope Control for the HMI One Click Dashboard is only possible with an HMI version > 1.12.752.0.



Array Bar Scope Chart

Channel 00	<Empty>	Num Channels	+ −
		Enable Record Controls	☑
		Record Time	Seconds 600
		Display Width	Seconds 10
		Channel Name 00	Channel 1

The *Array Bar Scope Chart* function block is used to display the inputs in the HMI Dashboard. The inputs are displayed in an Array Bar chart in the HMI Scope Control.

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

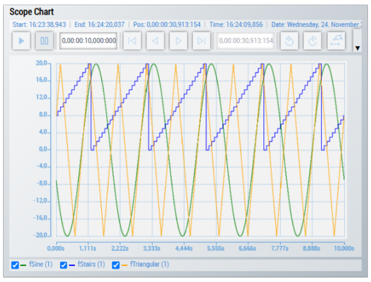
Configuration options

- **Num Channels:** the number of inputs to be displayed in the HMI Scope Control.
- **Channel Name 00/XX:** the display names of the inputs. These appear as legends in the HMI Scope Control.
- **Enable Record Controls:** if the checkbox is activated, a recording can be started and stopped from the HMI.
- **Record Time:** the recording time of a record. After reaching the time, the scope chart runs in a ring buffer.
- **Display Width:** the selected display width of the image. This property can also be modified in the HMI Scope Control.

Standard HMI Controls

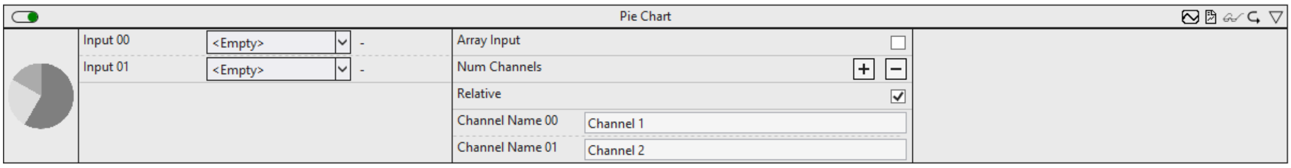
For the algorithm *Array Bar Scope Chart* the following HMI Control is available for generating an Analytics Dashboard.

The HMI Scope Control for displaying a Scope Chart.



No other HMI controls can be added to the function block, as it is intended exclusively for displaying the HMI Scope Control.

6.4.12.2 Pie Chart



Pie Chart

Input 00	<Empty>	Array Input	□
Input 01	<Empty>	Num Channels	+ −
		Relative	☑
		Channel Name 00	Channel 1
		Channel Name 01	Channel 2

The *Pie Chart* function block can be used to create a pie chart for later use in the HMI Dashboard. The number of pie pieces can be parameterized via the parameter*Num Channels*.

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

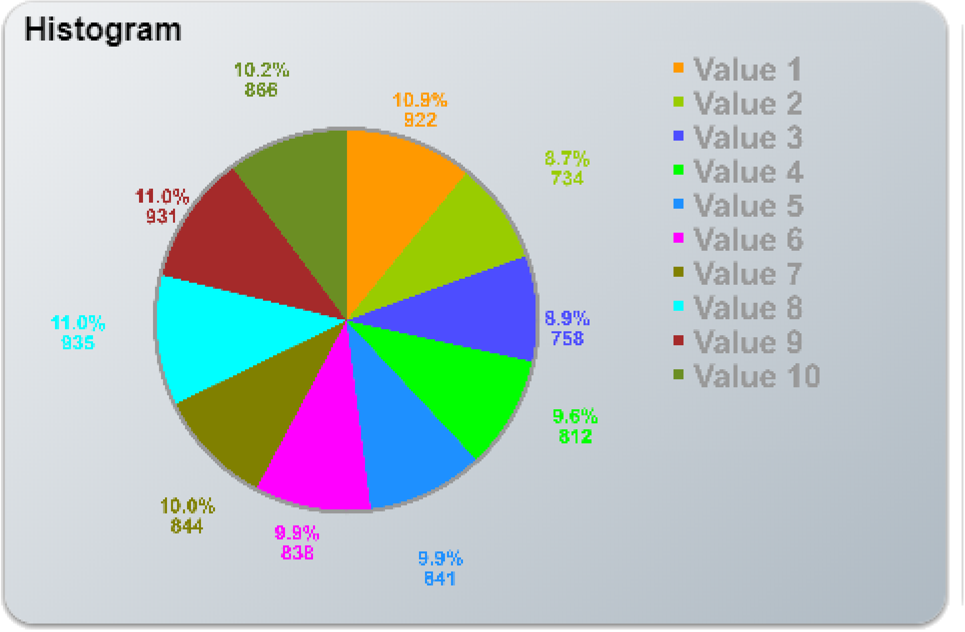
Configuration options

- **Array Input:** Selection of whether a single array should be used as an input or several independent inputs. Is set automatically as soon as an array is dragged onto the first input. As long as the array is still linked, the parameter cannot be reset.
- **Num Channels**: Specifies the number of configurable channels. This corresponds to the number of pie pieces in the pie chart.
- **Channel Name 00:** Specifies the name of the first channel. The name assigned here will later be used for this channel in the HMI Dashboard.
- **Channel Name 01:** Specifies the name of the second channel. The name assigned here will later be used for this channel in the HMI Dashboard.
- **Channel Name xx:** Analog for all other channels.

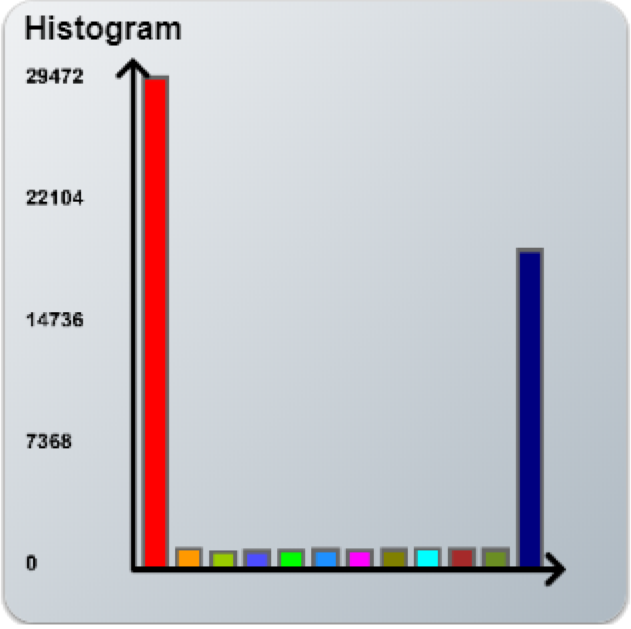
Standard HMI Controls

For the *Pie Chart* algorithm the following HMI controls are available for generating an Analytics Dashboard.

Pie Chart to display the data in a pie chart.

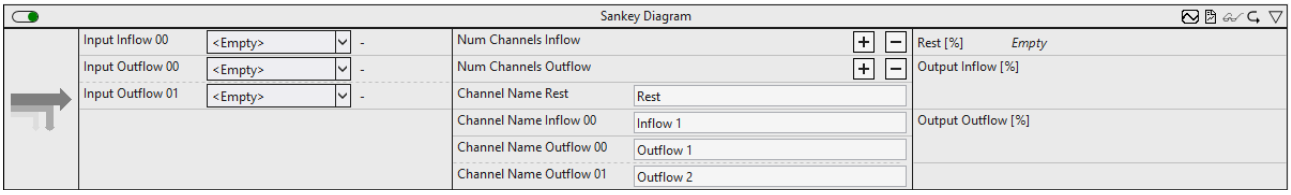


Histogram to display the data in a histogram.



Alternatively, customer-specific HMI controls can be mapped in the *Pie Chart* algorithm using the Mapping Wizard [▶ 432].

6.4.12.3 Sankey diagram


Sankey Diagram

Input Inflow 00 <Empty> -
Input Outflow 00 <Empty> -
Input Outflow 01 <Empty> -

Num Channels Inflow
Num Channels Outflow
Channel Name Rest: Rest
Channel Name Inflow 00: Inflow 1
Channel Name Outflow 00: Outflow 1
Channel Name Outflow 01: Outflow 2

Rest [%] *Empty*
Output Inflow [%]
Output Outflow [%]

The *Sankey Diagram* function block can be used to create a Sankey diagram for later use in the HMI Dashboard. A distinction is made between inflows and outflows. The number of inflows and outflows can be individually parameterized via the parameters *Num Channels Inflow* and *Num Channels Outflow* respectively.

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

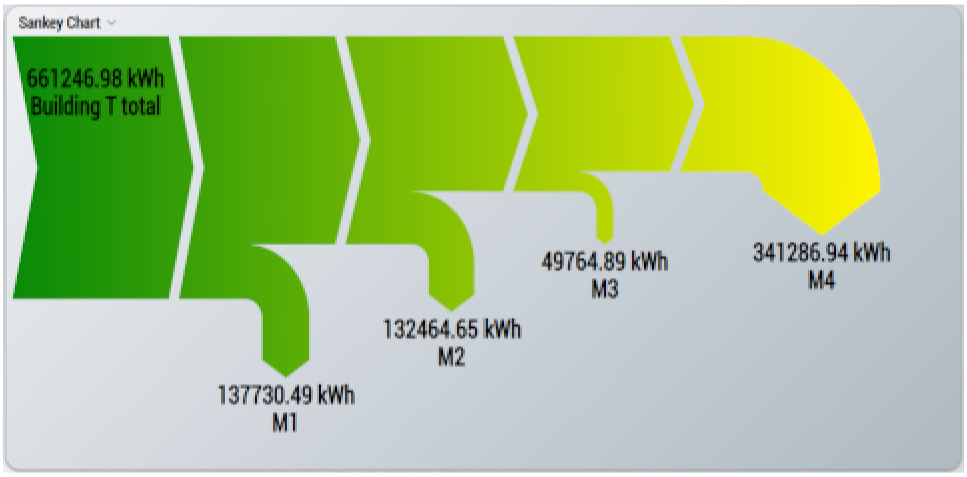
Configuration options

- **Num Channels Inflow**: specifies the number of channels of inflows in the Sankey diagram.
- **Num Channels Outflow**: specifies the number of channels of outflows in the Sankey diagram.
- **Channel Name Rest:** specifies the name of the channel with which a possible rest value is to be designated. The name assigned here will be used later for this outflow in the HMI Dashboard.
- **Channel Name Inflow 00:** specifies the name of the first inflow channel. The name assigned here will be used later for this inflow in the HMI Dashboard.
- **Channel Name Inflow xx:** analog for all other inflow channels.
- **Channel Name Outflow 00:** specifies the name of the first outflow channel. The name assigned here will be used later for this outflow in the HMI Dashboard.
- **Channel Name Outflow 01:** specifies the name of the second outflow channel. The name assigned here will be used later for this outflow in the HMI Dashboard.
- **Channel Name Outflow xx:** analog for all other channels.

Standard HMI Controls

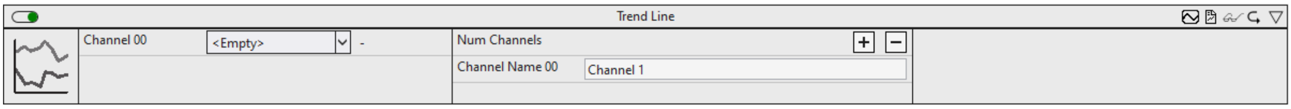
For the *Sankey Diagram* algorithm the following HMI control is available for generating an Analytics Dashboard.

Sankey chart to display the data.



Alternatively, customer-specific HMI controls can be mapped in the *Sankey Diagram* algorithm using the Mapping Wizard [▸ 432].

6.4.12.4 Trend Line



Channel 00 <Empty> -

Num Channels

Channel Name 00 Channel 1

The *Trend Line* function block provides for the forwarding of trends of the input signals for later use in the HMI dashboard. Any channels can be selected for forwarding; the number of channels can be controlled via NumChannels. One value per minute is then stored for each channel. This means that the function block is not suitable for signal forwarding, but rather for displaying trends for signals that change slowly.

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

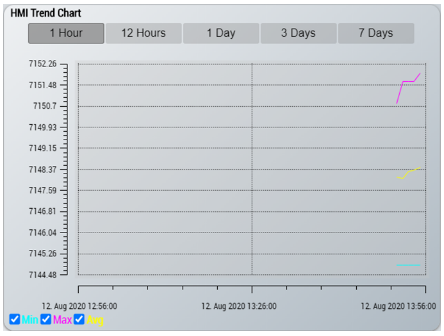
- **Num Channels**: Indicates the number of configurable channels for forwarding the input signals.
- **Channel 00:** Indicates the name of the *first* channel to be forwarded. The name assigned here will later be used for this channel in the HMI Dashboard.

- **Channel 01:** Indicates the name of the *second* channel to be forwarded. The name assigned here will later be used for this channel in the HMI Dashboard.
- **Channel 02:** Indicates the name of the *third* channel to be forwarded. The name assigned here will later be used for this channel in the HMI Dashboard.
- **Channel Name xx:** Analog for all other channels.

Standard HMI Controls

For the algorithm *Trend Line* the following HMI Control is available for generating an Analytics Dashboard.

Trend Line to display the data over time.



No other HMI controls can be added to the function block, as it is intended exclusively for displaying the Trend Line Control.

6.4.12.5 Referenced Scope

The generation of the HMI Scope Control for the HMI One Click Dashboard is only possible with an HMI version > 1.12.752.0.

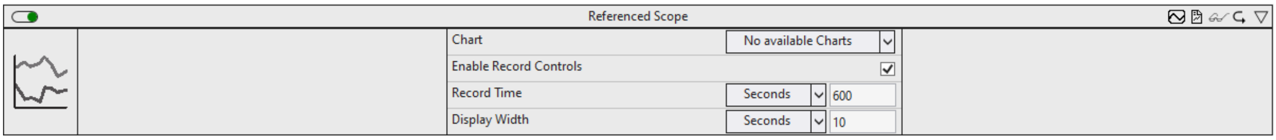

Referenced Scope

Chart	No available Charts	
Enable Record Controls	☑	
Record Time	Seconds	600
Display Width	Seconds	10

The *Referenced Scope* function block is used to display a Scope Chart from the Referenced Scope in the HMI Dashboard

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

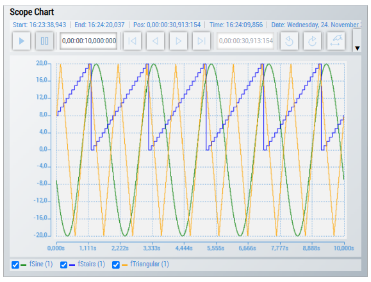
Configuration options

- **Chart:** the chart from the referenced scope to be displayed in the HMI. So far the following charts are supported: YT Chart, XY Chart, Array Bar Chart and Single Bar Chart.
- **Enable Record Controls:** if the checkbox is activated, a recording can be started and stopped from the HMI.
- **Record Time:** the recording time of a record. After reaching the time, the scope chart runs in a ring buffer.
- **Display Width:** the selected display width of the image. This property can also be modified in the HMI Scope Control.

Standard HMI Controls

For the algorithm *Referenced Scope* the following HMI Control is available for generating an Analytics Dashboard.

The HMI Scope Control for displaying a Scope Chart.



Scope Chart

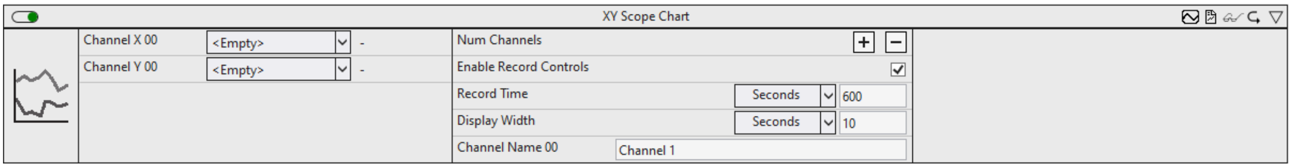
Start: 16:23:38,943 | End: 16:24:20,037 | Pos: 0,00:00:30,913:154 | Time: 16:24:09,856 | Date: Wednesday, 24. November

0,00:00:10,000:000 | 0,00:00:30,913:154

No other HMI controls can be added to the function block, as it is intended exclusively for displaying the HMI Scope Control.

6.4.12.6 XY Scope Chart

The generation of the HMI Scope Control for the HMI One Click Dashboard is only possible with an HMI version > 1.12.752.0.



XY Scope Chart

Channel X 00	<Empty>	-	Num Channels		
Channel Y 00	<Empty>	-	Enable Record Controls		☑
			Record Time	Seconds	600
			Display Width	Seconds	10
			Channel Name 00	Channel 1	

The *XY Scope Chart* function block is used to display the inputs in the HMI Dashboard. The inputs are displayed in an XY chart in the HMI Scope Control.

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

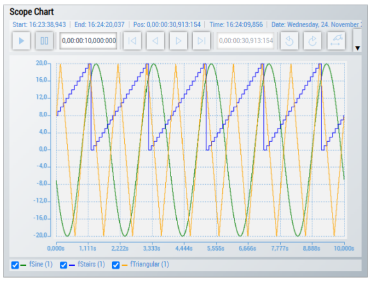
Configuration options

- **Num Channels:** the number of channels or input pairs to be displayed in the HMI Scope Control.
- **Channel Name 00/XX:** the display names of the channels. These appear as legends in the HMI Scope Control.
- **Enable Record Controls:** if the checkbox is activated, a recording can be started and stopped from the HMI.
- **Record Time:** the recording time of a record. After reaching the time, the scope chart runs in a ring buffer.
- **Display Width:** the selected display width of the image. This property can also be modified in the HMI Scope Control.

Standard HMI Controls

For the algorithm *XY Scope Chart* the following HMI Control is available for generating an Analytics Dashboard.

The HMI Scope Control for displaying a Scope Chart.



Scope Chart

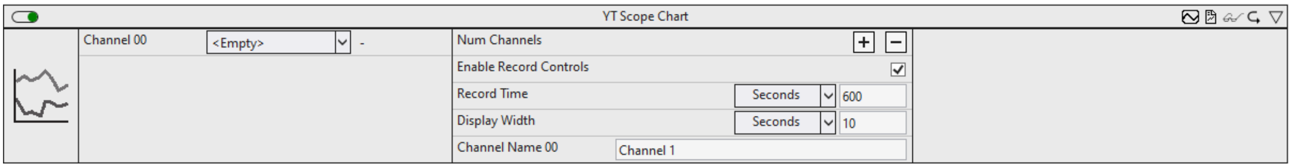
Start: 16:23:38,943 | End: 16:24:20,037 | Pos: 0,00:00:30,913:154 | Time: 16:24:09,856 | Date: Wednesday, 24. November

0,00:00:10,000:000 | 0,00:00:30,913:154

No other HMI controls can be added to the function block, as it is intended exclusively for displaying the HMI Scope Control.

6.4.12.7 YT Scope Chart

The generation of the HMI Scope Control for the HMI One Click Dashboard is only possible with an HMI version > 1.12.752.0.



Channel 00 <Empty> -

Num Channels

Enable Record Controls ☑

Record Time Seconds 600

Display Width Seconds 10

Channel Name 00 Channel 1

The *YT Scope Chart* function block is used to display the inputs in the HMI Dashboard. The inputs are displayed in a YT chart in the HMI Scope Control.

Since the function of this function block is irrelevant for the analysis itself, it belongs to the Visualization Only category.

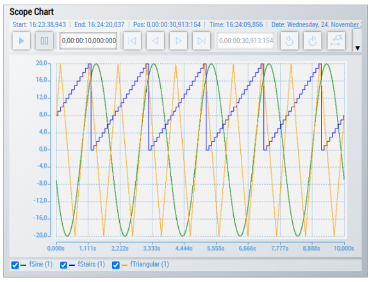
Configuration options

- **Num Channels:** the number of inputs to be displayed in the HMI Scope Control.
- **Channel Name 00/XX:** the display names of the inputs. These appear as legends in the HMI Scope Control.
- **Enable Record Controls:** if the checkbox is activated, a recording can be started and stopped from the HMI.
- **Record Time:** the recording time of a record. After reaching the time, the scope chart runs in a ring buffer.
- **Display Width:** the selected display width of the image. This property can also be modified in the HMI Scope Control.

Standard HMI Controls

For the algorithm *YT Scope Chart* the following HMI Control is available for generating an Analytics Dashboard.

The HMI Scope Control for displaying a Scope Chart.


Scope Chart

Start: 16:23:38,943 | End: 16:24:20,037 | Pos: 0,00:00:30,913:154 | Time: 16:24:09,856 | Date: Wednesday, 24. November

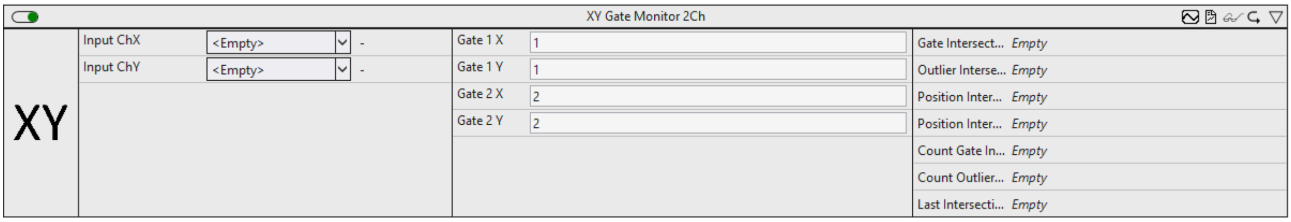
0,00:00:10,000:000 0,00:00:30,913:154

No other HMI controls can be added to the function block, as it is intended exclusively for displaying the HMI Scope Control.

6.4.13 Analytics - XY Path Analysis

The algorithms of the category *XY Path Analysis* provide functions for the position evaluation of XY channels. For example, it is possible to analyze whether the position determined by the input channels is within certain bounds or shapes and how often boundary crossings occur.

6.4.13.1 XY Gate Monitor 2Ch



XY Gate Monitor 2Ch

XY

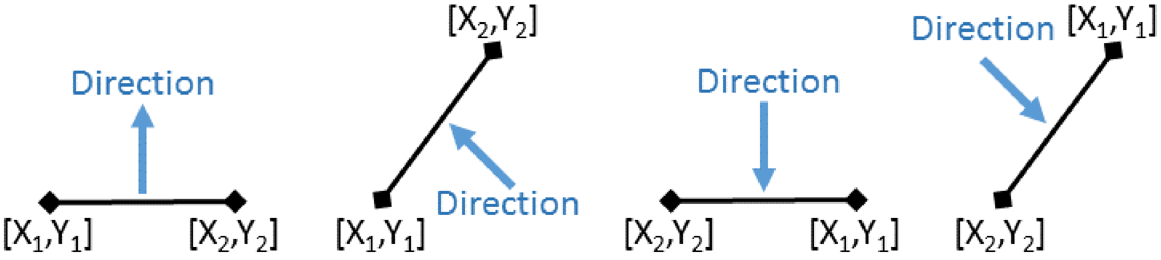
Input ChX	<Empty>	-	Gate 1 X	1		Gate Intersect...	Empty
Input ChY	<Empty>	-	Gate 1 Y	1		Outlier Interse...	Empty
			Gate 2 X	2		Position Inter...	Empty
			Gate 2 Y	2		Position Inter...	Empty
						Count Gate In...	Empty
						Count Outlier...	Empty
						Last Intersecti...	Empty

The *XY Gate Monitor 2Ch* counts the amount of intersections of an XY input with a specified gate or its projection (straight line between the gate points) depending on the configured Gate Mode. The analysis period can be started with the inputs *Start* and *Stop*. The algorithm is direction sensitive, which means that just intersection in the right direction are counted. The direction interpretation depends on the order of the gate points (X1/Y1) and (X2/Y2). The possible intersection directions are visualized below.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Directions of the intersection points:

The blue arrow represents the signal direction and the black lines illustrate the gate with its gate points (X1/Y1) and (X2/Y2). The direction of the intersection points is counted when the signal rotates counterclockwise around the first gate point (X1/Y1).



Configuration Options

- **Gate Mode:** Mode of the Gate Monitor:

  Intersect Gate: Determines if the XY signal intersects the gate in the configured direction. If there is an intersection during the analysis period, it will be classified as *OK*, otherwise *NOK*.

  Not Intersect Gate: Monitors if the XY signal does not intersect with the gate in the configured direction during the analysis period. Then it will be classified as *OK*, otherwise *NOK*.

  Intersect Projection: Determines if the XY signal intersects the projection of the gate in the configured direction. If there is an intersection during the analysis period, it will be classified as *OK*, otherwise *NOK*.

  Not Intersect Projection: Monitors if the XY signal does not intersect with the projection of the gate in the configured direction during the analysis period. Then it will be classified as *OK*, otherwise *NOK*.

  Intersect Gate Or Projection: Determines if the XY signal intersects the gate or its projection in the configured direction. If there is an intersection during the analysis period, it will be classified as *OK*, otherwise *NOK*.
- **Gate 1 X:** X position of the first gate point.
- **Gate 1 Y:** Y position of the first gate point.
- **Gate 2 X:** X position of the second gate point.
- **Gate 2 Y:** Y position of the second gate point.

Output values

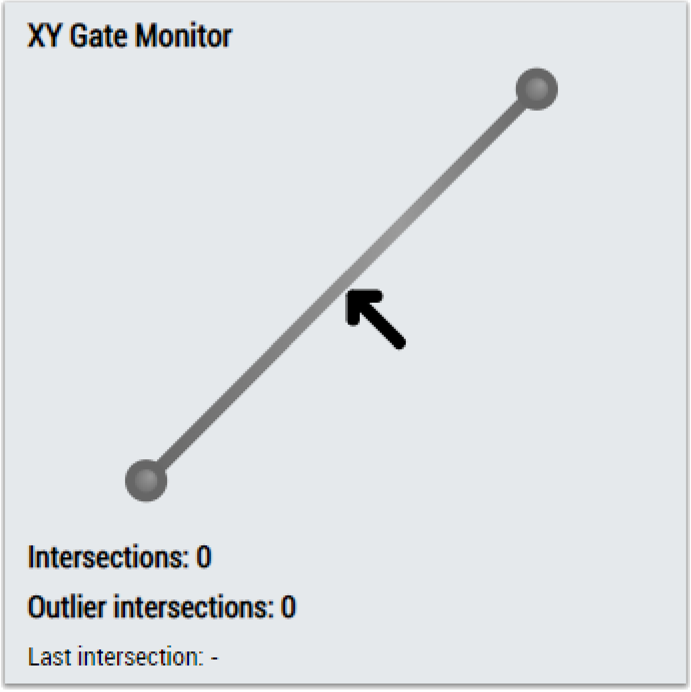
- **Gate Intersection:** Indicates whether there is currently a gate intersection.
- **Outlier Intersection:** Indicates whether there is currently an outlier intersection (gate projection intersection).
- **Position Intersection X:** X-position of the last intersection.

- **Position Intersection Y:** Y-position of the last intersection.
- **Count Gate Intersections:** Indicates the total number of gate intersections.
- **Count Outlier Intersections:** Indicates the total number of outlier intersections.
- **Time Intersection:** Indicates the time of the last intersection event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.
- **New Result: Indicates whether a new result was calculated or not.**
- **Executing:** Indicates whether the algorithm is active or inactive.
- **Classification:** Indicates the classification result. *OK* or *NOK*. The classification depends on the gate mode, as can be seen above.

Standard HMI Controls

For the XY Gate Monitor 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The XYGateMonitor control visualizes the output values Gate Intersection, Outlier Intersection, Count Gate Intersections, Count Outlier Intersections and Last Intersection as well as the direction of the intersection points.



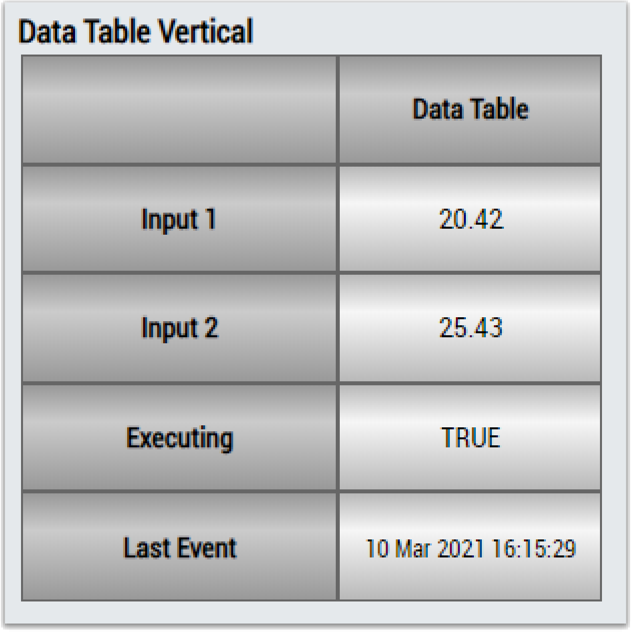
XY Gate Monitor

Intersections: 0

Outlier intersections: 0

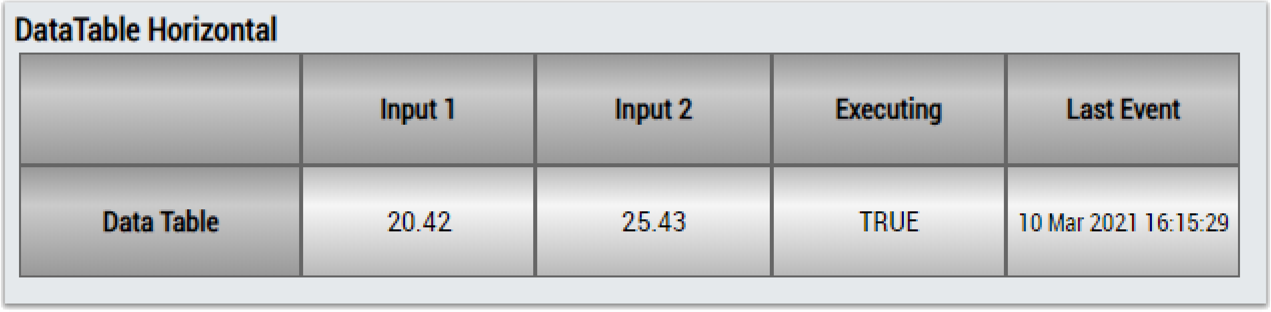
Last intersection: -

2. The Table Control or Multivalue Control visualizes all output values: Executing, Gate Intersection, Outlier Intersection, Position Intersection X, Position Intersection Y, Count Gate Intersections, Count Outlier Intersections, Last Intersection, Classification.



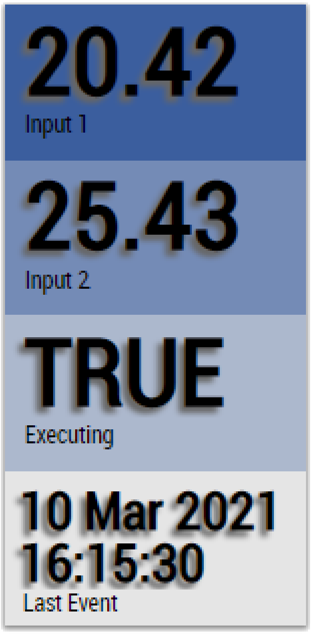
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

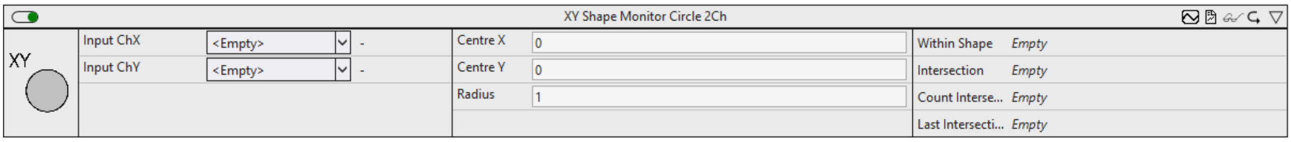
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the XY Gate Monitor 2Ch algorithm using the Mapping Wizard [▶ 432].

6.4.13.2 XY Shape Monitor Circle 2Ch



XY Shape Monitor Circle 2Ch

Input ChX	<Empty>	-	Centre X	0	Within Shape	Empty
Input ChY	<Empty>	-	Centre Y	0	Intersection	Empty
			Radius	1	Count Interse...	Empty
					Last Intersecti...	Empty

The *XY Shape Monitor Circle 2Ch* count the amount of intersections of an XY input with a specified circle shape.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Centre X:** X position of the circle center.
- **Centre Y:** Y position of the circle center.
- **Radius:** Radius of the circle.

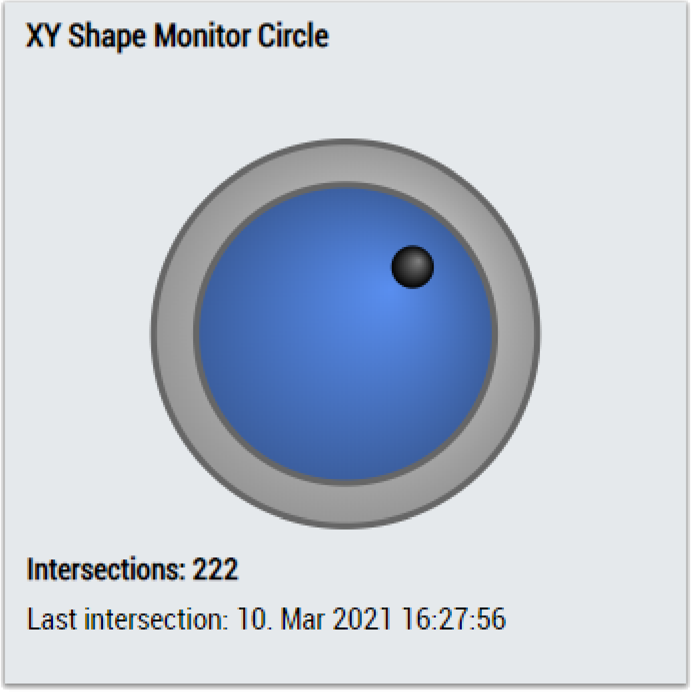
Output values

- **Within Shape:** Indicates whether the input signal is currently within the specified shape.
- **Intersection:** Indicates whether the input signal currently crosses the specified shape.
- **Count Intersections:** Counts the total number of intersections of input signal and shape.
- **Time Intersection:** Indicates the time of the last intersection event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the XY Shape Monitor Circle 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The XYShapeMonitor control visualizes the output values Within Shape, Intersection, Count Intersections and Last Intersection.

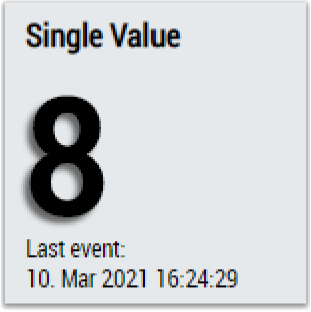


XY Shape Monitor Circle

Intersections: 222

Last intersection: 10. Mar 2021 16:27:56

2. The SingleValue control visualizes the output values Intersection and Last Intersection.

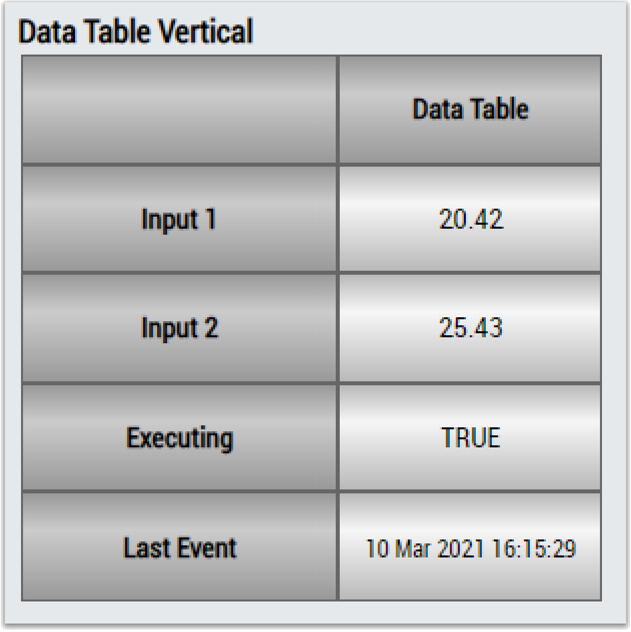


Single Value

8

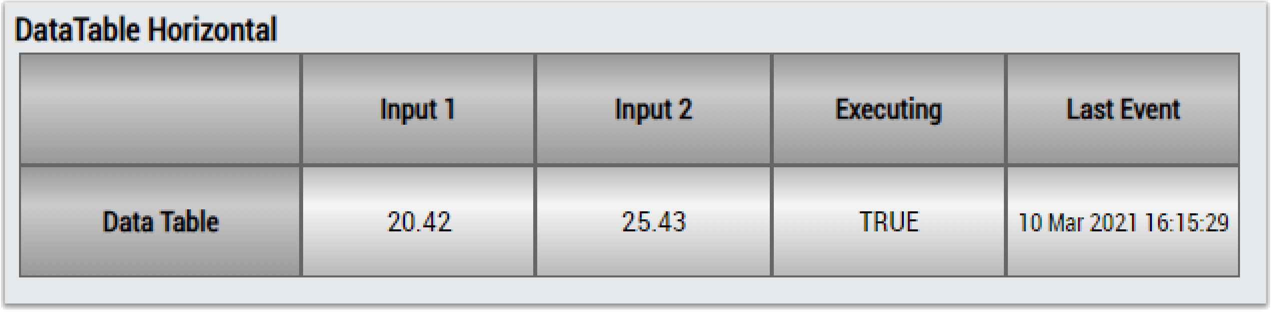
Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Within Shape, Intersection, Count Intersections, Last Intersection.



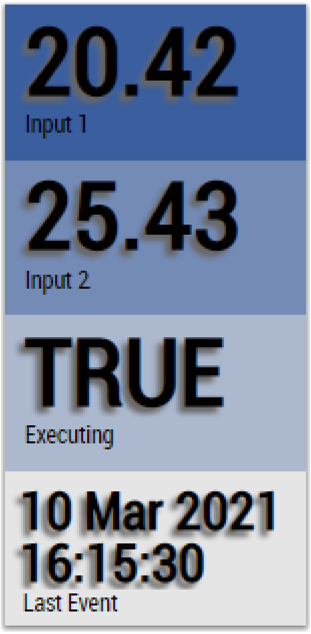
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

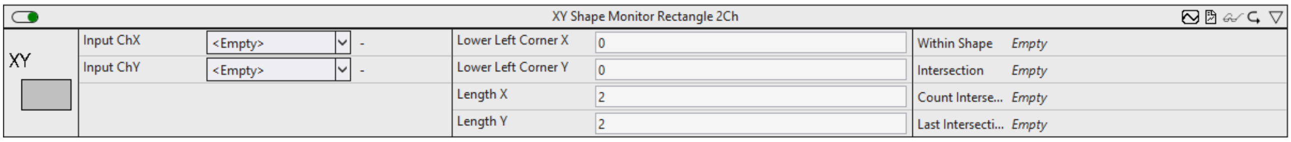
25.43
Input 2

TRUE
Executing

10 Mar 2021 16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the XY Shape Monitor Circle 2Ch algorithm using the Mapping Wizard [▶ 432].

6.4.13.3 XY Shape Monitor Rectangle 2Ch



XY Shape Monitor Rectangle 2Ch

XY

Input ChX	<Empty>	Lower Left Corner X	0	Within Shape	Empty
Input ChY	<Empty>	Lower Left Corner Y	0	Intersection	Empty
		Length X	2	Count Interse...	Empty
		Length Y	2	Last Intersecti...	Empty

The *XY Shape Monitor Rectangle 2Ch* count the amount of intersections of an XY input with a specified rectangle shape.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Lower Left Corner X:** X position of the lower left rectangle corner.
- **Lower Left Corner Y:** Y position of the lower left rectangle corner.
- **Length X:** Length of the rectangle in positive X direction.
- **Length Y:** Length of the rectangle in positive Y direction.

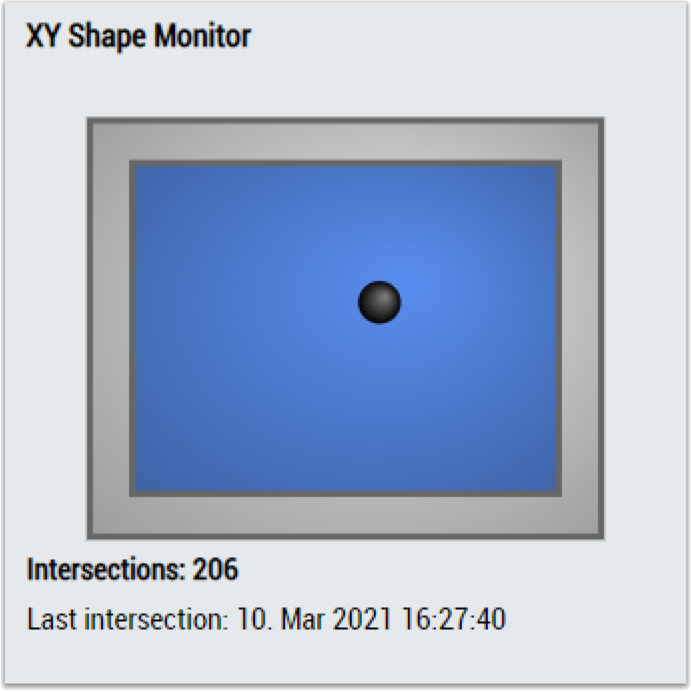
Output values

- **Within Shape:** Indicates whether the input signal is currently within the specified shape.
- **Intersection:** Indicates whether the input signal currently crosses the specified shape.
- **Count Intersections:** Counts the total number of intersections of input signal and shape.
- **Time Intersection:** Indicates the time of the last intersection event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the XY Shape Monitor Rectangle 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

1. The XYShapeMonitor control visualizes the output values Within Shape, Intersection, Count Intersections and Last Intersection.

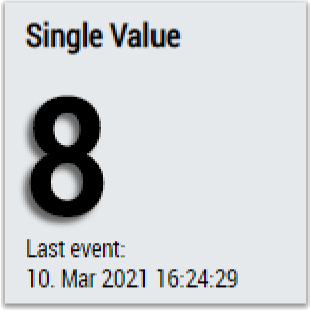


XY Shape Monitor

Intersections: 206

Last intersection: 10. Mar 2021 16:27:40

2. The SingleValue control visualizes the output values Intersection and Last Intersection.

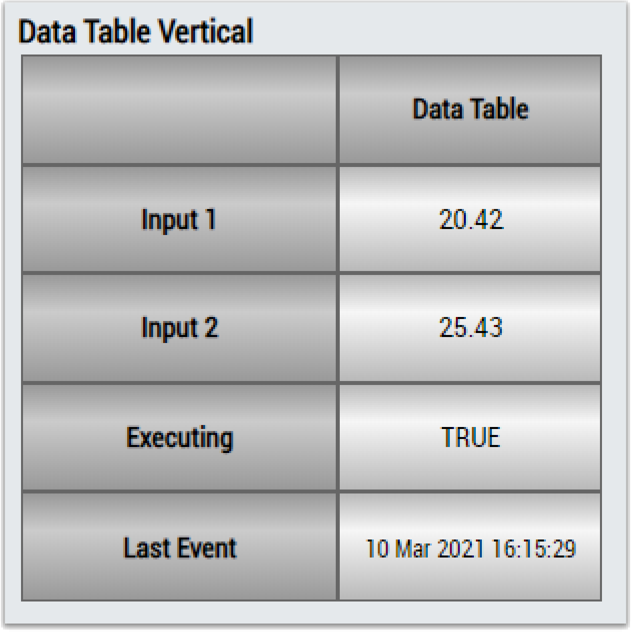


Single Value

8

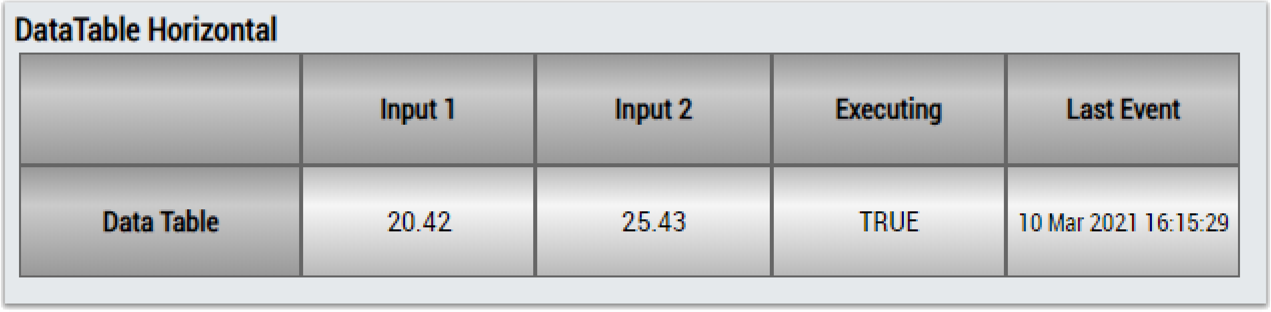
Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Within Shape, Intersection, Count Intersections, Last Intersection.



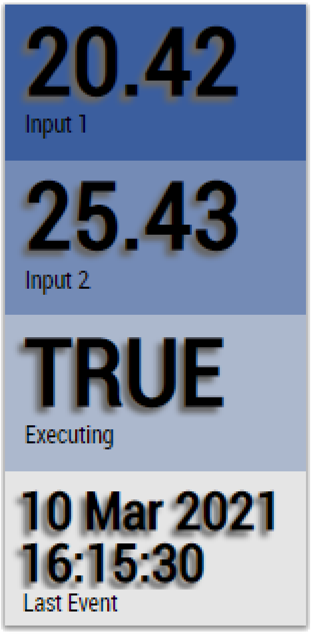
Data Table Vertical

	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

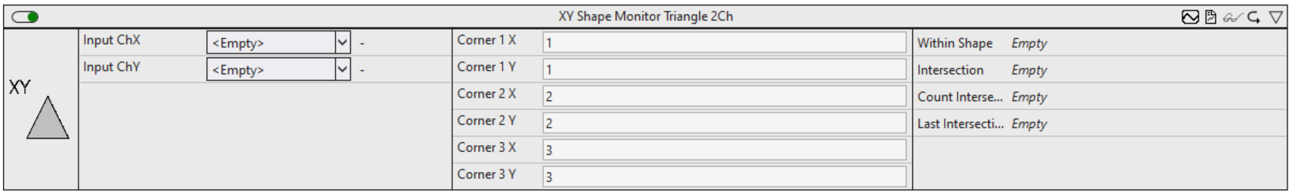
25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

Alternatively, customer-specific HMI controls can be mapped in the XY Shape Monitor Rectangle 2Ch algorithm using the Mapping Wizard [▶ 432].

6.4.13.4 XY Shape Monitor Triangle 2Ch



XY Shape Monitor Triangle 2Ch

XY

Input ChX	<Empty>	-
Input ChY	<Empty>	-

Corner 1 X	1
Corner 1 Y	1
Corner 2 X	2
Corner 2 Y	2
Corner 3 X	3
Corner 3 Y	3

Within Shape	*Empty*
Intersection	*Empty*
Count Interse...	*Empty*
Last Intersecti...	*Empty*

The *XY Shape Monitor Triangle 2Ch* counts the amount of intersections of an XY input with a specified triangle shape.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration Options

- **Corner 1 X:** X position of the first triangle corner.
- **Corner 1 Y:** Y position of the first triangle corner.
- **Corner 2 X:** X position of the second triangle corner.
- **Corner 2 Y:** Y position of the second triangle corner.
- **Corner 3 X:** X position of the third triangle corner.
- **Corner 3 Y:** Y position of the third triangle corner.

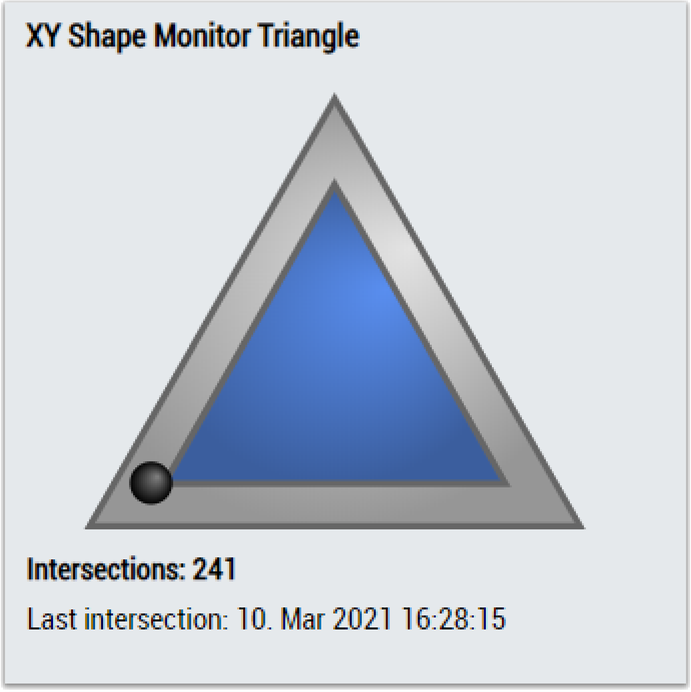
Output values

- **Within Shape:** Indicates whether the input signal is currently within the specified shape.
- **Intersection:** Indicates whether the input signal currently crosses the specified shape.
- **Count Intersections:** Counts the total number of intersections of input signal and shape.
- **Time Intersection:** Indicates the time of the last intersection event → the event can be dragged and dropped into the Scope chart to display it as a trigger-event.

Standard HMI Controls

For the XY Shape Monitor Triangle 2Ch algorithm, the following HMI controls are available for generating an Analytics Dashboard:

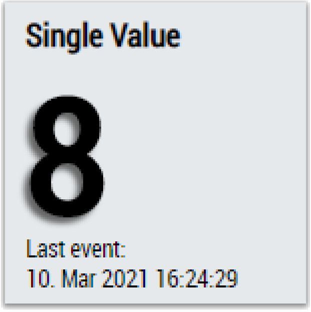
1. The XYShapeMonitor control visualizes the output values Within Shape, Intersection, Count Intersections and Last Intersection.



Intersections: 241

Last intersection: 10. Mar 2021 16:28:15

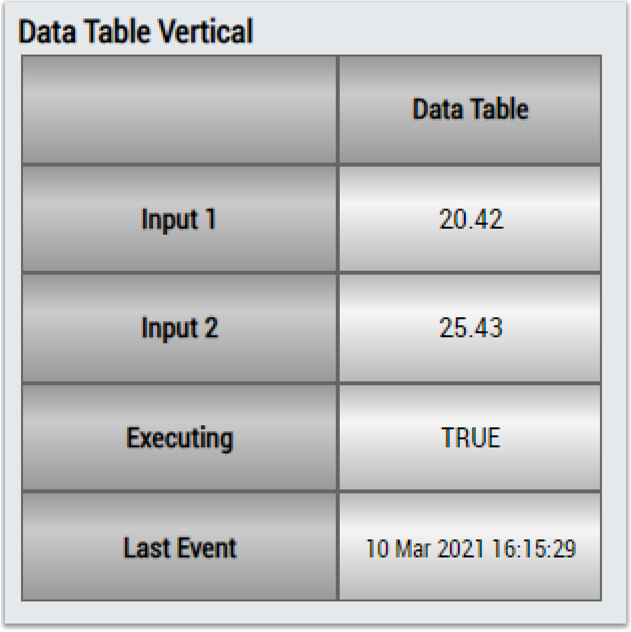
2. The SingleValue control visualizes the output values Intersection and Last Intersection.



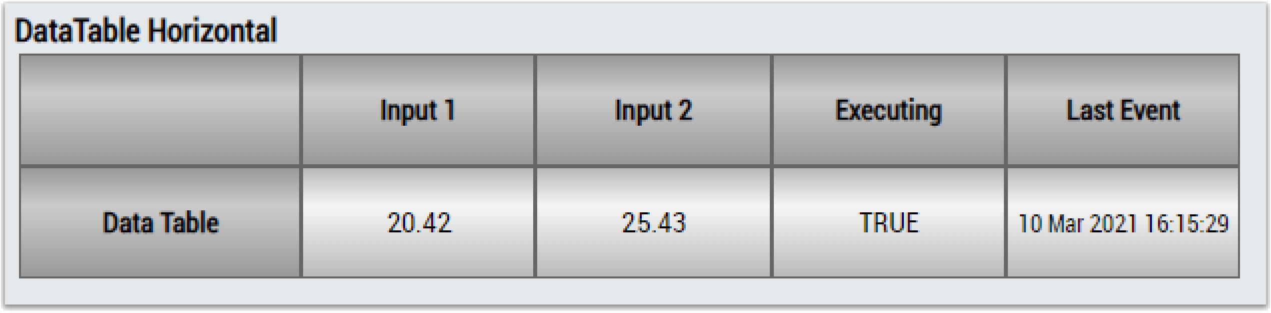
8

Last event:
10. Mar 2021 16:24:29

3. The Table Control or Multivalue Control visualizes all output values: Within Shape, Intersection, Count Intersections, Last Intersection.

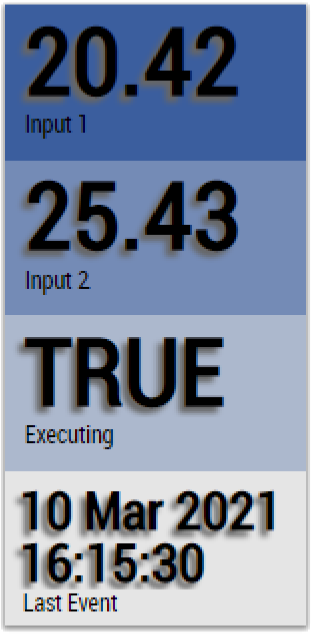


	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29



DataTable Horizontal

	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29



20.42
Input 1

25.43
Input 2

TRUE
Executing

10 Mar 2021
16:15:30
Last Event

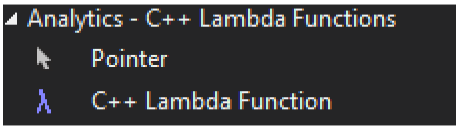
Alternatively, customer-specific HMI controls can be mapped in the XY Shape Monitor Triangle 2Ch algorithm using the Mapping Wizard [▶ 432].

6.4.14 Analytics - extension of the algorithms

In order to extend the algorithms of the TwinCAT Analytics Library, which are available in the TwinCAT Analytics Workbench configurator or in the TwinCAT Analytics Service Tool, two extension options are available. On the one hand, Analytics lambda functions can be used to develop user-specific algorithms and integrate them into the Analytics Workflow. On the other hand, algorithms from other libraries can extend the existing algorithms.

6.4.14.1 C++ lambda functions

The C++ lambda functions offer the possibility to develop user-specific algorithms and integrate them into the TwinCAT Analytics Workflow. To guide through the development of the algorithm, a lambda template is available in the Visual Studio® Toolbox.



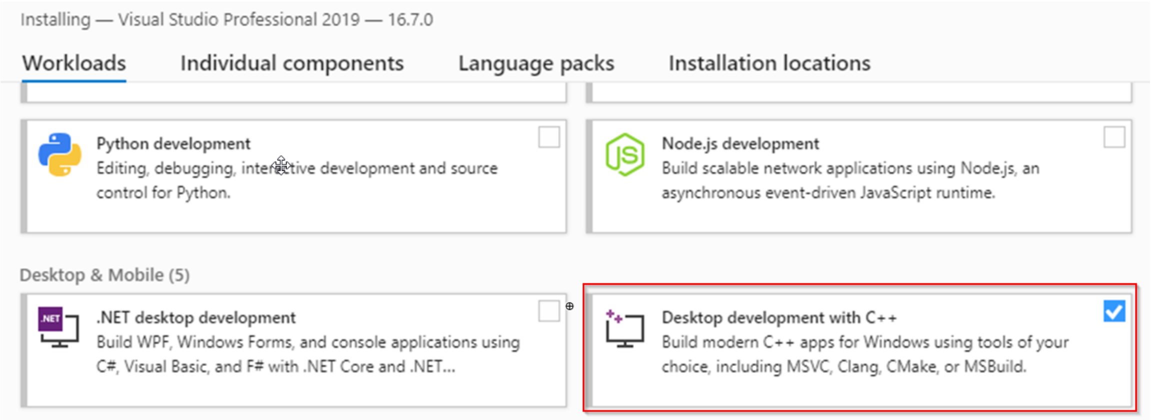
The Lambda Template can be used to configure, program and publish the lambda function. After the successful development of the lambda function, the algorithm is located in the toolbox group "Analytics - C++ Lambda Functions". In the further course, all subsequent steps of the Analytics workflow are available. For example, the lambda function can be linked to the HMI controls. After deployment, the lambda functions are available in the TwinCAT Analytics Runtime and are displayed on the generated HMI One Click Dashboard after appropriate configuration.

6.4.14.1.1 Requirements

Overview of minimum requirements

The following minimum requirements must be met for the implementation and debugging of TwinCAT 3 C++ modules.

The following must be installed on the engineering PC:

- Microsoft Visual Studio® 2017 or 2019 Professional / Enterprise.
  - When installing Visual Studio®, the **Desktop development with C++** option must be additionally selected, as this option is not selected with the automatic installation:
  - 

    Installing — Visual Studio Professional 2019 — 16.7.0

    Workloads | Individual components | Language packs | Installation locations

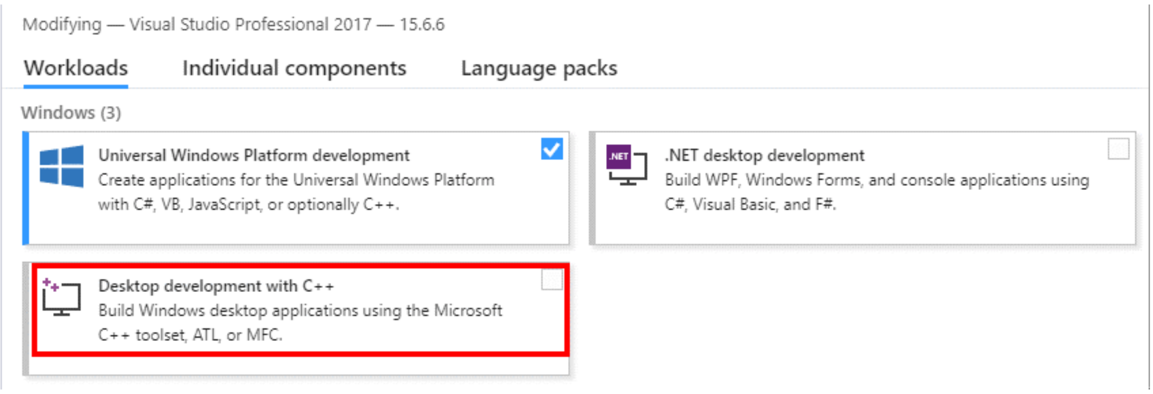
    Python development □
    Editing, debugging, interactive development and source control for Python.

    Node.js development □
    Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

    Desktop & Mobile (5)

    .NET desktop development □
    Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET Core and .NET...

    Desktop development with C++ ☑
    Build modern C++ apps for Windows using tools of your choice, including MSVC, Clang, CMake, or MSBuild.

    Modifying — Visual Studio Professional 2017 — 15.6.6

    Workloads | Individual components | Language packs

    Windows (3)

    Universal Windows Platform development ☑
    Create applications for the Universal Windows Platform with C#, VB, JavaScript, or optionally C++.

    .NET desktop development □
    Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.

    Desktop development with C++ □
    Build Windows desktop applications using the Microsoft C++ toolset, ATL, or MFC.
- TwinCAT 3 installation (XAE engineering)
- XAE-Shell is sufficient for the integration and use of existing binary C++ modules in a TwinCAT 3 PLC environment (Visual Studio® is not required.)

On the runtime PC:

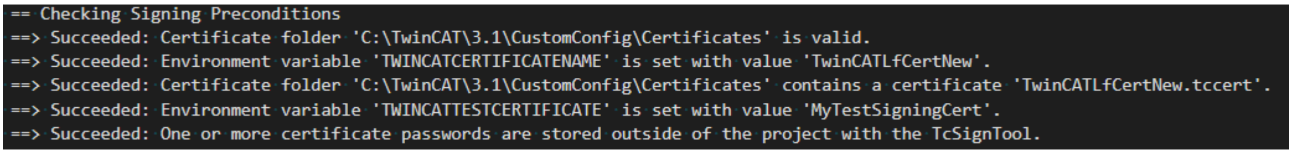
- IPC or Embedded CX PC with one of the operating systems Windows 7 / 10 or TwinCAT/BSD. Both 32 and 64-bit versions are supported.
- Microsoft Visual Studio® **does not** have to be installed.
- TwinCAT 3.1 installation (XAR runtime)

6.4.14.1.2 Preparation - only once

A PC for engineering lambda functions and thus TwinCAT C++ modules must be prepared. You only have to carry out these steps once per system:

- Configure the TwinCAT Basis as well as the configuration and platform toolbar.
- Create a test certificate to sign the modules
- Use the TcSignTool to store the certificate password outside the project
- Create an environment variable with the certificate name

When publishing the C++ Lambda function, an overview of the corresponding signature requirements is output in the log file. The log file can be searched for "Checking Signing Preconditions" and the corresponding error handling can be taken. A successful signing of the drivers looks like this:



```
== Checking Signing Preconditions
==> Succeeded: Certificate folder 'C:\TwinCAT\3.1\CustomConfig\Certificates' is valid.
==> Succeeded: Environment variable 'TWINCATCERTIFICATENAME' is set with value 'TwinCATLfCertNew'.
==> Succeeded: Certificate folder 'C:\TwinCAT\3.1\CustomConfig\Certificates' contains a certificate 'TwinCATLfCertNew.tccert'.
==> Succeeded: Environment variable 'TWINCATTESTCERTIFICATE' is set with value 'MyTestSigningCert'.
==> Succeeded: One or more certificate passwords are stored outside of the project with the TcSignTool.
```

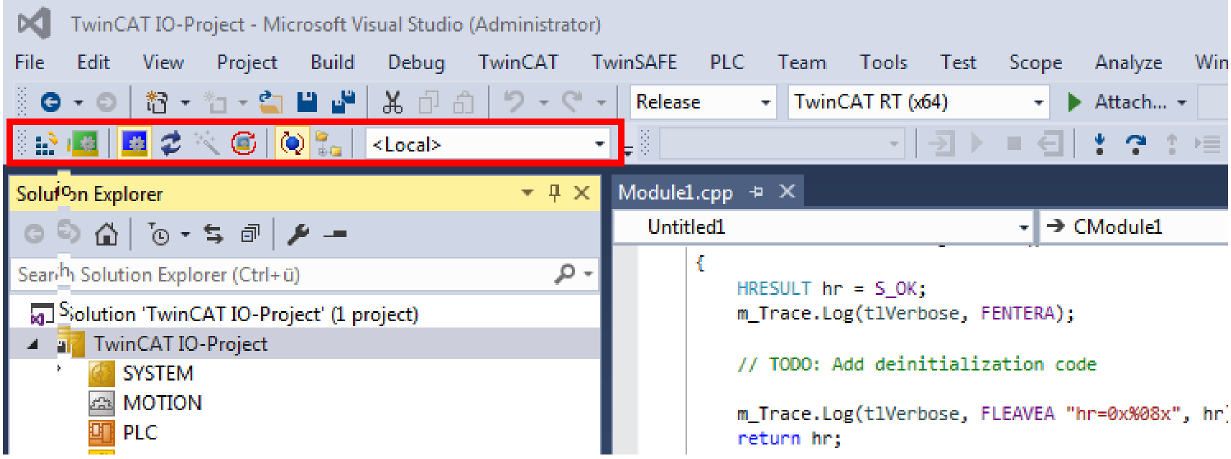
6.4.14.1.2.1 Visual Studio - TwinCAT XAE Base toolbar

Efficient engineering through TwinCAT XAE base toolbar

TwinCAT 3 integrates its own toolbar in the Visual Studio menu for better efficiency. It assists you in the creation of C++ projects. This toolbar is automatically added to the Visual Studio menu by the TwinCAT 3 setup. If you wish to add it manually, however, do the following:

1. Open the **View** menu and select **Toolbars\TwinCAT XAE Base**

⇨ The selected toolbar appears below the menu.


TwinCAT IO-Project - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug TwinCAT TwinSAFE PLC Team Tools Test Scope Analyze Win
Release
TwinCAT RT (x64)
Attach...
<Local>
Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'TwinCAT IO-Project' (1 project)
TwinCAT IO-Project
SYSTEM
MOTION
PLC
Module1.cpp
Untitled1
CModule1

```
{
    HRESULT hr = S_OK;
    m_Trace.Log(tlVerbose, FENTERA);

    // TODO: Add deinitialization code

    m_Trace.Log(tlVerbose, FLEAVEA "hr=0x%08x", hr
    return hr;
```

6.4.14.1.2.2 Driver signing

TwinCAT C++ modules must be signed with a certificate so that they can be executed.

The signature ensures that only C++ software whose origin can be traced is executed on productive systems.

For test purposes, certificates that cannot be verified can be used for signing. However, this is only possible if the operating system is in test mode so that these certificates are not used on productive systems.

Engineering requires no signing

Only the execution requires certificates - the engineering does not.

There are two ways to load modules. For this purpose, different certificates are used for signing:

- TwinCAT: the C++ modules are loaded by the TwinCAT runtime system and must be signed with a TwinCAT user certificate.
    - With TwinCAT 3.1. 4024 and higher, this procedure is also available.
    - This procedure is required to perform new functions such as Versioned C++ Projects and thus also the Online Change.
    - Required for TwinCAT/BSD.
- (For <4024.0, no longer recommended for new projects. Existing projects should be migrated) Operating system: the C++ modules are loaded as normal kernel drivers and must therefore also have a signature.

- ◦ With TwinCAT 3.1. 4022 or earlier, only this procedure is available.
- ◦ Windows 7 (Embedded) x86 (32bit) does not require signing.
- ◦ Cannot be used with TwinCAT/BSD.

Since a published module should be executable on various PCs, signing is always necessary for publishing.

Organizational separation of development and production software

Beckhoff recommends working organizationally with (at least) two certificates.

1. A certificate which is not countersigned, thus the test mode is needed for the development process. This certificate can also be issued individually by each developer.
2. Only the software that has passed the corresponding final tests is signed by a countersigned certificate. This software can thus also be installed on machines and delivered.

Such a separation of development and operation ensures that only tested software runs on productive systems.

6.4.14.1.2.2.1 TwinCAT

Versioned C++ projects are stored as binary in a TMX file (TwinCAT Module Executeable).

For the implementation of TwinCAT 3 C++ modules, this compiled, executable TMX file must be signed with a TwinCAT user certificate if it is to be loaded by the TwinCAT Runtime.

Signing

TwinCAT TMX files can only be loaded after a successful signing.

NOTICE

Signing on 32-bit and 64-bit systems

In contrast to the operating system signing, TwinCAT signing is intended for both 32-bit and 64-bit systems. Thus, the test mode is also assumed for a test signing on 32-bit systems.

For signing a TMX file, a TwinCAT user certificate is required [▶ 273], which is configured accordingly in the project for signing.

Test signing

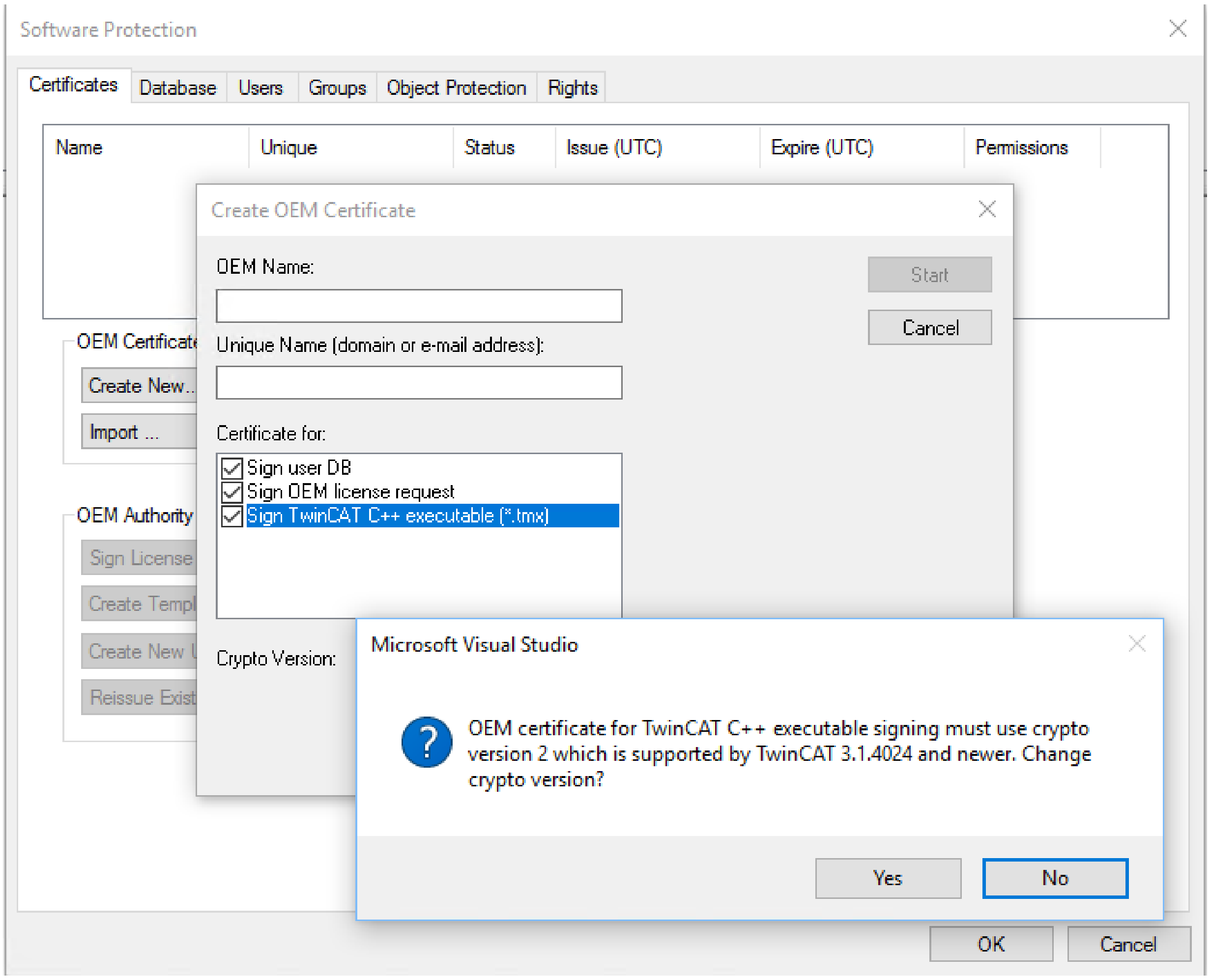
The user certificate can be created locally in TwinCAT. As long as it is not countersigned by Beckhoff, it is necessary to activate the test mode, as described here [▶ 273].

As soon as the TwinCAT user certificate has been countersigned by Beckhoff [▶ 275], the test mode can be dispensed with accordingly. It can be deactivated in the same way as it is activated.

6.4.14.1.2.2.1.1 Test signing

The test signing for TwinCAT can be carried out with the same TwinCAT user certificate as for the actual delivery (see Request TwinCAT 3 user certificate [▶ 276]).

1. For test operation, e.g. during software development, the creation of a TwinCAT user certificate, as described Creation of the Certificate Request file for TC0008 [▶ 277], is sufficient. Make sure that you select the purpose "Sign TwinCAT C++ executable (*.tmx)". For this the Crypto version 2 is required, a message appears.



Certificate for:
☑ Sign user DB
☑ Sign OEM license request
☑ Sign TwinCAT C++ executable (*.tmx)

Microsoft Visual Studio

OEM certificate for TwinCAT C++ executable signing must use crypto version 2 which is supported by TwinCAT 3.1.4024 and newer. Change crypto version?

On XAR (and XAE, if it is a local test), activate the test mode so that the operating system can accept the self-signed certificates. This can be done on both engineering systems (XAE) and runtime systems (XAR).

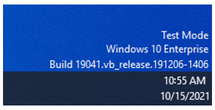
For Windows

Use the administrator prompt to execute the following:

```
bcdedit /set testsigning yes
```

and reboot the target system.
You may have to switch off "SecureBoot" for this, which can be done in the bios.

If test signing mode is enabled, this is displayed at the bottom right of the desktop. The system now accepts all signed drivers for execution.



Test Mode
Windows 10 Enterprise
Build 19041.vb_release.191206-1406
10:55 AM
10/15/2021

For TwinCAT/BSD

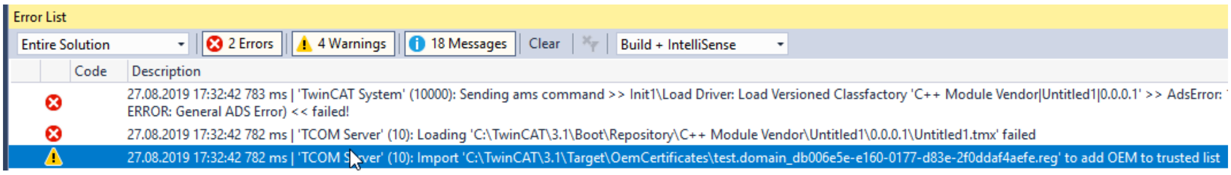
In the file `/usr/local/etc/TwinCAT/3.1/TcRegistry.xml` enter „`<Value Name="EnableTestSigning" Type="DW">1</Value>`" under Key "`System`".

```
<Key Name="System">
  <Value Name="RunAsDevice" Type="DW">1</Value>
  <Value Name="RTimeMode" Type="DW">0</Value>
  <Value Name="AmsNetId" Type="BIN">052445B00101</Value>
  <Value Name="LockedMemSize" Type="DW">33554432</Value>
  <Value Name="EnableTestSigning" Type="DW">1</Value>
</Key>
```

Then restart the TwinCAT System Service:
`doas service TcSystemService restart`

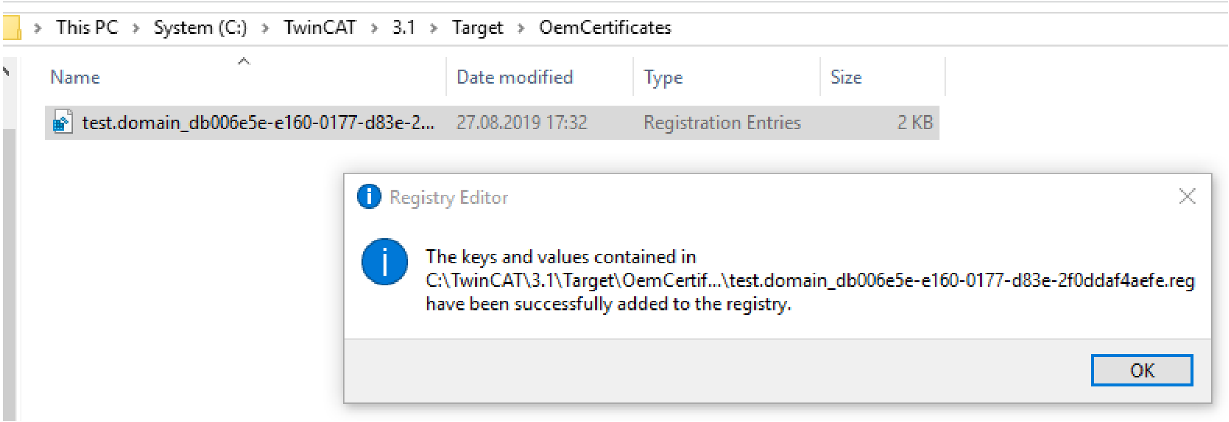
After the respective procedure, the system accepts all signed drivers for execution.

1. During the first activation (Activate Configuration) with a TwinCAT user certificate, the target system detects that the certificate is not trusted and the activation process is aborted:



```
27.08.2019 17:32:42 783 ms | 'TwinCAT System' (10000): Sending ams command >> Init1\Load Driver: Load Versioned Classfactory 'C++ Module Vendor|Untitled1|0.0.0.1' >> AdsError: 
ERROR: General ADS Error) << failed!
27.08.2019 17:32:42 782 ms | 'TCOM Server' (10): Loading 'C:\TwinCAT\3.1\Boot\Repository\C++ Module Vendor\Untitled1\0.0.0.1\Untitled1.tmx' failed
27.08.2019 17:32:42 782 ms | 'TCOM Server' (10): Import 'C:\TwinCAT\3.1\Target\OemCertificates\test.domain_db006e5e-e160-0177-d83e-2f0ddaf4aefe.reg' to add OEM to trusted list
```

   For Windows:
   A local user with administration rights can trust the certificate via the created REG file by simply executing it:


This PC › System (C:) › TwinCAT › 3.1 › Target › OemCertificates
Name
Date modified
Type
Size
test.domain_db006e5e-e160-0177-d83e-2...
27.08.2019 17:32
Registration Entries
2 KB
Registry Editor
OK

The keys and values contained in
C:\TwinCAT\3.1\Target\OemCertif...\test.domain_db006e5e-e160-0177-d83e-2f0ddaf4aefe.reg
have been successfully added to the registry.

   For TwinCAT/BSD:
   If the "Tcimportcert" package is not installed, install it: `pkg install tcimportcert`
   Trust the certificate via `doas tcimportcert /usr/local/etc/TwinCAT/3.1/Target/OemCertificates/<CreatedFile>.reg`.
   Then restart the TwinCAT System Service or reboot the system:
   `doas service TcSystemService restart`

   ⇨ This process only enables C++ modules with a signature from the trusted TwinCAT user certificates to run.

2. Following this process you can use the TwinCAT user certificate for signing with the test mode of the operating system.
   This is configured in the project properties.
   Use the TcSignTool [▶ 286] to avoid storing the password of the TwinCAT user certificate in the project, where it would also end up in version management, for example.

If you want to use the TwinCAT user certificate without TestMode for delivery, you must have the certificate countersigned by Beckhoff [▶ 275].

6.4.14.1.2.2.1.2 Signed TwinCAT user certificates for delivery without test mode

System requirements

- Min. TwinCAT 3.1 Build 4024
- Min. Windows 10 or TwinCAT/BSD (on the target system)

With TwinCAT Build 4024, Beckhoff offers existing customers the issuing of a "TwinCAT 3 OEM user certificate", which can be used for signing TMX files created with TwinCAT 3 in C++.

- This certificate requires secure validation of the applicant data, since it is used in the Windows environment. TwinCAT 3 user certificates must therefore be officially ordered for validation of the address and contact data, and are only issued to existing Beckhoff customers.
- Order number: **TC0008**
- The issuing of this TwinCAT 3 user certificate is free of charge.
- Directory for saving the certificate: **C:\TwinCAT\3.1\CustomConfig\Certificates**

The TwinCAT 3 user certificate is not required for using the TwinCAT 3 TMX files

The TwinCAT 3 user certificate is used exclusively for the one-time signing of the TMX files and is not required for the use of the TMX files signed with it.

On which computers is the TwinCAT 3 user certificate TC0008 required?

The TwinCAT 3 user certificate should be located exclusively on the engineering computer on which the TMX files are signed - i.e. **NOT** on each target system.

Validity of the TwinCAT 3 user certificate

The validity of the TwinCAT 3 user certificate is limited to two years for security reasons.

What happens if the certificate has expired?

You can no longer sign new TMX files.
However, the use of already signed TMX files is still possible without any restrictions.

You can apply for a renewal of your certificate before the expiry of the two years (and even after that).

To extend a TwinCAT 3 user certificate, the same process applies as for requesting a new certificate. In this case, the certificate must also be ordered (the order numbers for a certificate extension are the same as for a new certificate request).

In contrast to a new certificate, you do not generate a new OEM Certificate Request File but send your existing certificate to the Beckhoff certificate section for renewal. Please notify us in the email that this is a certificate extension and not a new issue. Otherwise, the same criteria apply regarding the content of the email as for the application for a new certificate.

The existing certificate receives a new expiration date, is then re-signed and is valid for another 2 years.

The newly signed certificate is thus fully compatible with the original version.

6.4.14.1.2.2.1.2.1 Request TwinCAT 3 user certificate

Overview of the ordering and validation process

An official order is required to request a TwinCAT user certificate.

- Order number: **TC0008** (TwinCAT 3 Certificate Extended Validation)
- The issuing (and renewal) of a TwinCAT 3 user certificate is free of charge.
- Since a TwinCAT 3 user certificate is a digital ID card, verification of the inquirer's contact data is required according to the usual market standards.
- A TwinCAT 3 user certificate is therefore only issued to existing Beckhoff customers.

Overview of the ordering and validation process

Your email address must be a company email account (freemailers such as GMail or similar are not permitted) and correspond with the company name of the inquirer.

1. Contact your Beckhoff sales contact and announce the request of a TwinCAT 3 OEM certificate. Order "TC0007" or "TC0008".
2. Important: as the inquirer, please provide your contact details as the delivery address (= contact name and email address) and the area of use of the certificate (company name, address).
3. The contact details provided in the order will be verified and you (the inquirer named in the delivery address) will be contacted by Beckhoff Sales.
4. When requesting a new OEM certificate, Creation of the Certificate Request file for TC0008 [▸ 277].
5. Determine the "File Fingerprint" of the OEM certificate file using TwinCAT Engineering (see Determining the file fingerprint of the OEM certificate file [▸ 280]). Please inform the Beckhoff sales contact of this File Fingerprint as part of your contact data verification. The transmission of the File Fingerprint must be done by a different communication channel than the one used for sending the OEM certificate request file.
6. Now send the "OEM certificate file" to the Beckhoff sales contact.
7. After signing the certificate file at the Beckhoff headquarters, you will receive it by e-mail from your contact person.

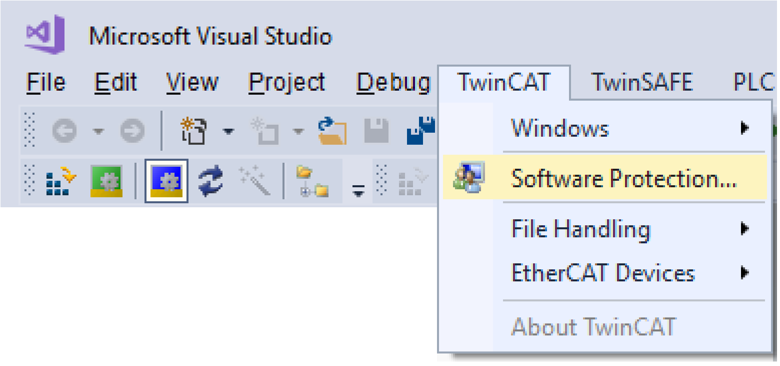
Please note that it may take a few days to validate your contact details and issue the certificate.

6.4.14.1.2.2.1.2.2 Creation of the Certificate Request file for TC0008

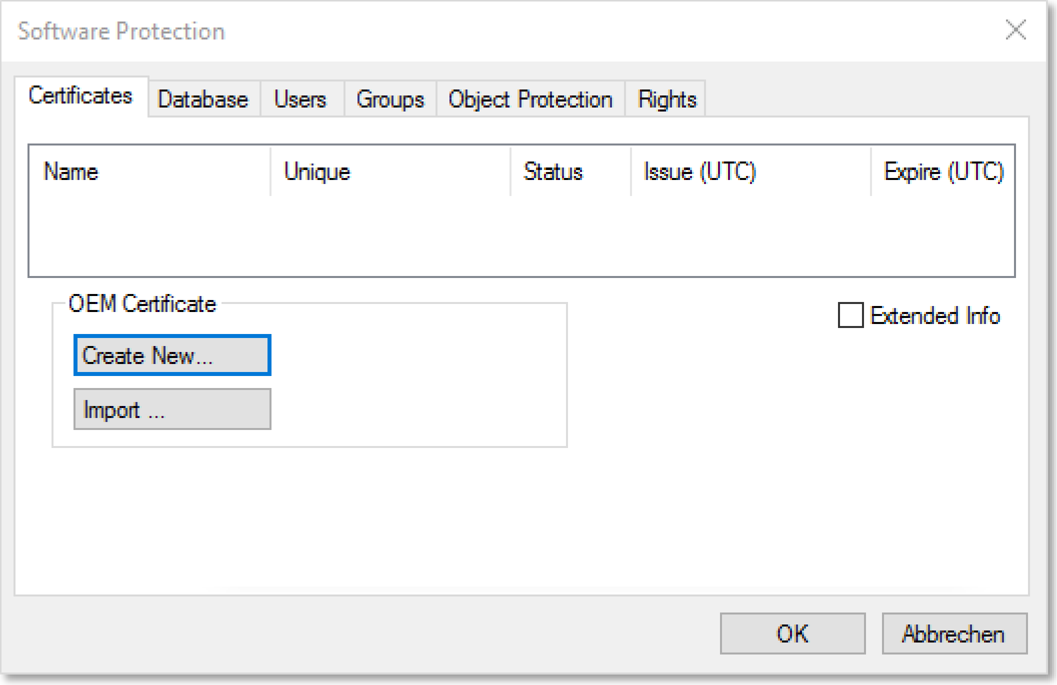
System requirements

- Min. TwinCAT 3.1 Build 4024
- Min. Windows 10 or TwinCAT/BSD (on the target system)

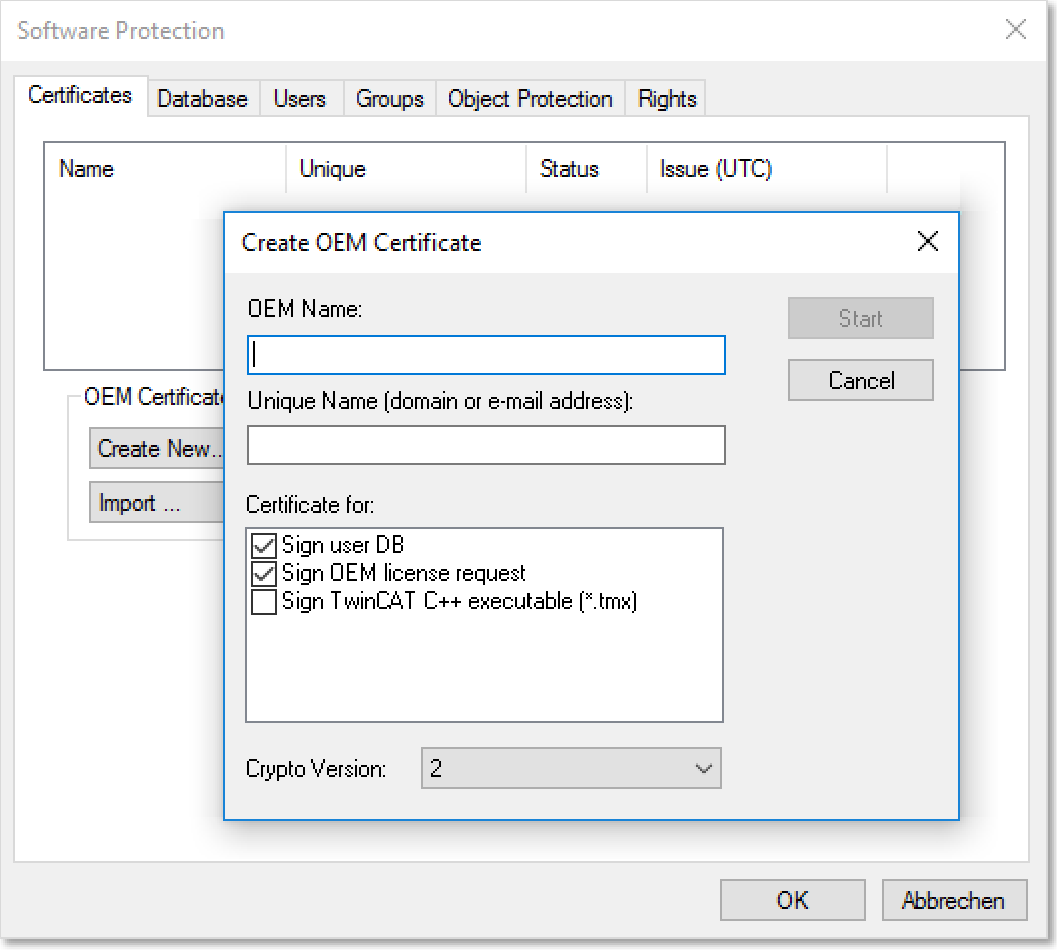
1. Call up the Software Protection configurator. To do this, select the menu item **Software Protection** in the main menu below the item **TwinCAT**:



2. In the window that opens, select the **Certificates** tab.
   Click **Create New....**:



3. The **Create OEM Certificate** input window opens:



OEM Name:

Unique Name (domain or e-mail address):

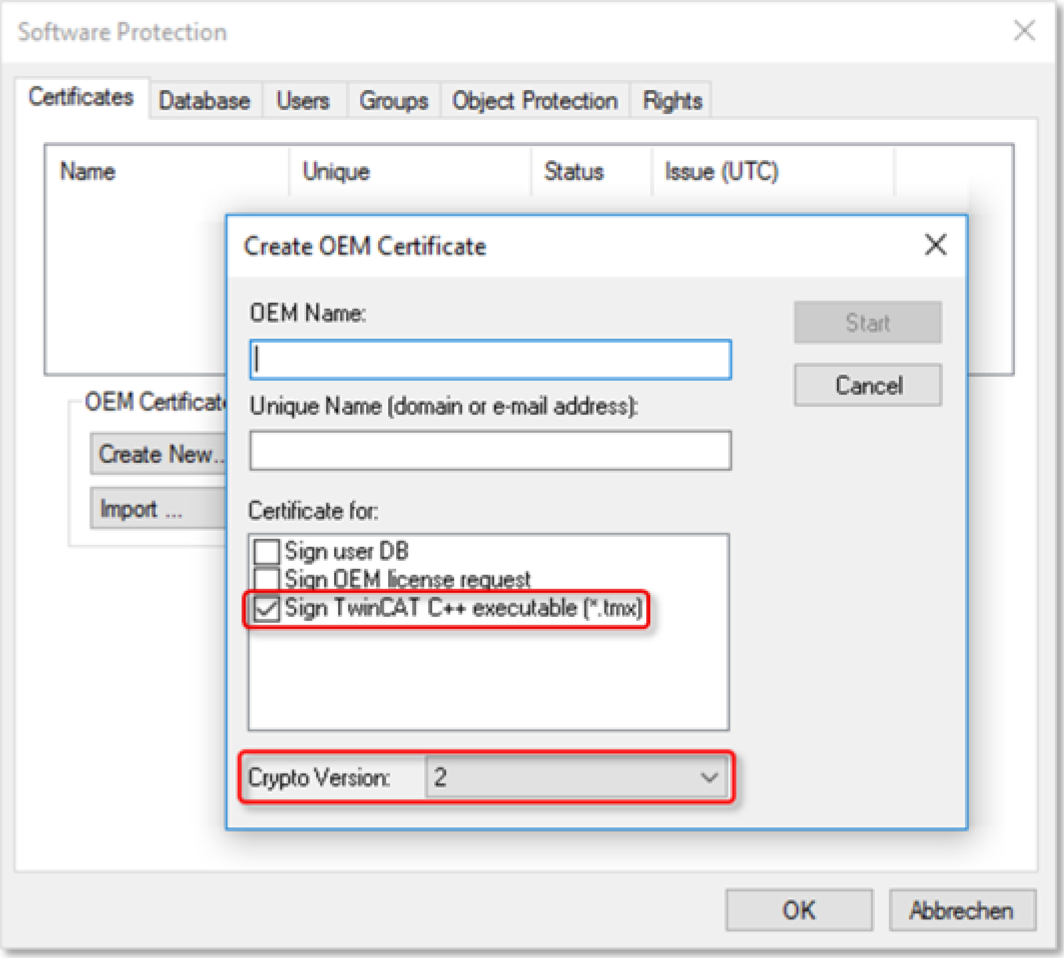
Certificate for:

☑ Sign user DB
☑ Sign OEM license request
□ Sign TwinCAT C++ executable (*.tmx)

Crypto Version: 2

4. Enter the required data:
   - Enter your company name in the **OEM Name** text field. The name must have a clear reference to your company or your business unit.

- Enter a **Unique Name**. The "OEM Unique Name" must be a unique name that uniquely identifies the owner of the certificate worldwide, preferably the URL of your company's website or your email address. The email address must be a company email address, i.e. it must be possible to assign it unambiguously to your company.
- Check the checkbox **Sign TwinCAT C++ executables**:



☐ Sign user DB
☐ Sign OEM license request
☑ Sign TwinCAT C++ executable (*.tmx)

Crypto Version: 2

If you only want to sign TwinCAT driver software with this certificate, uncheck the other two checkboxes. (These are only used in the PLC area)

- Make sure that Crypto version 2 (for the encrypted content of the certificate content) is set. (standard setting)

5. Once you have entered the data, click **Start** and select a directory to save the file.
You can simply accept the suggested directory "c:\twincat\3.1\customconfig\certificates". You need the newly created file in this directory in order to be able to read out the "File Fingerprint" for this file [▸ 280] **in a subsequent step.**

⇨ A dialog for selecting a password for the OEM Private Key opens.

6. Issue a password for the OEM Private Key**.**

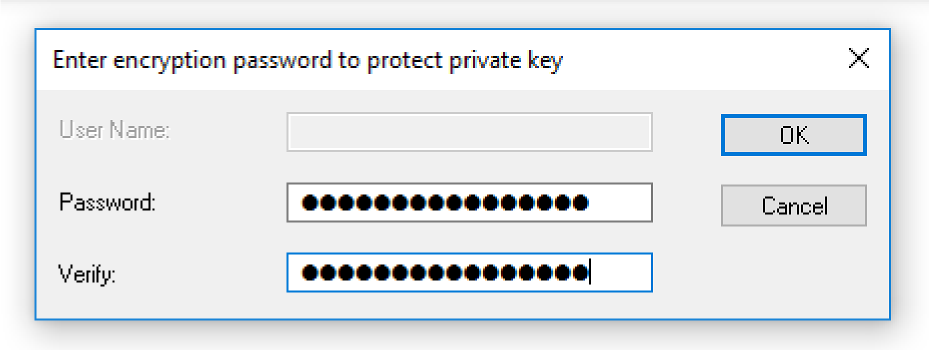
Important: Password security!

Be sure to use a strong password for your certificate!
Protect your password with suitable measures so that it cannot fall into unauthorized hands!

Password cannot be restored if lost

Beckhoff is unable to recover or reset your password. If you forget or lose the password for your certificate, you can no longer use it and have to request a new certificate.

7. Confirm the password by entering it again and close the dialog with **OK**.



Enter encryption password to protect private key

User Name:

Password: ●●●●●●●●●●●●●●●●●

Verify: ●●●●●●●●●●●●●●●●●

⇨ The file is saved.

The "Certificate Request File" generated in this way must now be signed by the Beckhoff certificate section in order to be valid. The procedure is described in chapter Requesting a certificate [▶ 276].

6.4.14.1.2.2.1.2.3 Determining the file fingerprint of the OEM certificate file

You need this functionality to request a **TwinCAT OEM Certificate Extended Validation** (TC0008).

System requirements

This functionality requires TwinCAT 3.1 Build 4024 of higher.

The OEM Certificate Request File becomes the TwinCAT OEM certificate once it is signed by Beckhoff. The files do not differ except for this signature. For this reason, the term "TwinCAT OEM certificate file" is used for both file versions in the following sections.

Reading the "file fingerprint" of an OEM certificate file via TwinCAT 3 Engineering

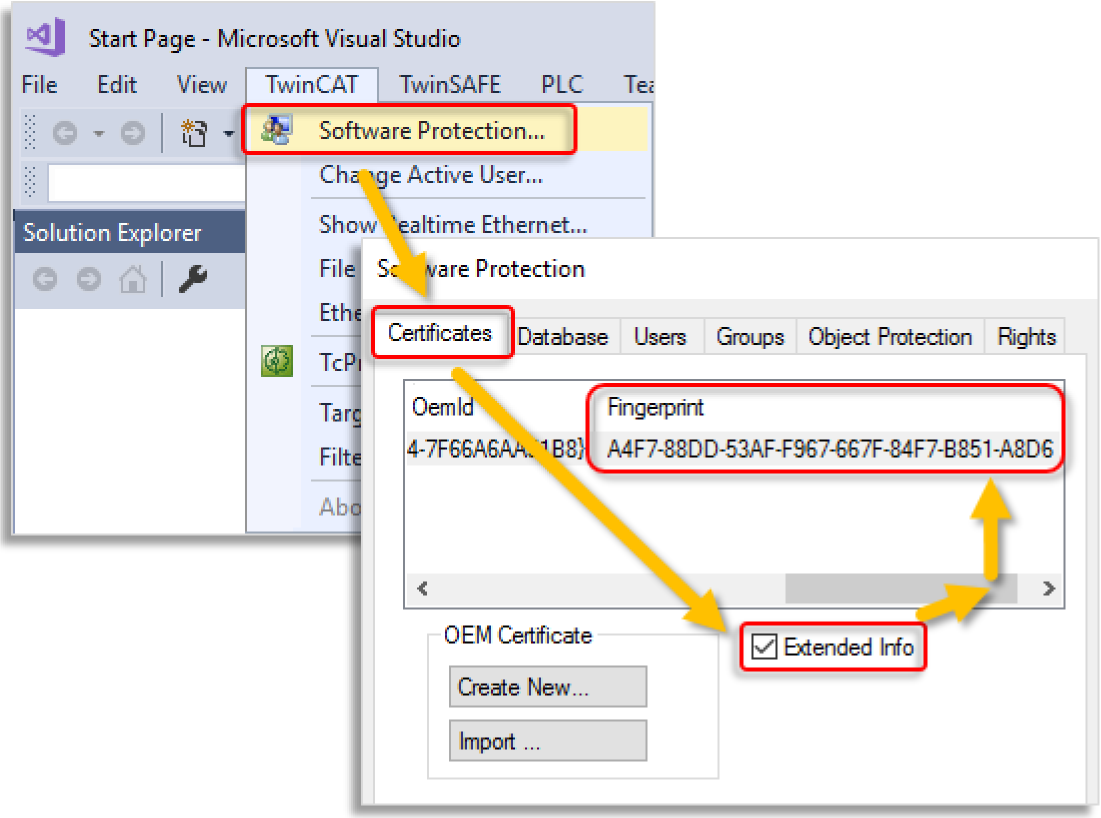
For this function it is necessary that the OEM certificate file is located in this directory: "c:\twincat\3.1\customconfig\certificates".

This directory contains your OEM certificate, if you already have a certificate and want to renew it.

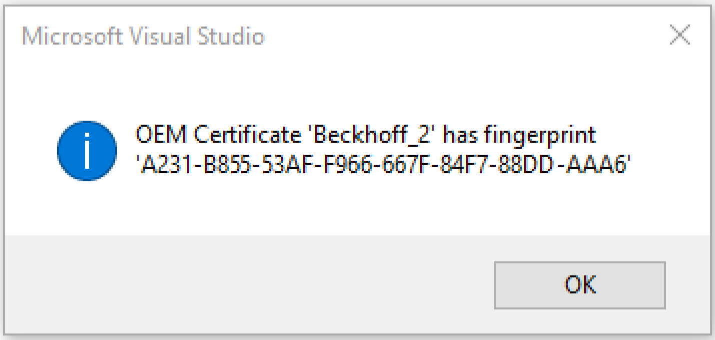
If you did not change the suggested directory when creating the "OEM Certificate Request File", the file is already in this directory.

Procedure:

1. Call up the TwinCAT 3 Software Protection configurator.



2. Select the **Certificates** tab.
3. Check the **Extended Info** checkbox.
4. In the window scroll to the right until you see the **Fingerprint** column. (As an alternative, you can simply double-click the certificate line. The Fingerprint file is then displayed in a pop-up window:



OEM Certificate 'Beckhoff_2' has fingerprint 'A231-B855-53AF-F966-667F-84F7-88DD-AAA6'

The shortcut [Ctrl] + [C] can be used to copy the fingerprint data from the message window to the Windows clipboard.

6.4.14.1.2.2.1.2.4 Saving the signed TwinCAT user certificate

Recommended directory for saving the certificate: **C:\TwinCAT\3.1\CustomConfig\Certificates**

System requirements

- Min. TwinCAT 3.1 Build 4024
- Min. Windows 10 or TwinCAT/BSD (on the target system)

The TwinCAT 3 user certificate is not required for using the TwinCAT 3 TMX files

The TwinCAT 3 user certificate is used exclusively for the one-time signing of the TMX files and is not required for the use of the TMX files signed with it.

On which computers is the TwinCAT 3 user certificate TC0008 required?

The TwinCAT 3 user certificate should be located exclusively on the engineering computer on which the TMX files are signed - i.e. **NOT** on each target system.

6.4.14.1.2.2.2 Operating system

NOTICE

Migration to TMX with TwinCAT Loader recommended

Since TwinCAT 3.1 4024.0 versioned C++ projects are available, whose binaries can be loaded directly from TwinCAT. Migration is recommended!

For the implementation of TwinCAT 3 C++ modules on x64 platforms, the driver (*.sys file) must be signed with a certificate if it is to be loaded by the operating system.

The signature, which is automatically executed during the TwinCAT 3 build process, is used by 64-bit Windows operating systems for the authentication of the drivers.

A certificate is required to sign a driver. This Microsoft documentation describes the process and background knowledge for obtaining a test and release certificate that is accepted by 64-bit Windows operating systems.

To use such a certificate in TwinCAT 3, configure the step after compiling your x64 build target as documented in "Creating a test certificate for test mode [▸ 282]".

Test certificates

For testing purposes, self-signed test certificates can be created and used without technical limitations.

The following tutorials describe how to enable this option.
To create drivers with real certificates for production machines, this option must be disabled.

- Creating a test certificate for test mode [▸ 282]
- Delete (test) certificates [▸ 284]

Further references:

MSDN, MakeCert test certificates (Windows drivers),
https://docs.microsoft.com/en-us/windows-hardware/drivers/install/makecert-test-certificate

6.4.14.1.2.2.2.1 Test signing

Overview

Implementing TwinCAT 3 C++ modules for x64 platforms requires signing the driver with a certificate.

This article describes how to create and install a test certificate for testing a C++ driver.

Note the procedure when creating test certificates

Developers may have a wide range of tools for creating certificates. Please follow this description exactly, in order to activate the test certificate mechanism.

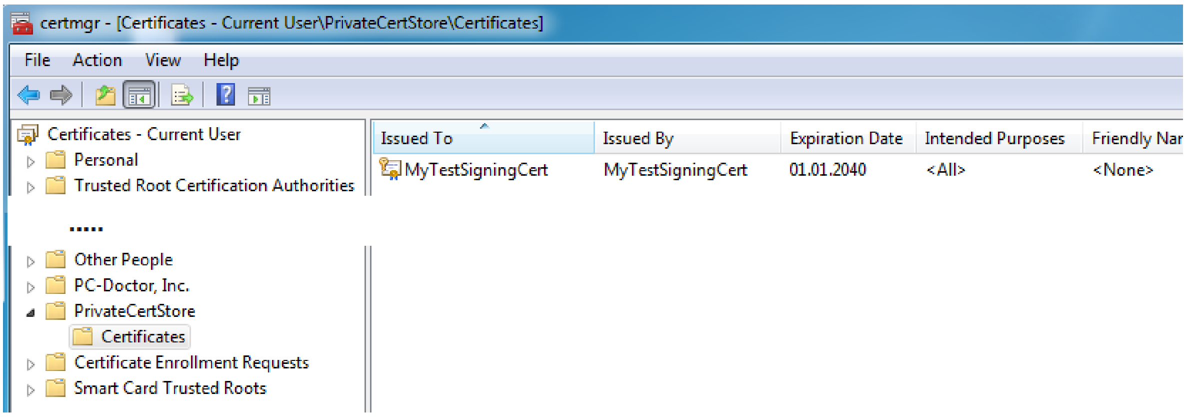
The following commands must be executed from a command line that has been opened in either way:

- **Visual Studio 2010 / 2012 prompt with administrator rights**. (Via: **All Programs** -> **Microsoft Visual Studio 2010/2012** -> **Visual Studio Tools** -> Visual Studio Command Prompt, then right-click **Run as administrator**)
- **Developer Command Prompt of Visual Studio 2017 / 2019 with administrator rights**. (Via: **All Programs** -> **Visual Studio 2017** -> Visual Studio Command Prompt for VS 2017/2019, then right-click on **Run as administrator**)
- Only if the WINDDK has been installed:
  Normal prompt (**Start** ->Command Prompt) with administrator rights, then change to directory *%WINDDK7%\bin\x86\*, which contains the corresponding tools.

1. On XAE:
   in the engineering system enter the following command in the Visual Studio 2010 / 2012 prompt with administrator rights (see note above):
   ```
   makecert -r -pe -ss PrivateCertStore -n CN=MyTestSigningCert
   MyTestSigningCert.cer
   ```
   (If you do not have access rights to the PrivateCertStore, you can use a different location. This must also be used in the PostBuild event, as described here [▸ 286].)

   ⇨ This is followed by creation of a self-signed certificate, which is stored in the file "MyTestSigningCert.cer" and in the Windows Certificate Store.

   ⇨ Check the result with mmc (**Use File->Add/Remove Snap-in->Certificates**):



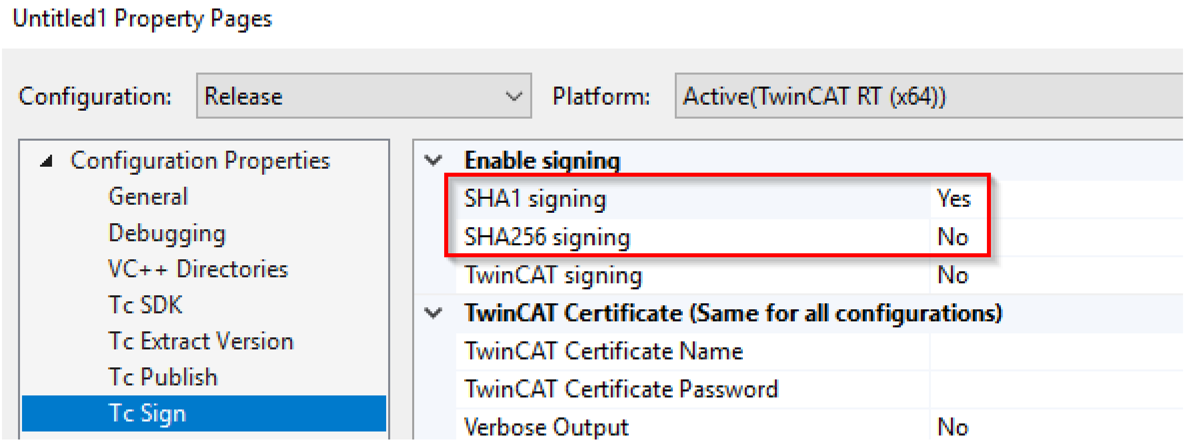
certmgr - [Certificates - Current User\PrivateCertStore\Certificates]

Certificates - Current User
- Personal
- Trusted Root Certification Authorities
- .....
- Other People
- PC-Doctor, Inc.
- PrivateCertStore
  - Certificates
- Certificate Enrollment Requests
- Smart Card Trusted Roots

Issued To	Issued By	Expiration Date	Intended Purposes	Friendly Nar
MyTestSigningCert	MyTestSigningCert	01.01.2040	<All>	<None>

2. On XAE:
   configure the certificate so that it is recognized by TwinCAT XAE on the engineering system.
   Set the environment variable TWINCATTESTCERTIFICATE to "MyTestSigningCert" in the engineering system or edit the post build event of Debug|TwinCAT RT (x64) and Release|TwinCAT RT (x64).
   The name of the variable is NOT the name of the certificate file, but the CN name (in this case MyTestSigningCert).

Notice From TwinCAT 3.1 4024.0, the configuration of the certificate to be used is carried out under Tc Sign in the project properties. To use signing via the operating system, as described here, please pay attention to the project settings:



Untitled1 Property Pages

Configuration: Release   Platform: Active(TwinCAT RT (x64))

- Configuration Properties
  - General
  - Debugging
  - VC++ Directories
  - Tc SDK
  - Tc Extract Version
  - Tc Publish
  - Tc Sign

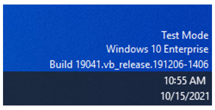
Enable signing	
SHA1 signing	Yes
SHA256 signing	No
TwinCAT signing	No
TwinCAT Certificate (Same for all configurations)	
TwinCAT Certificate Name	
TwinCAT Certificate Password	
Verbose Output	No

On XAR (and XAE, if it is a local test), activate the test mode so that the operating system can accept the self-signed certificates. This can be done on both engineering systems (XAE) and runtime systems (XAR).

For Windows

Use the administrator prompt to execute the following:
`bcdedit /set testsigning yes`
and reboot the target system.
You may have to switch off "SecureBoot" for this, which can be done in the bios.

If test signing mode is enabled, this is displayed at the bottom right of the desktop. The system now accepts all signed drivers for execution.

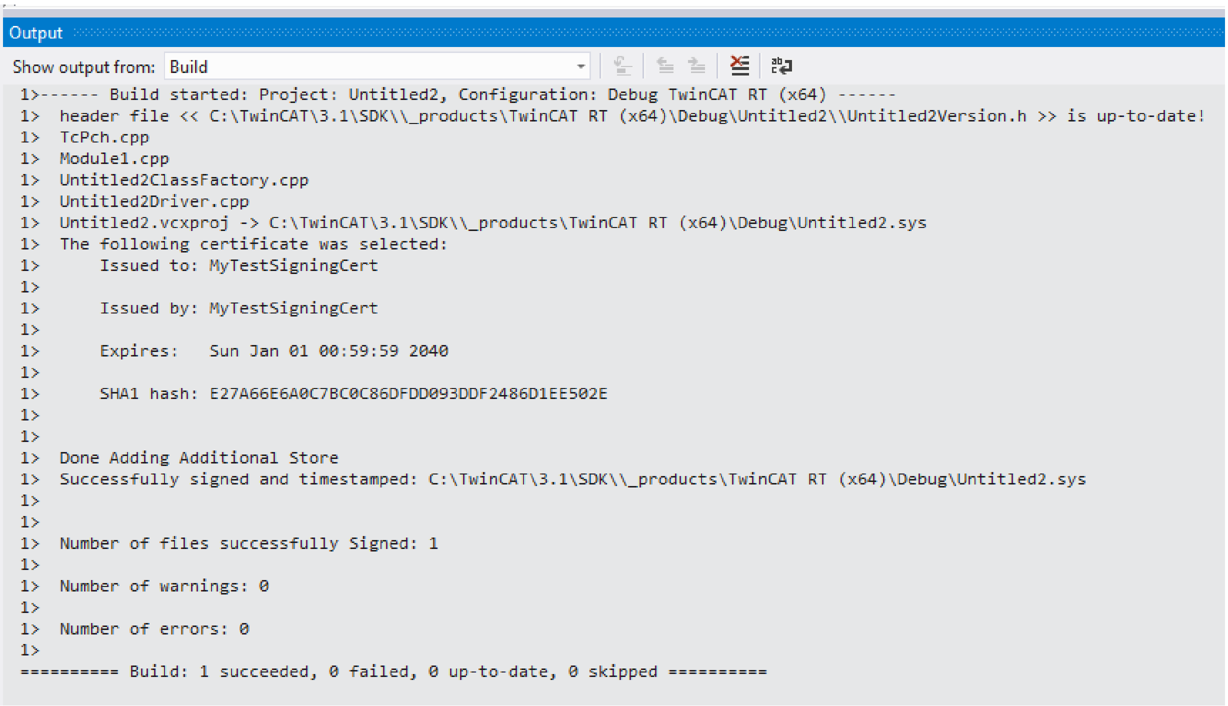


Test Mode
Windows 10 Enterprise
Build 19041.vb_release.191206-1406
10:55 AM
10/15/2021

After the respective procedure, the system accepts all signed drivers for execution.

1. Test whether a configuration with a TwinCAT module implemented in a TwinCAT C++ driver can be enabled and started on the target system.

⇨ Compilation of the x64 driver generates the following output:



```
1>------ Build started: Project: Untitled2, Configuration: Debug TwinCAT RT (x64) ------
1>  header file << C:\TwinCAT\3.1\SDK\\_products\TwinCAT RT (x64)\Debug\Untitled2\\Untitled2Version.h >> is up-to-date!
1>  TcPch.cpp
1>  Module1.cpp
1>  Untitled2ClassFactory.cpp
1>  Untitled2Driver.cpp
1>  Untitled2.vcxproj -> C:\TwinCAT\3.1\SDK\\_products\TwinCAT RT (x64)\Debug\Untitled2.sys
1>  The following certificate was selected:
1>      Issued to: MyTestSigningCert
1>
1>      Issued by: MyTestSigningCert
1>
1>      Expires:   Sun Jan 01 00:59:59 2040
1>
1>      SHA1 hash: E27A66E6A0C7BC0C86DFDD093DDF2486D1EE502E
1>
1>
1>  Done Adding Additional Store
1>  Successfully signed and timestamped: C:\TwinCAT\3.1\SDK\\_products\TwinCAT RT (x64)\Debug\Untitled2.sys
1>
1>
1>  Number of files successfully Signed: 1
1>
1>  Number of warnings: 0
1>
1>  Number of errors: 0
1>
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

References:
MSDN, MakeCert test certificates (Windows drivers),
https://docs.microsoft.com/en-us/windows-hardware/drivers/install/makecert-test-certificate

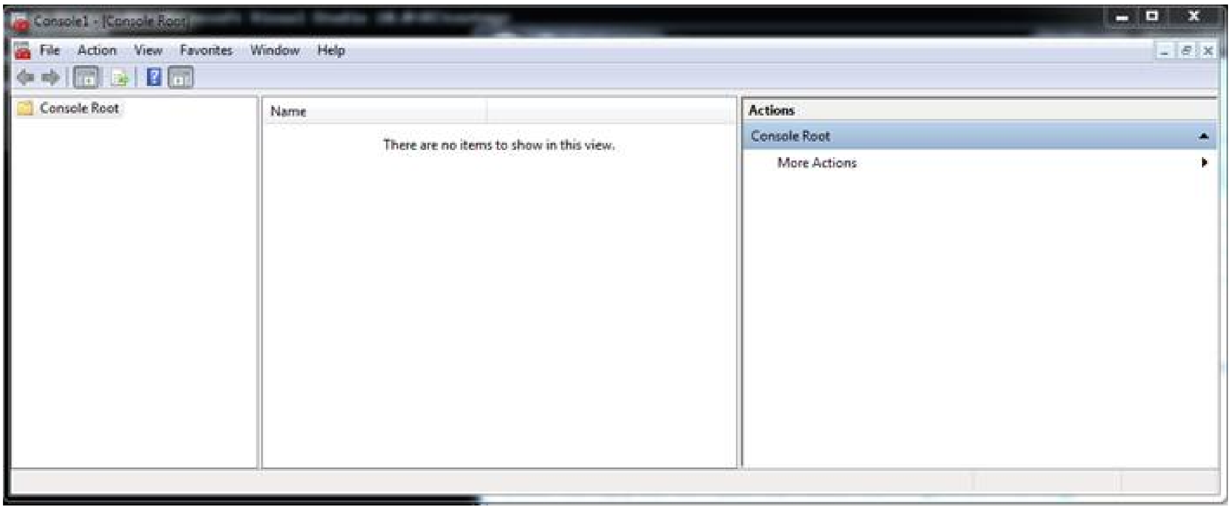
6.4.14.1.2.2.2.2 Delete test certificate

This article is about how to delete a test certificate.

Overview

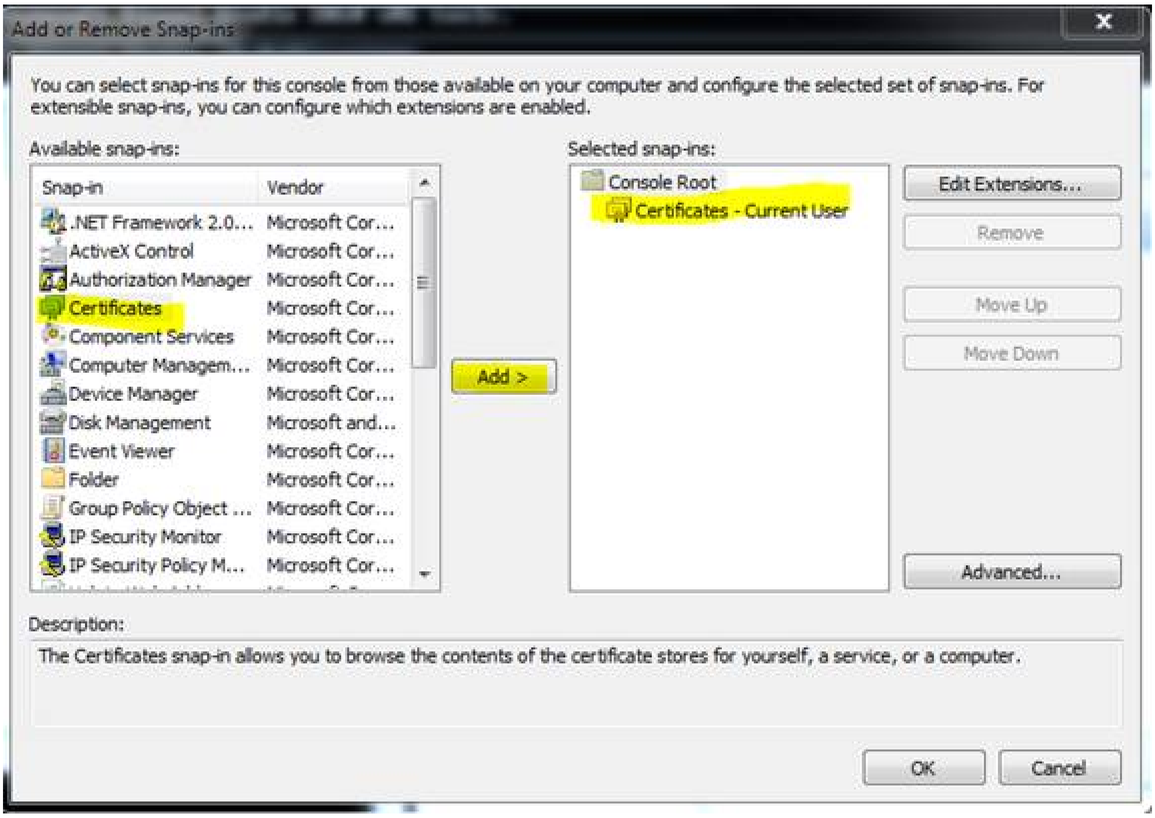
A certificate can be deleted with the Microsoft Management Console:

1. Start the management console MMC.exe via the Start menu or the user interface.



There are no items to show in this view.

2. Click in the menu on **File** -> **Add/Remove Snap-in..** and select the certificate snap-in for the current user; conclude with **OK**.



You can select snap-ins for this console from those available on your computer and configure the selected set of snap-ins. For extensible snap-ins, you can configure which extensions are enabled.

Available snap-ins:

Snap-in	Vendor
.NET Framework 2.0...	Microsoft Cor...
ActiveX Control	Microsoft Cor...
Authorization Manager	Microsoft Cor...
Certificates	Microsoft Cor...
Component Services	Microsoft Cor...
Computer Managem...	Microsoft Cor...
Device Manager	Microsoft Cor...
Disk Management	Microsoft and...
Event Viewer	Microsoft Cor...
Folder	Microsoft Cor...
Group Policy Object ...	Microsoft Cor...
IP Security Monitor	Microsoft Cor...
IP Security Policy M...	Microsoft Cor...

Selected snap-ins:

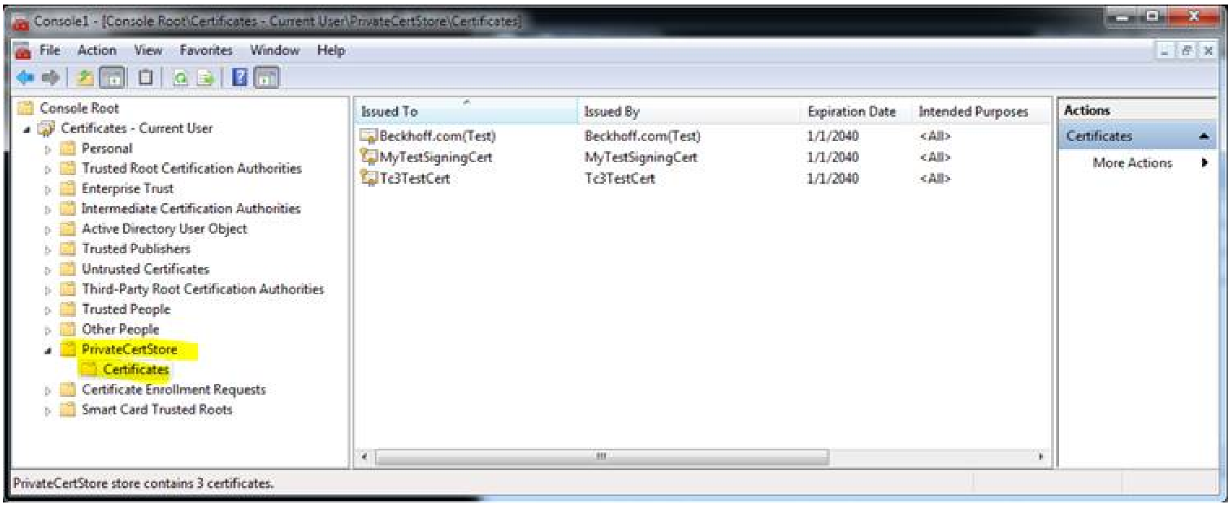
- Console Root
  - Certificates - Current User

Description:

The Certificates snap-in allows you to browse the contents of the certificate stores for yourself, a service, or a computer.

⇨ The certificates are listed in the node under **PrivateCertStore/Certificates**.

3. Select the certificate to be deleted.



Console1 - [Console Root\Certificates - Current User\PrivateCertStore\Certificates]

- Console Root
  - Certificates - Current User
    - Personal
    - Trusted Root Certification Authorities
    - Enterprise Trust
    - Intermediate Certification Authorities
    - Active Directory User Object
    - Trusted Publishers
    - Untrusted Certificates
    - Third-Party Root Certification Authorities
    - Trusted People
    - Other People
    - PrivateCertStore
      - Certificates
    - Certificate Enrollment Requests
    - Smart Card Trusted Roots

Issued To	Issued By	Expiration Date	Intended Purposes
Beckhoff.com(Test)	Beckhoff.com(Test)	1/1/2040	<All>
MyTestSigningCert	MyTestSigningCert	1/1/2040	<All>
Tc3TestCert	Tc3TestCert	1/1/2040	<All>

Actions
Certificates
More Actions

PrivateCertStore store contains 3 certificates.

6.4.14.1.2.2.2.3 Windows driver without test mode

For Windows operating systems the driver has to be signed via "Attestation Signing". This requires an EV certificate.
Microsoft provides relevant instructions: https://docs.microsoft.com/en-us/windows-hardware/drivers/dashboard/attestation-signing-a-kernel-driver-for-public-release
The drivers created in this way are also suitable for devices that have secure boot enabled.

Microsoft announced that the previous procedure using CrossSigning certificates (signing tool with parameter /ac) will be discontinued from July 2021. After this date it can no longer be used (depending on the expiry date of the individual CrossSigning certificate), as documented here.
Versioned C++ projects, which are loaded via the TwinCAT Loader, have been available for TwinCAT C++ for some time; they are not drivers for the purposes of the operating system. Beckhoff therefore recommends using versioned C++ projects.

A guide is available in the How-to section for the migration of TwinCAT C++ drivers to versioned C++ projects.

6.4.14.1.2.3 TcSignTool - Storage of the certificate password outside the project

The TcSignTool can be used to store a password for a TwinCAT user certificate in the registry. Thus, the password is not needed in the projects, where the passwords would end up unintentionally in version control systems.

The TcSignTool is a command line program located in the path *C:\TwinCAT\3.x\sdk\Bin\.*

The storage of the password is carried out with the following parameters:

```
tcsigntool grant /f "C:\TwinCAT\3.1\CustomConfig\Certificates\MyCertificate.tccert" /p MyPassword
```

The password is deleted with the following parameters:

```
tcsigntool grant /f "C:\TwinCAT\3.1\CustomConfig\Certificates\MyCertificate.tccert" /r
```

The unencrypted password is stored under `HKEY_CURRENT_USER\SOFTWARE\Beckhoff\TcSignTool\`

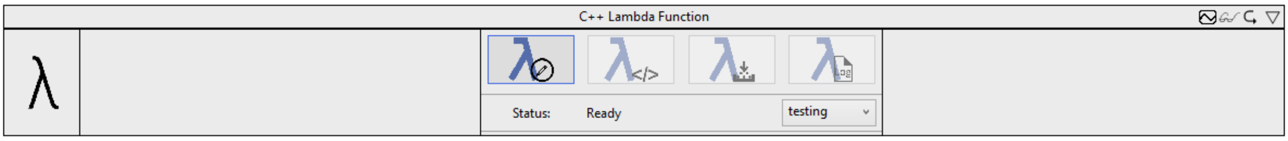
6.4.14.1.2.4 Environment variables

Finally, two environment variables must be created with the certificate name. These are read during the generation of the lambda function and sign the lambda function with the corresponding certificates.

Variable name	Variable value
TWINCATTESTCERTIFICATE	<Operating System Certificate Name> (e.g. MyTestSigningCert)
TWINCATCERTIFICATENAME	<TwinCAT certificate name> (e.g. TwinCATTestCert)

The file extensions (.cert and .tccert) must not be included.

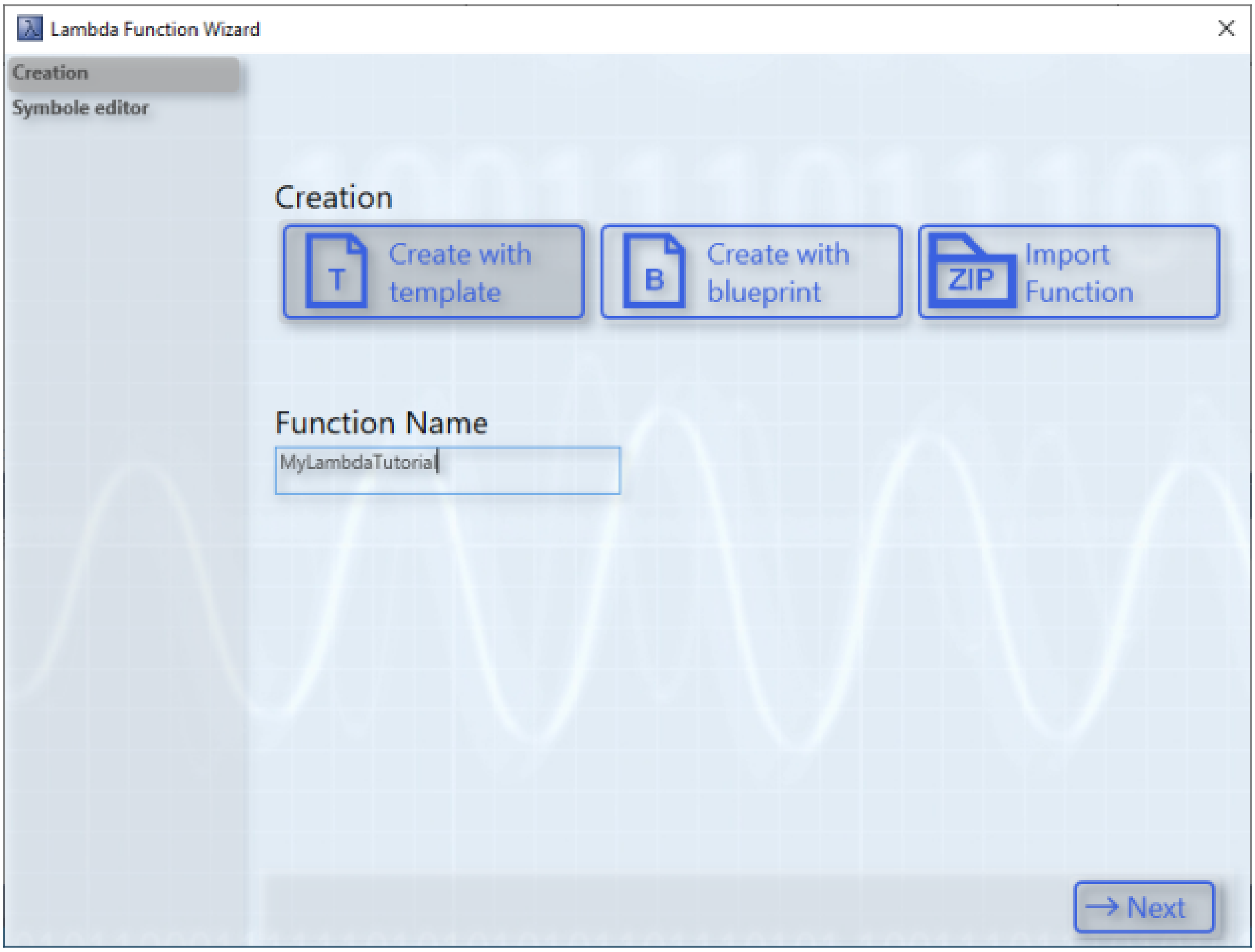
6.4.14.1.3 Lambda Template



The lambda template guides through the development of a lambda function. Within the parameter area there is a control. This control contains four buttons for configuring, programming and publishing the lambda function. The fourth button is for viewing the log files of the development process. A status bar provides additional information during the process. A drop-down menu at the bottom right offers the possibility to choose between a release and a testing version of the lambda function.

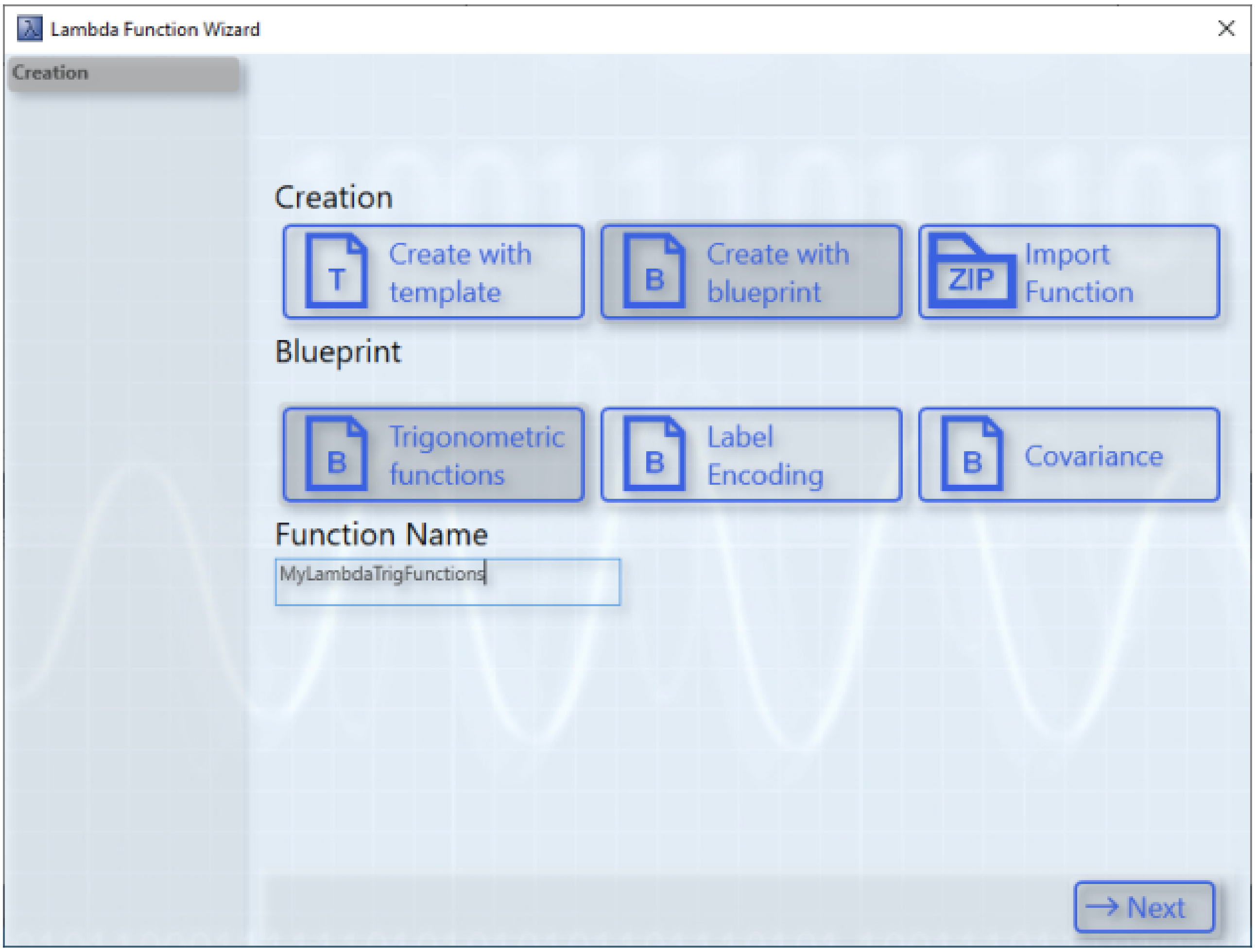
Configuring the lambda function

The first button from the left opens a wizard. With the help of this wizard you can configure the lambda function.



Function Name: MyLambdaTutorial

On the first page you can choose between three modes. The first mode is called Template. With this mode new lambda functions can be developed. All files and dependencies are prepared in a template and can be implemented with user-specific logic.



Function Name: MyLambdaTrigFunctions

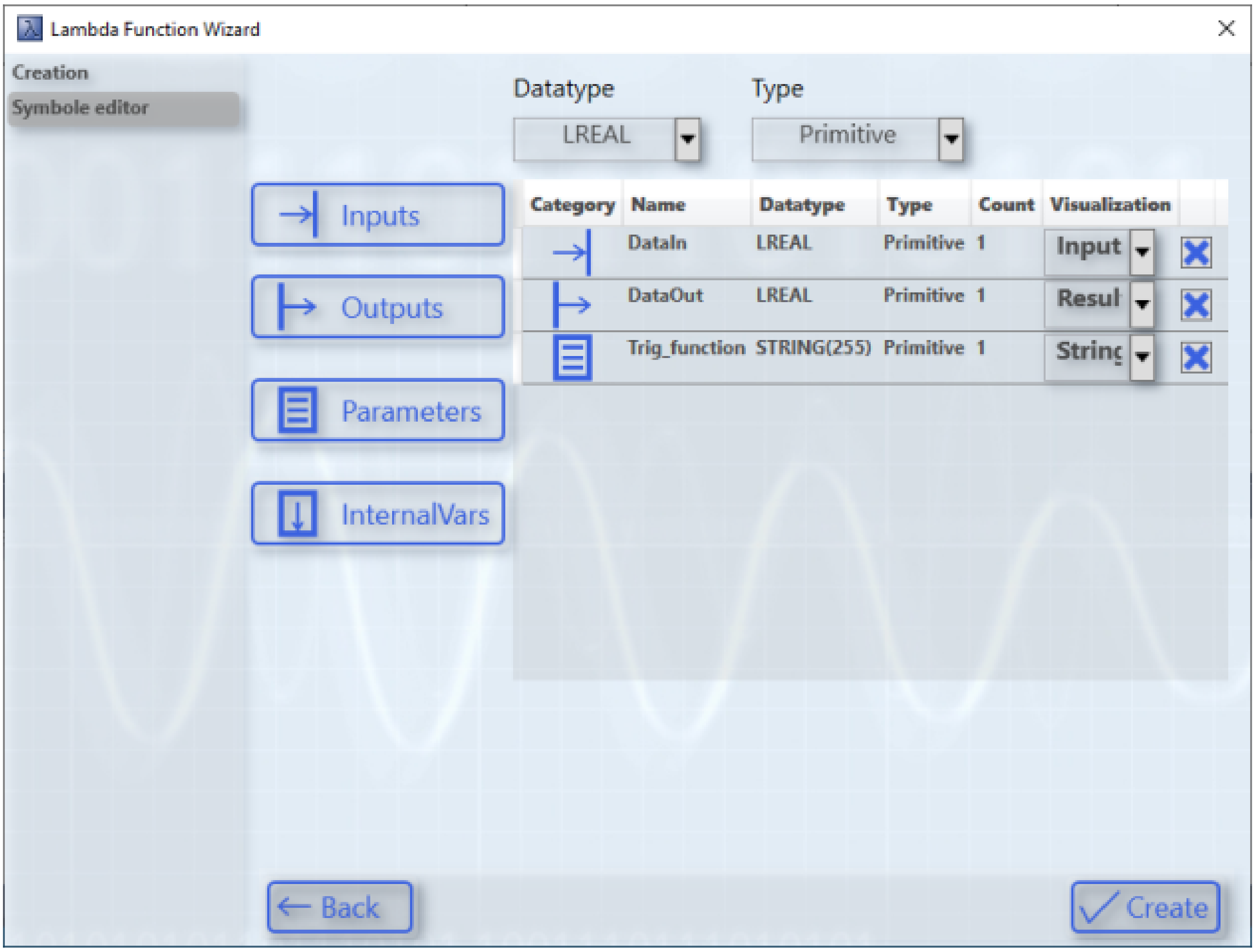
The second mode is called Blueprint. There are three blueprints already implemented in each case. There is a blueprint for calculating trigonometric functions, for label encoding and for calculating covariance. These blueprints serve as an example configuration and implementation of a lambda function. More detailed information is available at the following link (see: Lambda Blueprints [▸ 295]).

Via the third mode you import exported lambda functions. With a click on the button **Import Function**, an Open File dialog opens. Then you can select the exported lambda function, which should be available as a ZIP archive. As soon as the lambda function is imported, the name can be recognized and the configured symbols are shown on the next page.

The name of the lambda function can be selected individually. However, the version number in the name of the lambda function should not be adjusted or changed. Otherwise, the proper use of the lambda functions may be impaired. Also the function name should contain at least four characters.

On the second page of the wizard you configure the symbols and variables of the lambda function.

In the Blueprint mode, some symbols are already preconfigured.

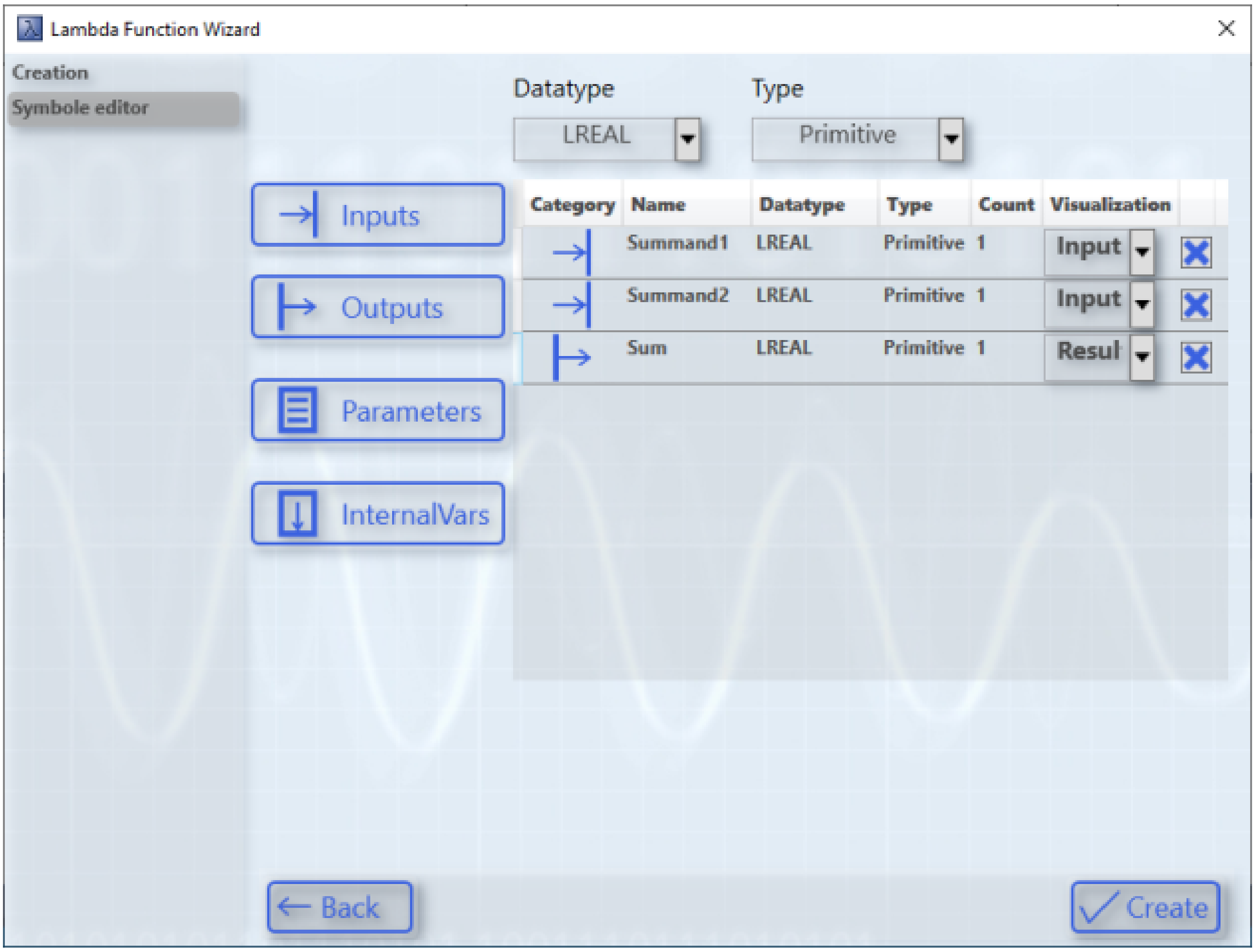


Datatype: LREAL

Type: Primitive

Category	Name	Datatype	Type	Count	Visualization	
	DataIn	LREAL	Primitive	1	Input	☒
	DataOut	LREAL	Primitive	1	Resul	☒
	Trig_function	STRING(255)	Primitive	1	String	☒

In the template mode the list of symbols is empty. As shown in the following figure, you can configure user-specific symbols.



Datatype: LREAL

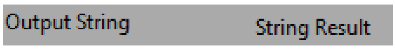
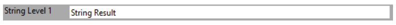
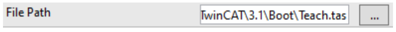
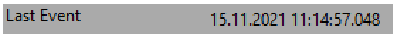
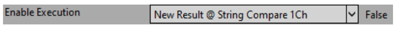
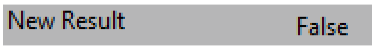
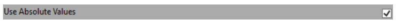
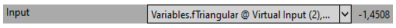
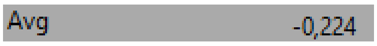
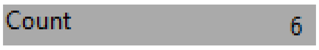
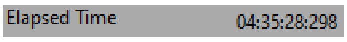
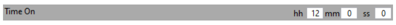
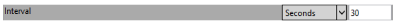
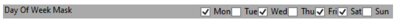
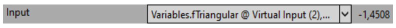
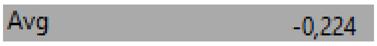
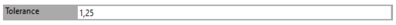
Type: Primitive

Category	Name	Datatype	Type	Count	Visualization	
⇥	Summand1	LREAL	Primitive	1	Input	☒
⇥	Summand2	LREAL	Primitive	1	Input	☒
↦	Sum	LREAL	Primitive	1	Resul	☒

Use the two drop-down boxes at the very top of the page to select the data type and the data structure type, respectively.

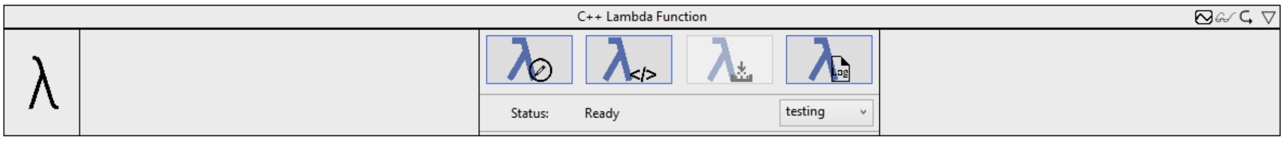
Available data types				
LREAL	REAL	BOOL	WORD	INT
LINT	ULINT	LTIME	STRING(255)	

For the data structure types you can choose between primitive and array.
Use the buttons on the right to add the variables with the selected data type and data structure type.
Internal variables are member variables. They are not displayed in the algorithm interface and cannot be linked in Analytics Workbench. Their purpose is to store variable values over cycles.
If the data structure type Array is selected for a variable, the size of the array can be specified in the Count column of the variable. To create multi-dimensional arrays, the dimensions must be separated with the letter "x" (example: 2x4).
You can configure the visualization of the symbol in the Workbench using the drop-down menu. Depending on the data type, different visualizations are possible. Undefined visualizations are not possible. No visualization can be selected for the internal variables because they are not displayed in the Workbench. The following visualizations are possible for the respective data types for inputs, outputs and parameters:

Data type	Visualization	Sample
STRING(255)	String (Input)	
	String (Output)	 Output String String Result
	String (Parameter)	 String Level 1 String Result
	FileOpen/FileSave (Parameter)	 File Path TwinCAT\3.1\Boot\Teach.tas ...
ULINT	Timestamp (Output)	 Last Event 15.11.2021 11:14:57.048
BOOL	Boolean (Input)	 Enable Execution New Result @ String Compare 1Ch False
	Boolean (Output)	 New Result False
	Boolean (Parameter)	 Use Absolute Values ☑
REAL	Input (Input)	 Input Variables.fTriangular @ Virtual Input (2),... -1,4508
	Result (Output)	 Avg -0,224
INT	Count (Output)	 Count 6
LINT	Timespan (Output)	 Elapsed Time 04:35:28:298
	Clock (Parameter)	 Time On hh 12 mm 0 ss 0
	Interval (Parameter)	 Interval Seconds 30
WORD	DayOfWeekMask (Parameter)	 Day Of Week Mask ☑ Mon ☐ Tue ☑ Wed ☐ Thu ☑ Fri ☑ Sat ☐ Sun
LREAL	Input (Input)	 Input Variables.fTriangular @ Virtual Input (2),... -1,4508
	Result (Output)	 Avg -0,224
	Level (Parameter)	 Tolerance 1,25

After clicking the Create button, the C++ project is generated in the background and the wizard is closed.

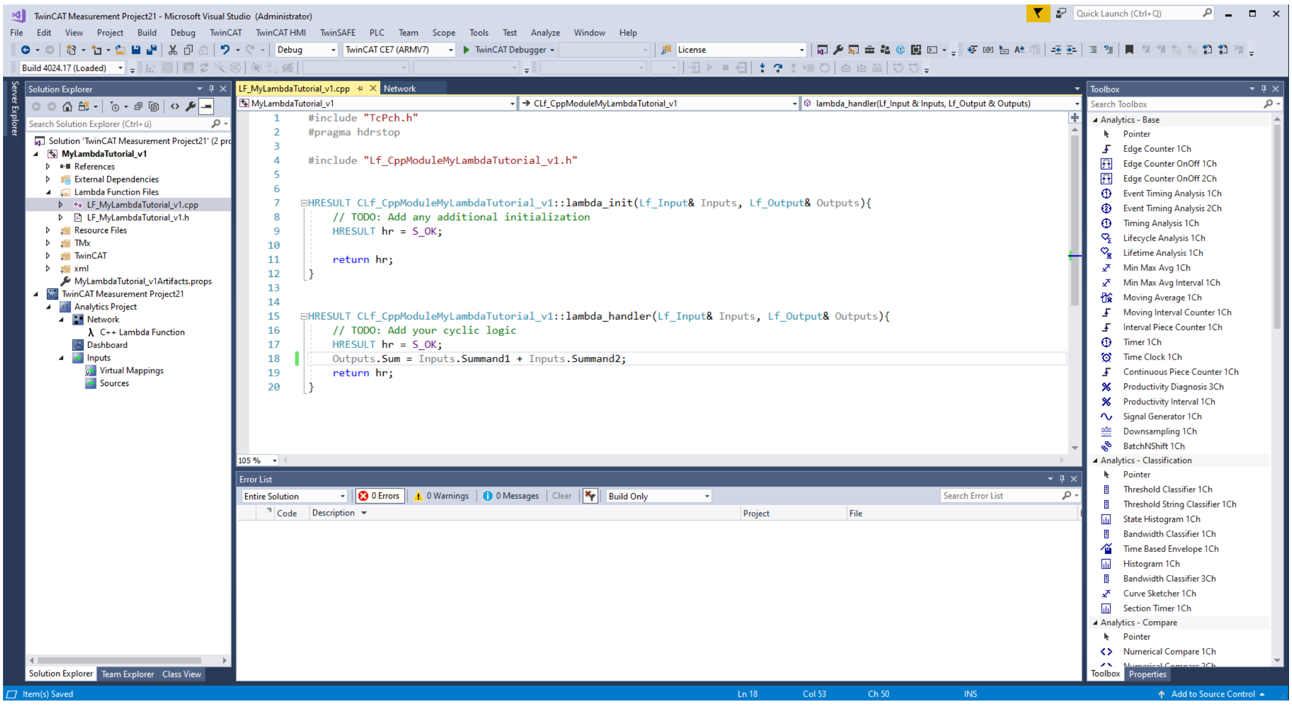
Programming the lambda function


C++ Lambda Function

Status: Ready

testing

After you click the second button, the C++ project will be added to the Solution Explorer. The project contains all files, dependencies and configuration to develop an executable TcCOM module for the Analytics Workbench as well as for use in a real-time context.



```
#include "TcPch.h"
#pragma hdrstop

#include "Lf_CppModuleMyLambdaTutorial_v1.h"


HRESULT CLf_CppModuleMyLambdaTutorial_v1::lambda_init(Lf_Input& Inputs, Lf_Output& Outputs){
    // TODO: Add any additional initialization
    HRESULT hr = S_OK;

    return hr;
}


HRESULT CLf_CppModuleMyLambdaTutorial_v1::lambda_handler(Lf_Input& Inputs, Lf_Output& Outputs){
    // TODO: Add your cyclic logic
    HRESULT hr = S_OK;
    Outputs.Sum = Inputs.Summand1 + Inputs.Summand2;
    return hr;
}
```

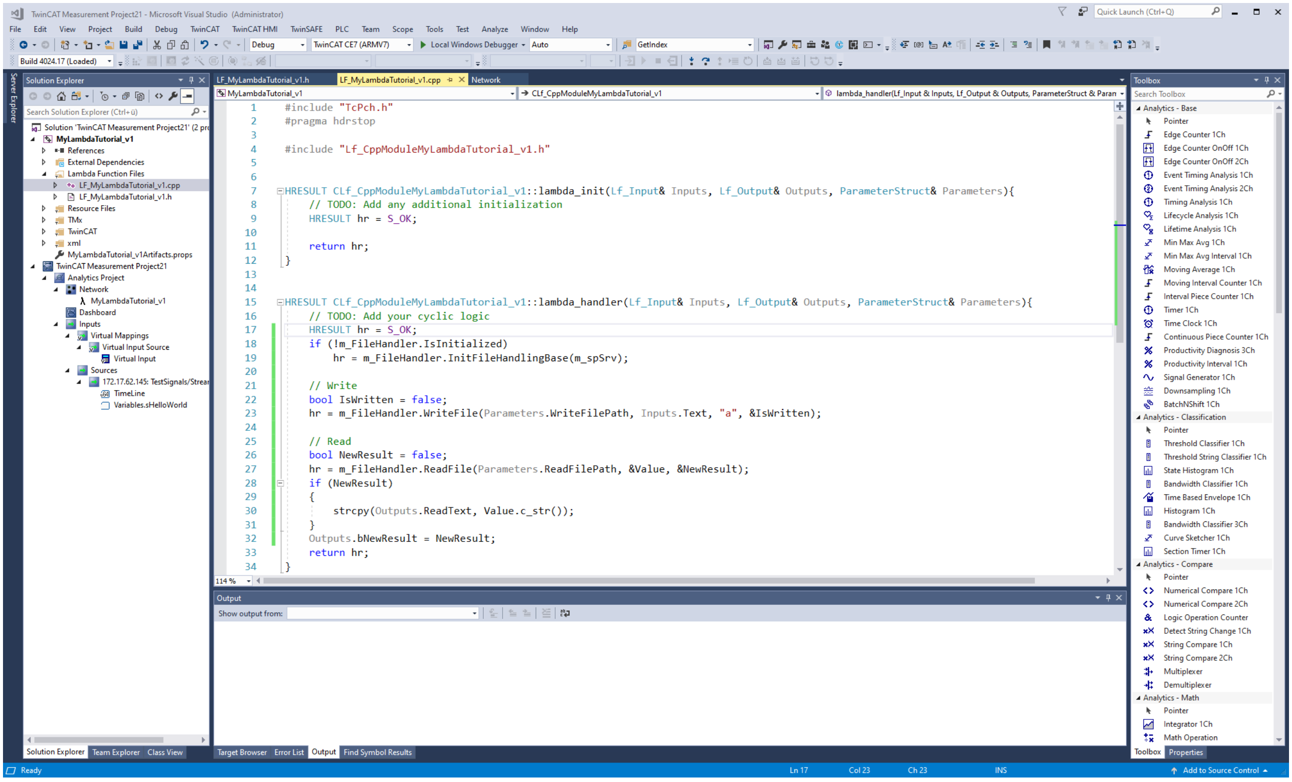
The automatically opened file contains several encapsulated functions. The so-called lambda_init method is for the initialization of variables. The lambda_handler method is executed cyclically. The lambda_uninit method is for deinitializing variables. In both methods the inputs, outputs, parameters and internal variables are available as structures in the method parameters. The following inputs and outputs are available by default in the lambda functions:

Category	Symbol name	Data type	Description	Visibility
Input	tTimestamp	ULINT	The variable contains the TwinCAT timestamp.	Hidden
	bReset	BOOL	The variable resets the configured symbols.	Optional
	bEnableExecution	BOOL	The variable activates the lambda_handler method.	Optional
Output	bNewResult	BOOL	The variable indicates whether the algorithm returns new results.	Optional

Variables created within the methods lose their content after one cycle. Internal variables offer the possibilities to store variable values over cycles. An object named "m_FileHandler" is available in the two Lambda methods. You must initialize this object once, if you have not already done so. Then you can use the two functions "ReadFile" and "WriteFile" of the object. The functions have the following properties:

Function name	Parameter				Return value
InitFileHandlingBase	ITComObjectServer* ipSrv				Hresult
ReadFile	PCCH filename	Std::string* value	bool* bNewResult		Hresult
WriteFile	PCCH filename	Std::string value	Std::string mode	bool* bIsWritten	Hresult

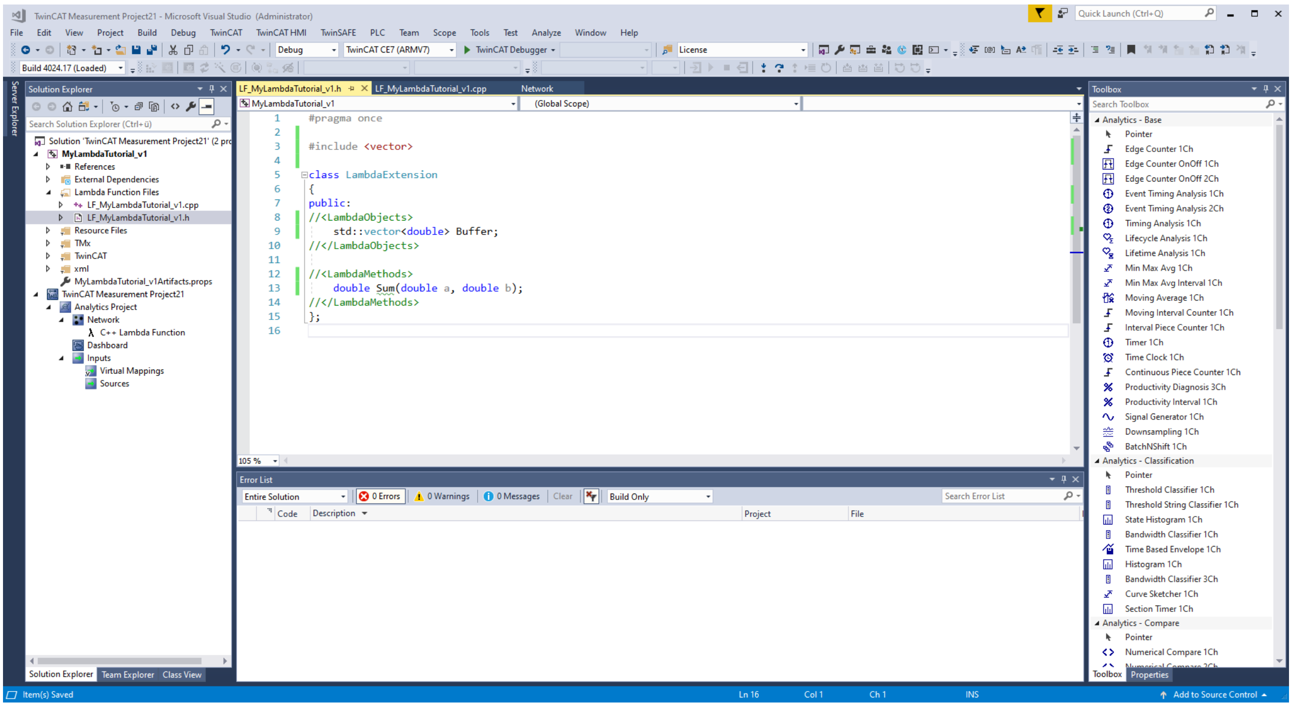
In TwinCAT Analytics Workbench, file access is synchronous, whereas in Analytics Runtime it is asynchronous. This feature is automatically distinguished in the methods. In case of an asynchronous read access, the method does not return the read value in the same cycle. Only as soon as the parameter bNewResult returns the value true, the parameter value contains the read value. With asynchronous write access, the data is stored internally and written at the next opportunity. Once the data is written, the bIsWritten parameter returns true. You can use the two parameters bNewResult and bIsWritten analogously for a synchronous file access. The following figure shows an example usage of the file access in the lambda functions:



```
#include "TcPch.h"
#pragma hdrstop

#include "Lf_CppModuleMyLambdaTutorial_v1.h"


HRESULT CLf_CppModuleMyLambdaTutorial_v1::lambda_init(Lf_Input& Inputs, Lf_Output& Outputs, ParameterStruct& Parameters){
    // TODO: Add any additional initialization
    HRESULT hr = S_OK;

    return hr;
}


HRESULT CLf_CppModuleMyLambdaTutorial_v1::lambda_handler(Lf_Input& Inputs, Lf_Output& Outputs, ParameterStruct& Parameters){
    // TODO: Add your cyclic logic
    HRESULT hr = S_OK;
    if (!m_FileHandler.IsInitialized)
        hr = m_FileHandler.InitFileHandlingBase(m_spSrv);

    // Write
    bool IsWritten = false;
    hr = m_FileHandler.WriteFile(Parameters.WriteFilePath, Inputs.Text, "a", &IsWritten);

    // Read
    bool NewResult = false;
    hr = m_FileHandler.ReadFile(Parameters.ReadFilePath, &Value, &NewResult);
    if (NewResult)
    {
        strcpy(Outputs.ReadText, Value.c_str());
    }
    Outputs.bNewResult = NewResult;
    return hr;
}
```

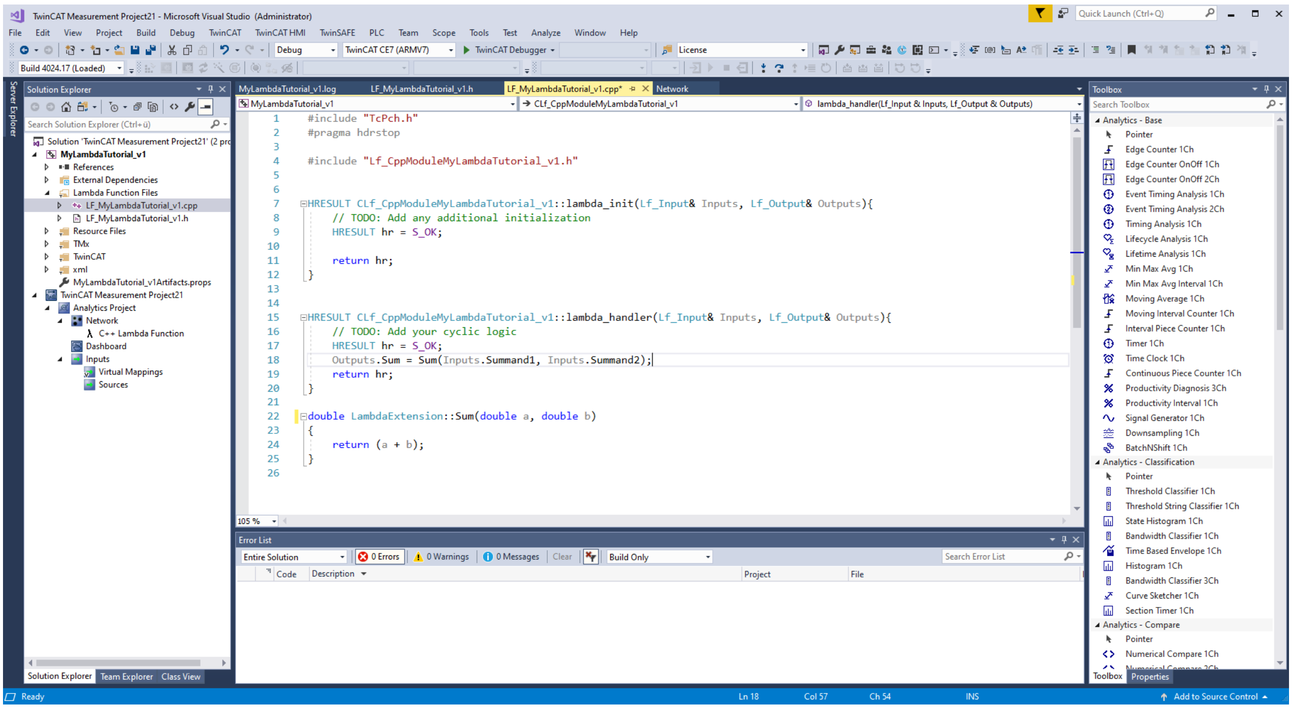
Create the project as usual in Visual Studio® to check the syntax.

In the header file in the "Lambda Function Files" folder you can declare user-specific objects and methods.



```
#pragma once

#include <vector>

class LambdaExtension
{
public:
//<LambdaObjects>
    std::vector<double> Buffer;
//</LambdaObjects>

//<LambdaMethods>
    double Sum(double a, double b);
//</LambdaMethods>
};

```

Above the class "LambdaExtension" libraries can be included. Within the class, objects and methods can be declared and used in the lambda functions. The newly declared methods can be implemented below the two lambda functions.



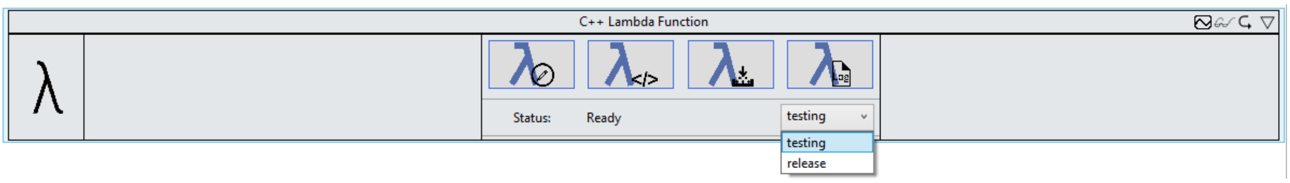
```
#include "TcPch.h"
#pragma hdrstop

#include "Lf_CppModuleMyLambdaTutorial_v1.h"


HRESULT CLf_CppModuleMyLambdaTutorial_v1::lambda_init(Lf_Input& Inputs, Lf_Output& Outputs){
    // TODO: Add any additional initialization
    HRESULT hr = S_OK;

    return hr;
}


HRESULT CLf_CppModuleMyLambdaTutorial_v1::lambda_handler(Lf_Input& Inputs, Lf_Output& Outputs){
    // TODO: Add your cyclic logic
    HRESULT hr = S_OK;
    Outputs.Sum = Sum(Inputs.Summand1, Inputs.Summand2);
    return hr;
}

double LambdaExtension::Sum(double a, double b)
{
    return (a + b);
}
```

Error List: Entire Solution, 0 Errors, 0 Warnings, 0 Messages, Build Only

After the logic is integrated, you need to save the project. Then switch back to the lambda template in the Analytics project.
If sufficient know-how in dealing with TcCOM modules is available, the C++ project can be extended individually. If the TcCOM modules are used improperly, errors may occur during subsequent steps of the lambda functions. More information on the development of TcCOM modules can be found at the following link TwinCAT C++ development.

Publishing the lambda function

Before publishing, you can configure the type of lambda function using the drop-down menu at the bottom right.

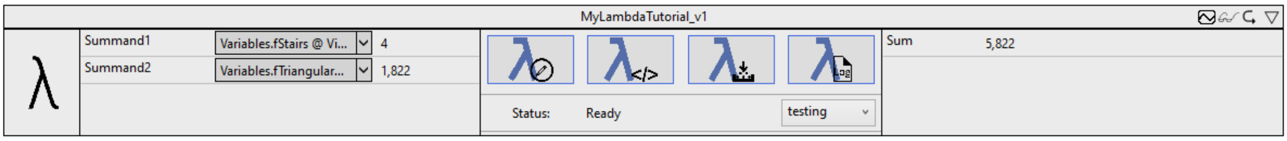


C++ Lambda Function

Status: Ready

testing
- testing
- release

A lambda function of type "testing" contains the control with the four buttons for developing lambda functions. This type is useful during the development and testing phase.



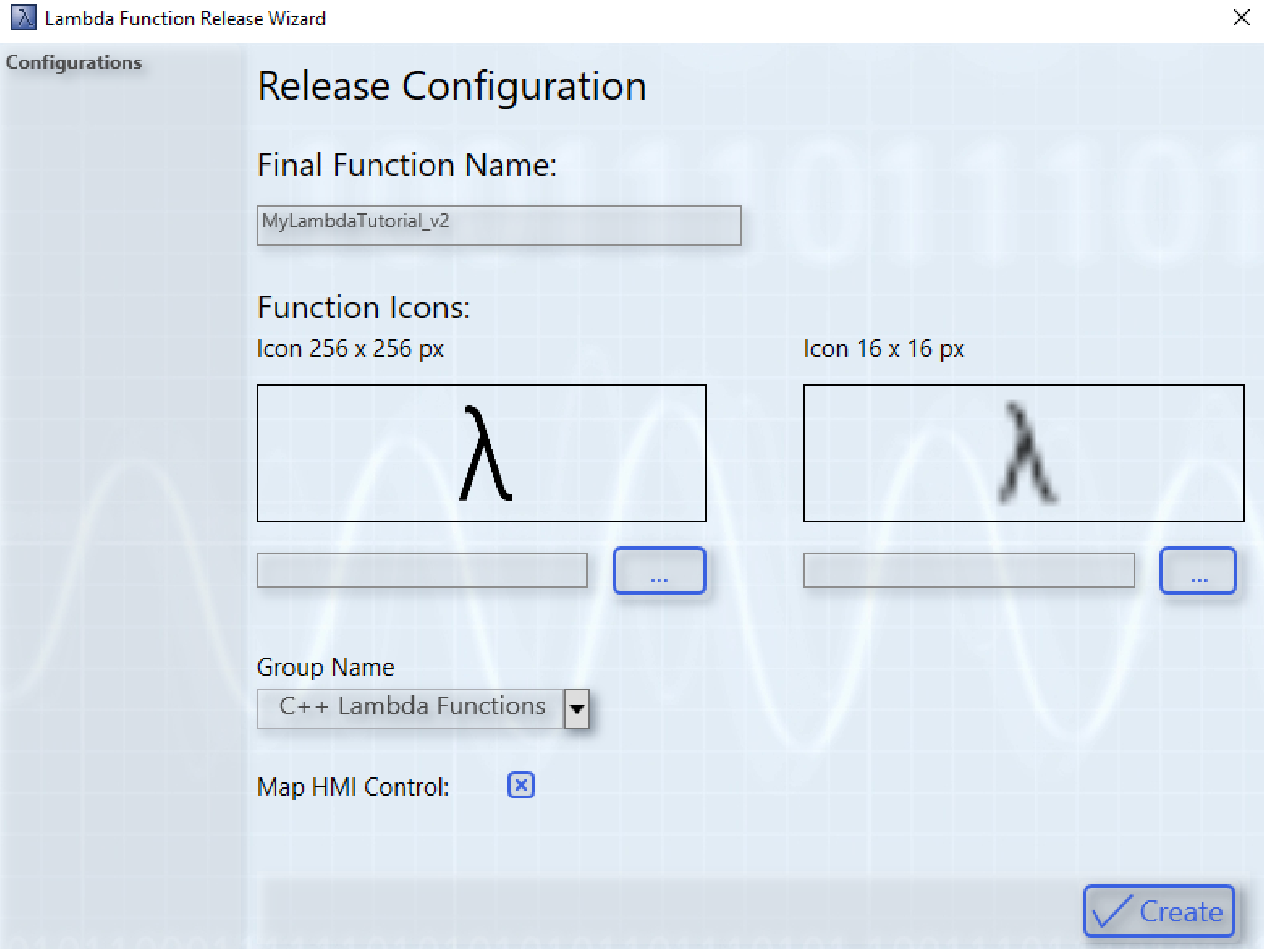
MyLambdaTutorial_v1

Summand1	Variables.fStairs @ Vi...	4
Summand2	Variables.fTriangular...	1,822

Status: Ready — testing

Sum 5,822

Once the lambda function is working, select the "release" type. After clicking the third button, a Release Wizard opens.



Lambda Function Release Wizard

Configurations

Release Configuration

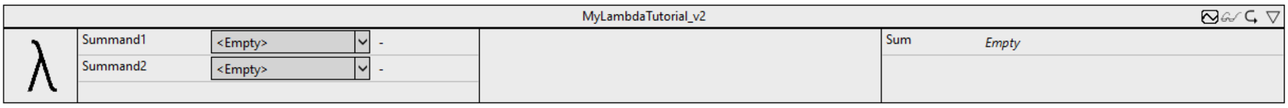
Final Function Name: MyLambdaTutorial_v2

Function Icons:

Group Name: C++ Lambda Functions

Map HMI Control: ☒

The wizard offers the possibility to assign a final name and select individual icons. One icon with 256 x 256 pixels and one icon with 16 x 16 pixels must be selected. Alternatively, you can continue to use the Lambda icons. Use the drop-down menu to assign the lambda function to a specific group in the toolbox. By default, the "C++ Lambda Functions" group is selected. For example, a final lambda function can be placed in the "Analytics - Base" section of the toolbox. If the "Map HMI Control" checkbox is activated, a wizard for linking HMI controls with the lambda function opens after publishing. In Analytics HMI Control Manager, the points for module selection are preselected directly with the corresponding lambda function (see also Use customized and own controls [▸ 432]). Use the Create button to generate the lambda function.



MyLambdaTutorial_v2

Summand1	<Empty>	-	Sum	Empty
Summand2	<Empty>	-		

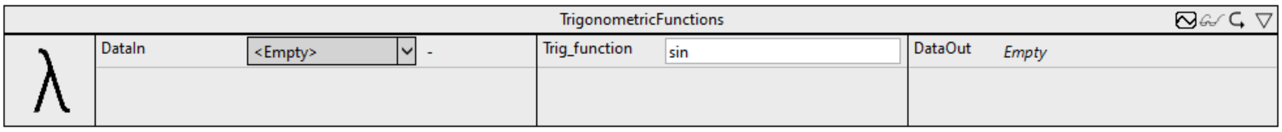
After successful publishing, the lambda template is automatically replaced by the newly generated lambda function.

Viewing the log files

If an error occurs while publishing the lambda function, it is shown in the status display and the log file can be viewed via the fourth button.

6.4.14.1.4 Lambda Blueprints

Trigonometric functions



TrigonometricFunctions

DataIn: <Empty> -

Trig_function: sin

DataOut: Empty

The blueprint **Triginometric function** performs the calculation of a trigonometric function on an input channel and returns the result to the output channel. The input channel expects the angle in radians. The respective trigonometric function can be specified in the parameter area.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

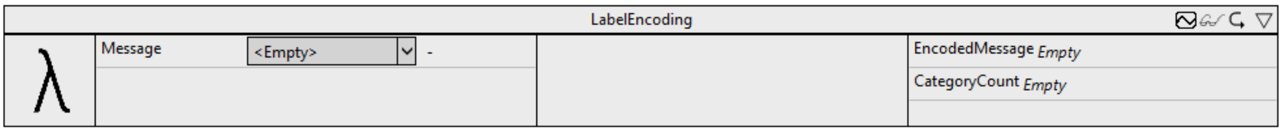
Configuration options

- **Trig_function**: the short form of the trigomic function to be calculated, e.g. sin for sine. The choices include: sin, cos, tan, asin, acos and atan. The input value for asin and acos must lie in the interval [-1,1].

Output values

- **DataOut**: outputs the result of the calculation of the trigonometric function.

Label Encoding



LabelEncoding

Message	<Empty> -
EncodedMessage	Empty
CategoryCount	Empty

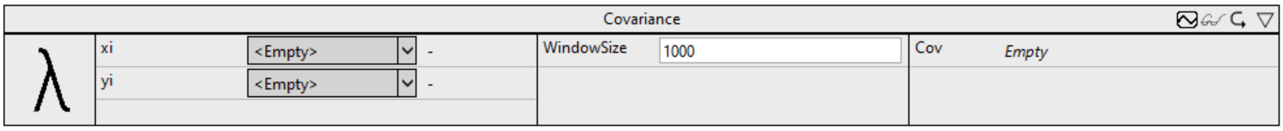
The blueprint **Label Encoding** converts text on an input channel to numeric values. For this purpose, the texts are classified into different categories at the input channel. Equal input texts return the same numeric value.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Output values

- **EncodedMessage**: this output channel returns the index of the corresponding category.
- **CategoryCount**: this output channel outputs the number of categories.

Covariance



Covariance

xi	<Empty> -
yi	<Empty> -
WindowSize	1000
Cov	Empty

The blueprint **Covariance** calculates the covariance between two input channels. WindowSize specifies how many values are included in the calculation of the covariance.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

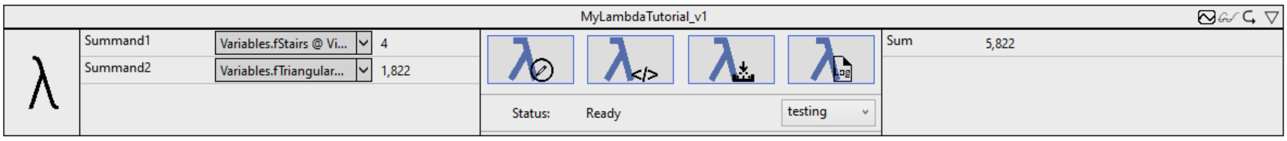
- **WindowSize**: this parameter specifies the number of cycles over which the covariance is calculated. A sliding window is implemented in the blueprint. The current values are included in the analysis and outputs are updated with each cycle. The size of the router memory should be taken into account when setting this parameter.

Output values

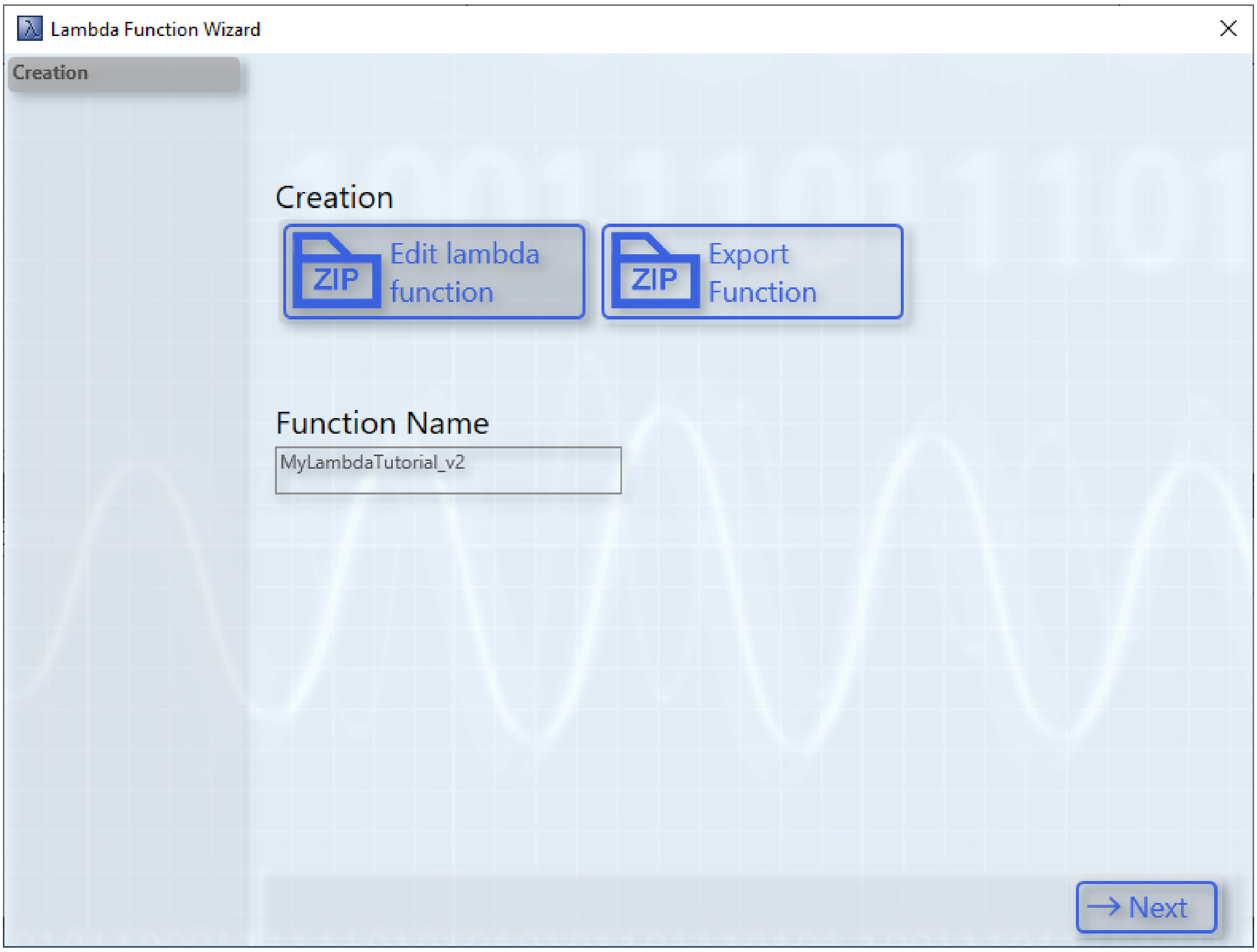
- **Cov**: this output channel indicates the covariance of the two input channels over the configured window size.

6.4.14.1.5 Lambda Function

In the development and testing phase, lambda functions should be published as testing versions. These contain the configured symbols, as well as the Lambda Control with the four buttons.

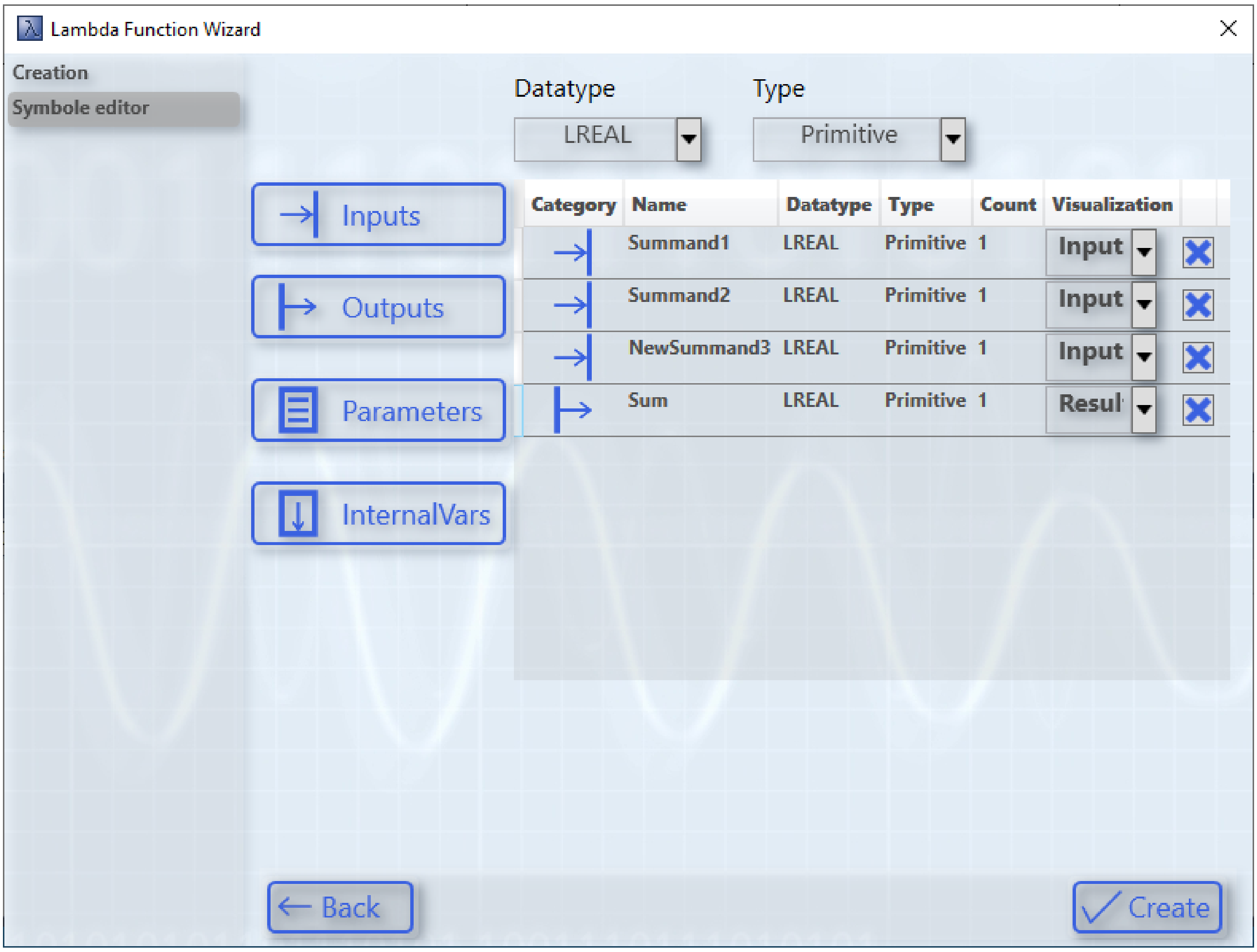


This lambda function can be tested like any other algorithm in Analytics Workbench. If changes are necessary, the Lambda Control guides you through the further development. After clicking the first button, a wizard for configuring the lambda function opens. The wizard for existing lambda functions offers two modes.



Function Name: MyLambdaTutorial_v2

The button **Export Function** offers the possibility to export lambda functions. For this purpose, the C++ project and the drivers of the lambda function are saved in a zip archive. Using the import function in the lambda template, the lambda function can be conveniently imported and integrated on another system. Via the mode **Edit lambda function** the already developed lambda function can be edited. The second button can be used to export the current lambda function. A Save File dialog opens for this purpose. This dialog can be used to specify the file location. Exporting starts after the selection. The exported lambda function can then be distributed to other systems. The other systems can import the lambda function from the lambda template using the wizard.
In the **Edit lambda function** mode, the symbols can be customized on the next page of the wizard.

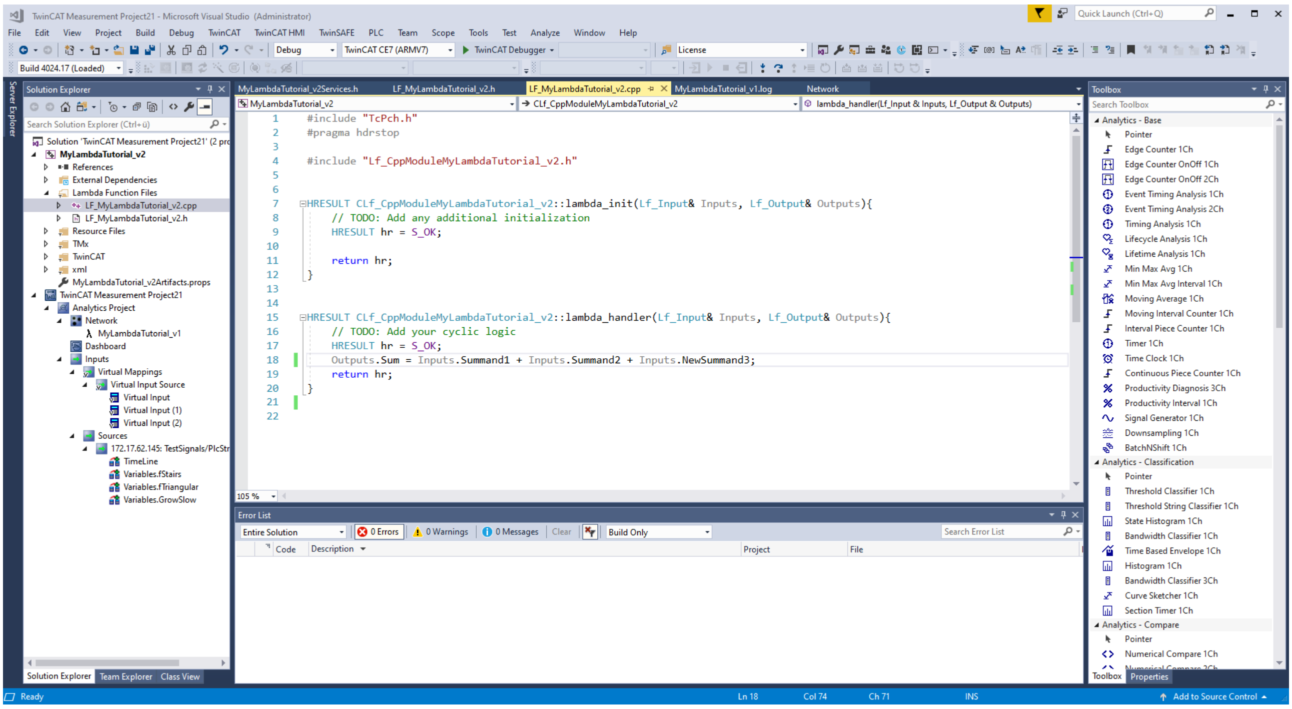


Datatype: LREAL

Type: Primitive

Category	Name	Datatype	Type	Count	Visualization
→	Summand1	LREAL	Primitive	1	Input
→	Summand2	LREAL	Primitive	1	Input
→	NewSummand3	LREAL	Primitive	1	Input
↦	Sum	LREAL	Primitive	1	Resul

In this example, another input named "NewSummand3" has been added. These changes are also included in the C++ file in the subsequent step.


TwinCAT Measurement Project21 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug TwinCAT TwinCAT HMI TwinSAFE PLC Team Scope Tools Test Analyze Window Help
Debug
TwinCAT CE7 (ARMV7)
TwinCAT Debugger
License
Solution Explorer
MyLambdaTutorial_v2Services.h
LF_MyLambdaTutorial_v2.h
LF_MyLambdaTutorial_v2.cpp
MyLambdaTutorial_v1.log
Network
MyLambdaTutorial_v2
CLf_CppModuleMyLambdaTutorial_v2
lambda_handler(Lf_Input & Inputs, Lf_Output & Outputs)
Toolbox
Error List
Entire Solution
0 Errors
0 Warnings
0 Messages
Build Only
Ready
Ln 18
Col 74
Ch 71
INS

```
#include "TcPch.h"
#pragma hdrstop

#include "Lf_CppModuleMyLambdaTutorial_v2.h"


HRESULT CLf_CppModuleMyLambdaTutorial_v2::lambda_init(Lf_Input& Inputs, Lf_Output& Outputs){
    // TODO: Add any additional initialization
    HRESULT hr = S_OK;

    return hr;
}


HRESULT CLf_CppModuleMyLambdaTutorial_v2::lambda_handler(Lf_Input& Inputs, Lf_Output& Outputs){
    // TODO: Add your cyclic logic
    HRESULT hr = S_OK;
    Outputs.Sum = Inputs.Summand1 + Inputs.Summand2 + Inputs.NewSummand3;
    return hr;
}

```

The new variable is included in the structure from inputs and can be used in the program code. The program code from the previous version was taken over. If symbols have been removed across a version, the corresponding changes must be made in the program code.
The further steps in the development of the lambda function are analogous to the procedure for the lambda template.

Debugging Lambda Functions

To debug lambda functions, the tools of TwinCAT C++ have to be used. For this purpose a new TwinCAT project must be created and the C++ project of the lambda function must be inserted under the C++ node. All TwinCAT C++ debugging options are available (see Debugging). The Debug tab can be used to attach to the TcScopeSever process. Afterwards the analysis is to be started in the Workbench and the set breakpoints are reached in the C++ project.

Tidying up the toolbox

After some lambda functions have been developed and tested, the toolbox may be very full. To clean up the toolbox, the corresponding lambda functions must be deleted from the Modules folder. The folder can be found under the path *<TwinCAT3Dir>/CustomConfig/Modules*.

6.4.14.1.6 Further steps for the lambda functions

Runtime deployment

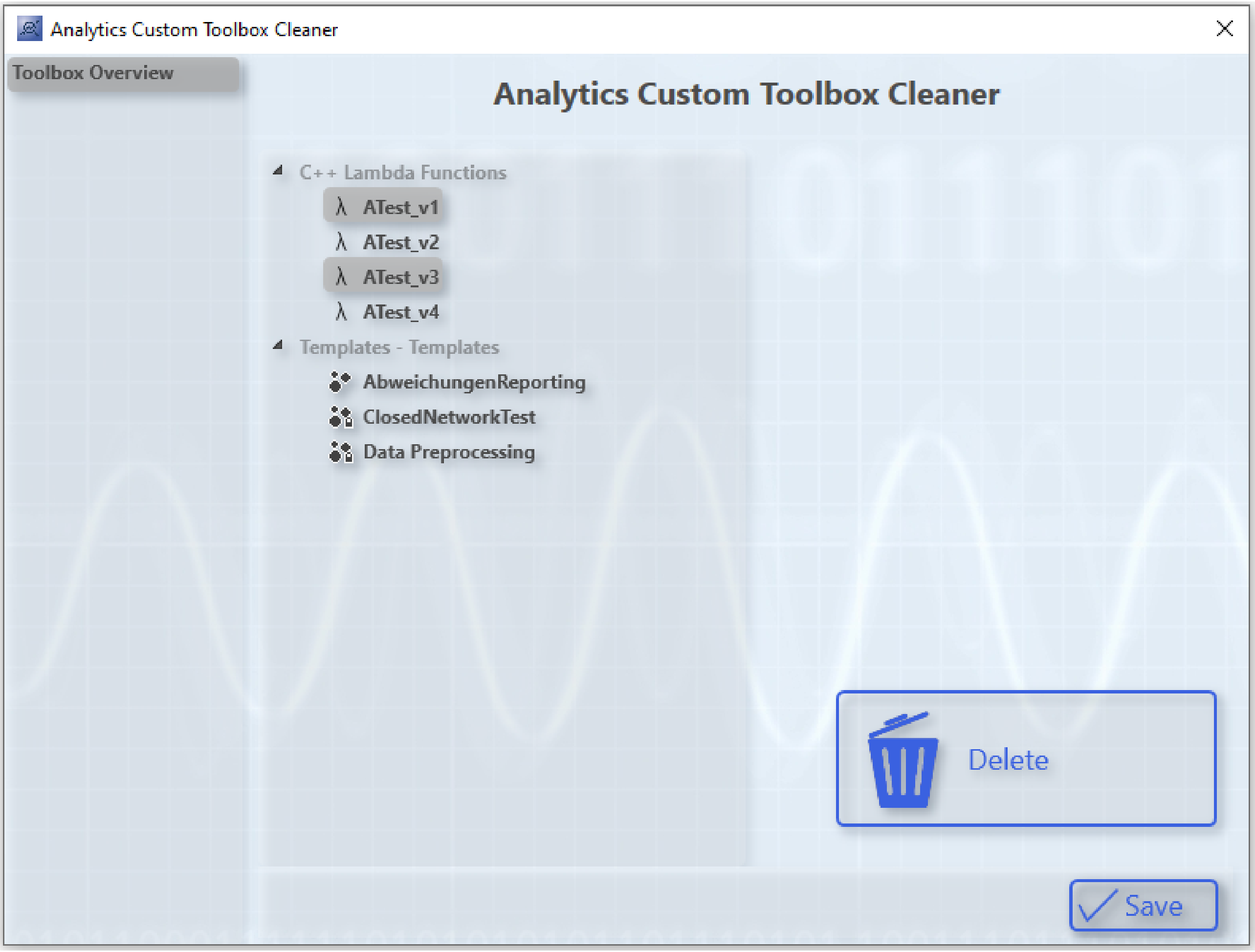
Lambda functions are also suitable for runtime deployment. The developed lambda function is created as a TcCOM module in the TwinCAT project for this purpose. A PLC function block handles the TcCOM module. The PLC function block calls the corresponding interface methods of the lambda function for this purpose. The interface methods of the lambda function, as well as the associated PLC function block are generated automatically and do not require any explicit configuration. Thus, the user-specific lambda functions can be downloaded into a TwinCAT Analytics Runtime and an individual 24/7 data analysis can be realized. More information can be found at the following link (see Runtime deployment [▸ 353]).

HMI One Click Dashboard

To display the results of the lambda functions, the integration into the HMI One Click Dashboard is ideal. A new control can be linked via the Dashboard node of an Analytics project. In the Analytics Dashboard Wizard, an HMI Control and the corresponding lambda function can be selected and linked. The Analytics Dashboard Wizard is described in more detail at the following link (see Manage dashboard structure and content in Analytics project [▸ 422]).

6.4.14.1.7 Analytics Custom Toolbox Cleaner

A wide range of toolbox elements can be created using the network templates and the Lambda functions. To conveniently remove these elements, Analytics Custom Toolbox Cleaner can be used. The following figure shows the wizard:



The individual toolbox elements can be selected via the tree structure. By holding down CTRL several elements can be selected in the building structure. The Delete button removes the elements from the building structure. The changes are confirmed by clicking on the Save button. If the wizard is closed otherwise, the changes will not be applied.

6.4.14.2 Algorithms from other TwinCAT libraries

By implementing the TwinCAT Filter and TwinCAT Condition Monitoring libraries, a further variety of algorithms is available. The integration of these algorithms in TwinCAT Analytics turns programming tasks into pure configuration tasks. This reduces the effort required to create an analysis and also enables these algorithms to be used together with those of the TwinCAT Analytics Library directly in the TwinCAT Analytics Workbench configurator or Service Tool. They can be found in the toolbox in the same way as the other algorithm groups. The additional algorithms can be used for the engineering module without an additional license. However, the deployment of the algorithms in the TwinCAT Analytics runtime and the generation of an HMI One-Click Dashboard additionally require the licenses of the respective library. The following list shows an overview of the integrated libraries and algorithms, the license required for deployment or dashboard generation, and a link to the documentation of the respective PLC function block.

Library	Algorithms	License	Documentation
TwinCAT Filter	*Filter*	TF3680 TC3 Filter / TF3600 TC3 Condition Monitoring	
	PT1		PT1 [▸ 317]
	IIRCoeff		IIRCoeff [▸ 318]
	IIRSpec		IIRSpec [▸ 318]
	PT2		PT2 [▸ 319]
	PT3		PT3 [▸ 319]
	PTn		PTn [▸ 320]
	IIRSos		IIRSos [▸ 320]
	Notch		Notch [▸ 320]
	LeadLag		LeadLag [▸ 321]
	PT2oscillation		PT2oscillation [▸ 321]
	PTt		PTt [▸ 322]
	Median		Median [▸ 322]
	ActualValue		ActualValue [▸ 323]
	Gaussian		Gaussian [▸ 323]
TwinCAT Condition Monitoring	*Condition Monitoring*	TF3600 TC3 Condition Monitoring	
	ArgSort		ArgSort [▸ 303]
	Crest Factor		Crest Factor [▸ 304]
	Crest Factor Plus		Crest Factor Plus [▸ 304]
	DiscreteClassification		Discrete Classification [▸ 305]
	Envelope		Envelope [▸ 305]
	Envelope Spectrum		Envelope Spectrum [▸ 306]
	Fatigue Analysis		Fatigue Analysis [▸ 307]
	Instantaneous Frequency		Instantaneous Frequency [▸ 308]
	Instantaneous Phase		Instantaneous Phase [▸ 303]
	Integrated RMS		Integrated RMS [▸ 309]
	Magnitude Spectrum		Magnitude Spectrum [▸ 310]
	Multi-Band RMS		Multi Band RMS [▸ 311]
	Power Spectrum		Power Spectrum [▸ 312]
	RMS		RMS [▸ 313]
	Spike Energy Spectrum		Spike Energy Spectrum [▸ 314]
	Vibration Assessment		Vibration Assessment [▸ 315]
	Watch Upper Thresholds		Watch Upper Thresholds [▸ 316]

The **TwinCAT Filter** library enables the implementation of digital filters of different types using various function blocks.

The library **TwinCAT Condition Monitoring** offers mathematical algorithms for condition monitoring of machines and plants.

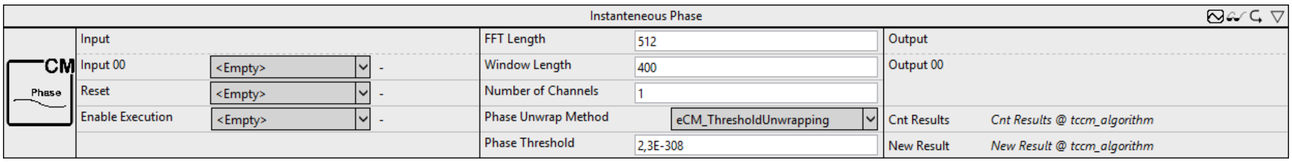
6.4.14.2.1 Condition Monitoring

The library **TwinCAT Condition Monitoring** offers mathematical algorithms for condition monitoring of machines and plants.

Algorithms Overview

Algorithm	Description	PLC function block
ArgSort [▸ 303]	Sorts the incoming arguments.	FB_CMA_ArgSort
Crest Factor [▸ 304]	Calculates the crest factor for each channel for single and multi-channel time series.	FB_CMA_CrestFactor
Crest Factor Plus [▸ 304]	Calculates the crest factor plus for each channel for single and multi-channel time series.	FB_CMA_CrestFactorPlus
Discrete Classification [▸ 305]	Classification of multi-channel data based on configurable threshold values.	FB_CMA_DiscreteClassification
Envelope [▸ 305]	Calculates the envelope of a time signal.	FB_CMA_Envelope
Envelope Spectrum [▸ 306]	Calculates the envelope spectrum of a time signal.	FB_CMA_EnvelopeSpectrum
Fatigue Analysis [▸ 307]	Service life analysis and damage calculation.	FB_CMA_RainflowCounting FB_CMA_MeanStressCorrection FB_CMA_MinersRule
Instantaneous Frequency [▸ 308]	Calculation of the instantaneous frequency of a time signal.	FB_CMA_InstantaneousFrequency
Instantaneous Phase [▸ 303]	Calculation of the instantaneous phase of a time signal.	FB_CMA_InstantaneousPhase
Integrated RMS [▸ 309]	Calculates (optionally integrated) RMS values for single- and multi-channel real-valued time series.	FB_CMA_IntegratedRMS
Magnitude Spectrum [▸ 310]	Calculates the magnitude spectrum (also referred to as amplitude spectrum) of a real-valued input signal.	FB_CMA_MagnitudeSpectrum
Multi Band RMS [▸ 311]	Calculated RMS value for single- and multi-channel real-valued time series for configurable frequency bands.	FB_CMA_MultiBandRMS
Power Spectrum [▸ 312]	Calculation of the power spectrum of a real-valued input signal, and optional decibel scaling.	FB_CMA_PowerSpectrum
RMS [▸ 313]	Calculates the temporal RMS value for single and multi-channel real-valued signals.	FB_CMA_RMS
Spike Energy Spectrum [▸ 314]	Analysis of peak energy of high-frequency signal components.	FB_CMA_SpikeEnergySpectrum
Vibration Assessment [▸ 315]	Vibration assessment of real-valued input signals.	FB_CMA_VibrationAssessment
Watch Upper Thresholds [▸ 316]	Configurable threshold value monitoring of multi-channel data.	FB_CMA_WatchUpperThresholds

6.4.14.2.1.1 Instantaneous Phase



Instanteneous Phase

Input		FFT Length	512	Output	
Input 00	<Empty> -	Window Length	400	Output 00	
Reset	<Empty> -	Number of Channels	1		
Enable Execution	<Empty> -	Phase Unwrap Method	eCM_ThresholdUnwrapping	Cnt Results	*Cnt Results @ tccm_algorithm*
		Phase Threshold	2,3E-308	New Result	*New Result @ tccm_algorithm*

Calculation of the instantaneous phase of a time signal.

The documentation of the corresponding PLC function block can be found here: FB_CMA_InstantaneousPhase

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

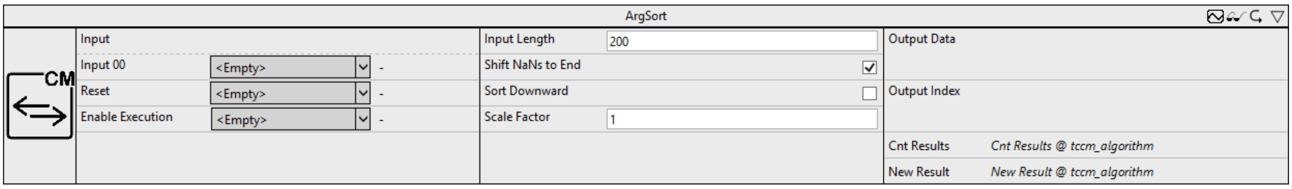
Configuration options

- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.

Phase Unwrap Method: defines the method used for phase-unwrapping with regard to the phase in multiples of 2 PI (see E_CM_UnwrapMethod).

- **Phase Threshold**: limit value for calculating the instantaneous phase. The value is related to the signal envelope. Interpretation: if the signal level is too low, the calculation of the phase is numerically too uncertain and cannot be evaluated reliably. 0 is then output as the phase.

6.4.14.2.1.2 ArgSort



ArgSort

Input		Input Length	200	Output Data	
Input 00	<Empty> -	Shift NaNs to End	☑		
Reset	<Empty> -	Sort Downward	□	Output Index	
Enable Execution	<Empty> -	Scale Factor	1		
				Cnt Results	*Cnt Results @ tccm_algorithm*
				New Result	*New Result @ tccm_algorithm*

Sorts the incoming arguments.

The documentation of the corresponding PLC function block can be found here: FB_CMA_ArgSort

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

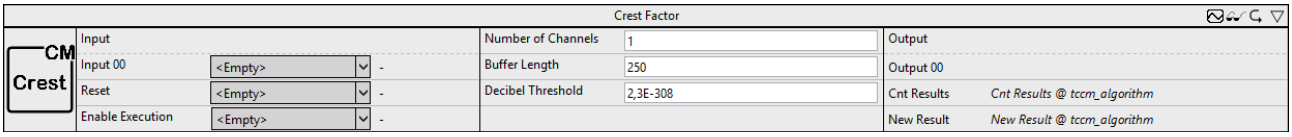
Configuration options

- **Input Length**: is the length of the input array.
- **Shift NaNs to End**: is a flag to select whether NaNs are shifted to the end of the output array.
- **Sort Downward**: is a flag with which you can select whether the data are to be sorted in ascending (`FALSE`) or descending (`TRUE`) order.
- **Scale Factor**: this parameter can be used to display directly the amplitudes with associated frequencies instead of the index position (Scale Factor = 1), for example.

Output values

- **Output Data**: output array with identical length of the input array, which contains the optionally ascending or descending sorted input values.
- **Output Index**: output array with identical length of the input array, which contains the (scalable) indices belonging to the sorted values in the input array.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.3 Crest Factor



Crest Factor

CM Crest

Input			Number of Channels	1	Output	
Input 00	<Empty>	-	Buffer Length	250	Output 00	
Reset	<Empty>	-	Decibel Threshold	2,3E-308	Cnt Results	*Cnt Results @ tccm_algorithm*
Enable Execution	<Empty>	-			New Result	*New Result @ tccm_algorithm*

Calculates the crest factor for each channel for single and multi-channel time series.

The documentation of the corresponding PLC function block can be found here: FB_CMA_CrestFactor

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

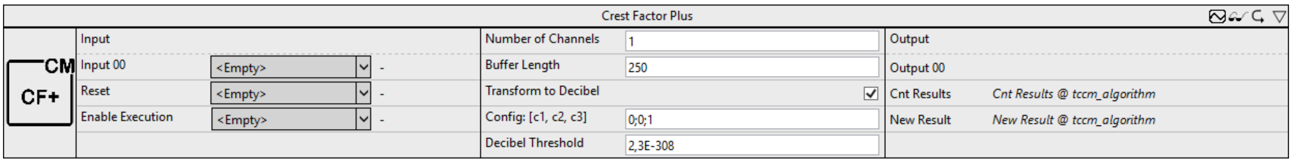
Configuration options

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Buffer Length**: is the number of input values per channel held in the internal buffer.
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)

Output values

- **Output**: crest factor of the associated input data stream.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.4 Crest Factor Plus



Crest Factor Plus

CM CF+

Input			Number of Channels	1	Output	
Input 00	<Empty>	-	Buffer Length	250	Output 00	
Reset	<Empty>	-	Transform to Decibel	☑	Cnt Results	*Cnt Results @ tccm_algorithm*
Enable Execution	<Empty>	-	Config: [c1, c2, c3]	0;0;1	New Result	*New Result @ tccm_algorithm*
			Decibel Threshold	2,3E-308		

Calculates the crest factor for each channel for single and multi-channel time series.

The documentation of the corresponding PLC function block can be found here: FB_CMA_CrestFactorPlus

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

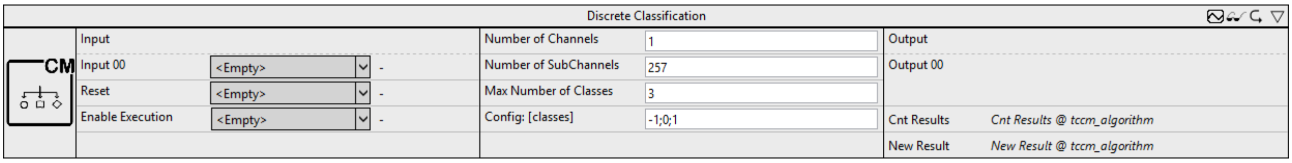
Configuration options

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Buffer Length**: is the number of input values per channel held in the internal buffer.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).
- **Config: [c1, c2, c3]:** configurable weights of the maximum amplitude (c1), the RMS value (c2) and the crest factor (c3).
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)

Output values

- **Output**: crest factor plus of the associated input data stream with respect to the configurable weights.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.5 Discrete Classification



Discrete Classification

Input		Number of Channels	1	Output	
Input 00	<Empty>	Number of SubChannels	257	Output 00	
Reset	<Empty>	Max Number of Classes	3		
Enable Execution	<Empty>	Config: [classes]	-1;0;1	Cnt Results	*Cnt Results @ tccm_algorithm*
				New Result	*New Result @ tccm_algorithm*

Classification of multi-channel data based on configurable threshold values.

The documentation of the corresponding PLC function block can be found here: FB_CMA_DiscreteClassification

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

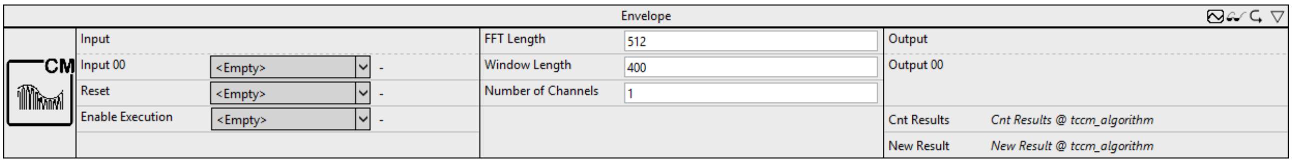
Configuration options

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Number of SubChannels**: defines the number of independent subchannels. This value is automatically adjusted to the length of the linked input array. If multiple channels are used, all lengths must match.
- **Max Number of Classes**: defines the maximum number of classes that will be configured. The value must be at least one.
- **Config: [classes]**: definition of the configurable threshold values.

Output values

- **Output**: output array with identical length of the input array, which contains the classifications of the entries.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.6 Envelope



Envelope

Input		FFT Length	512	Output	
Input 00	<Empty>	Window Length	400	Output 00	
Reset	<Empty>	Number of Channels	1		
Enable Execution	<Empty>			Cnt Results	*Cnt Results @ tccm_algorithm*
				New Result	*New Result @ tccm_algorithm*

Calculates the envelope of a time signal.

The documentation of the corresponding PLC function block can be found here: FB_CMA_Envelope

For time-synchronous presentation and further processing, see the section Timeshift [▶ 305].

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

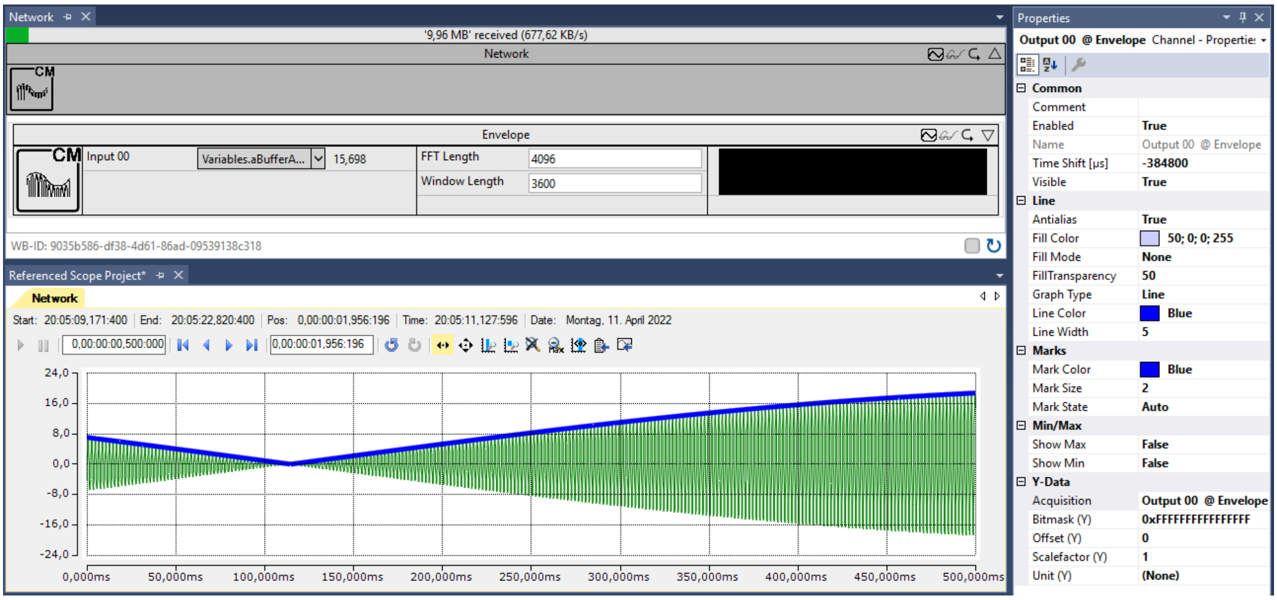
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.

Output values

- **Output**: output array with half window length containing the calculated envelope.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

Time shift

Due to the use of the Welch method (50% overlap) for the analysis as well as a segmented convolution of the analytical signal in the synthesis, four complete data buffers of length `nWindowLength/2` are needed to calculate a valid output signal. Furthermore, the result is delayed by `(nFFTLentgth + nWindowLength)/2` samples. This can be corrected by a suitable selection of the Time Shift at the channel of the output.



Network

'9,96 MB' received (677,62 KB/s)

Network

Envelope

Input 00	Variables.aBufferA...	15,698	FFT Length	4096
			Window Length	3600

[redacted]

WB-ID: 9035b586-df38-4d61-86ad-09539138c318

Referenced Scope Project*

Network

Start: 20:05:09,171:400 | End: 20:05:22,820:400 | Pos: 0,00:00:01,956:196 | Time: 20:05:11,127:596 | Date: Montag, 11. April 2022

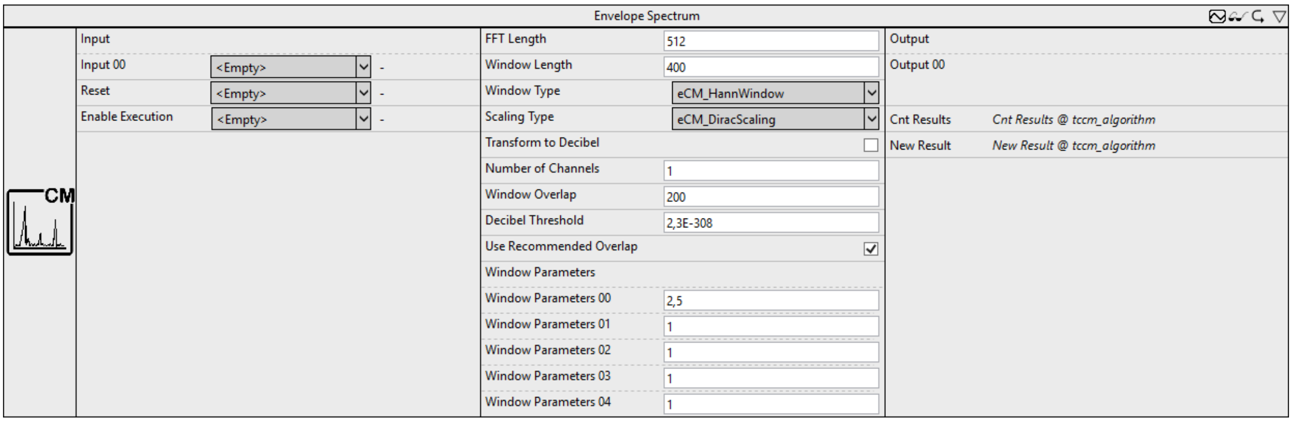
0,00:00:00,500:000 0,00:00:01,956:196

Properties

Output 00 @ Envelope Channel - Propertie:

Common	
Comment	
Enabled	True
Name	Output 00 @ Envelope
Time Shift [µs]	-384800
Visible	True
Line	
Antialias	True
Fill Color	50; 0; 0; 255
Fill Mode	None
FillTransparency	50
Graph Type	Line
Line Color	Blue
Line Width	5
Marks	
Mark Color	Blue
Mark Size	2
Mark State	Auto
Min/Max	
Show Max	False
Show Min	False
Y-Data	
Acquisition	Output 00 @ Envelope
Bitmask (Y)	0xFFFFFFFFFFFFFFFF
Offset (Y)	0
Scalefactor (Y)	1
Unit (Y)	(None)

6.4.14.2.1.7 Envelope Spectrum



Envelope Spectrum

Input		FFT Length	512	Output	
Input 00	<Empty> -	Window Length	400	Output 00	
Reset	<Empty> -	Window Type	eCM_HannWindow		
Enable Execution	<Empty> -	Scaling Type	eCM_DiracScaling	Cnt Results	*Cnt Results @ tccm_algorithm*
		Transform to Decibel	□	New Result	*New Result @ tccm_algorithm*
		Number of Channels	1		
		Window Overlap	200		
		Decibel Threshold	2,3E-308		
		Use Recommended Overlap	☑		
		Window Parameters			
		Window Parameters 00	2,5		
		Window Parameters 01	1		
		Window Parameters 02	1		
		Window Parameters 03	1		
		Window Parameters 04	1		

Calculates the envelope spectrum of a time signal.

The documentation of the corresponding PLC function block can be found here: FB_CMA_EnvelopeSpectrum

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

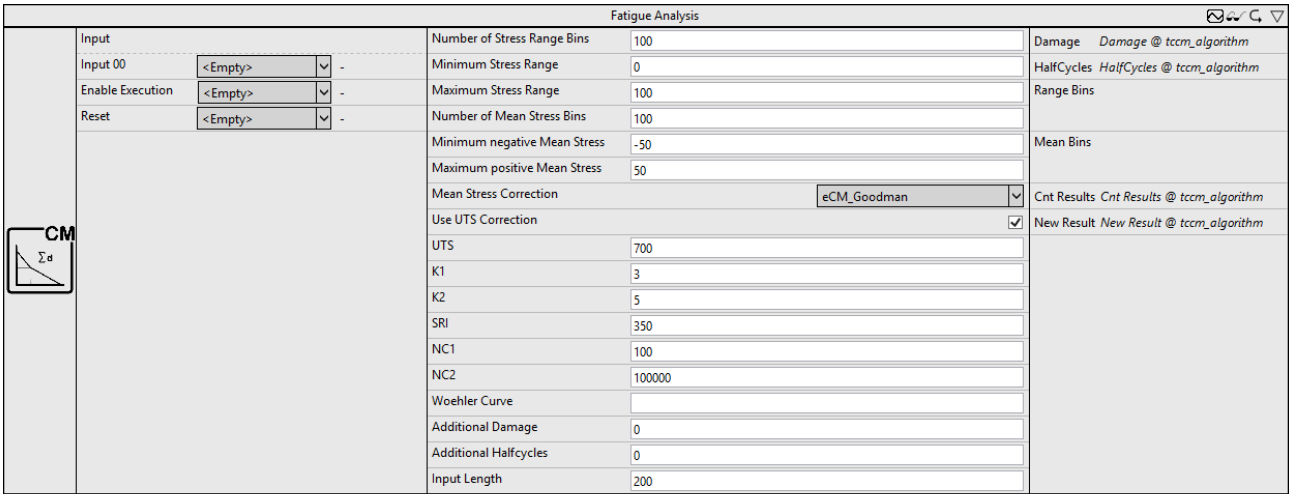
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Scaling Type**: Enables the selection of the scaling (of the type E_CM_ScalingType) to be used, in case absolute scaling is required. The default value is `eCM_DiracScaling`. When selecting the scaling the type of signal should be considered: either deterministic signals or wide-band signals with stochastic portion. Both types require different scalings.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: output array of length `N / 2 + 1` (for FFT length `N`), which contains the calculated spectral lines of the envelope spectrum.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.8 Fatigue Analysis



Input		
Input 00	<Empty>	-
Enable Execution	<Empty>	-
Reset	<Empty>	-

Number of Stress Range Bins	100
Minimum Stress Range	0
Maximum Stress Range	100
Number of Mean Stress Bins	100
Minimum negative Mean Stress	-50
Maximum positive Mean Stress	50
Mean Stress Correction	eCM_Goodman
Use UTS Correction	☑
UTS	700
K1	3
K2	5
SRI	350
NC1	100
NC2	100000
Woehler Curve	
Additional Damage	0
Additional Halfcycles	0
Input Length	200

Damage	*Damage @ tccm_algorithm*
HalfCycles	*HalfCycles @ tccm_algorithm*
Range Bins	
Mean Bins	
Cnt Results	*Cnt Results @ tccm_algorithm*
New Result	*New Result @ tccm_algorithm*

Service life analysis and damage calculation to estimate the fatigue process of monitored components.

The documentation of the corresponding PLC function blocks can be found here: FB_CMA_RainflowCounting, FB_CMA_MeanStressCorrection and FB_CMA_MinersRule

For the calculation of the Wöhler curve, the function F_CM_CalculateWoehlerCurve is executed on the basis of fictitious material parameters.

For more detailed explanations, refer to the Condition Monitoring documentation in the chapter Application Concepts, section Service Life Analysis and Damage Calculation.

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

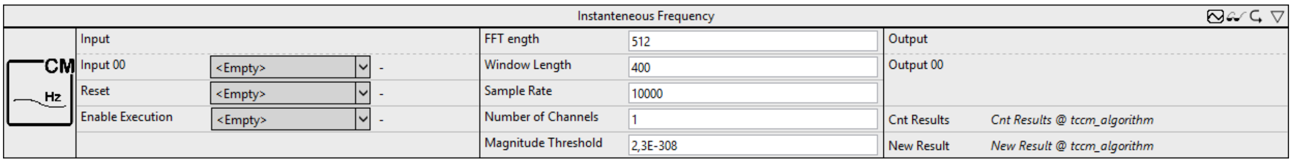
- **Number of Stress Range Bins**: defines the number of bins for the stress level. The total number of bins is "nBins + 2"; the value must be at least one. The first bin of each row contains the stress levels that are less than or equal to the defined minimum; analogously, all values greater than the defined maximum fall into the last bin of each row.
- **Minimum Stress Range**: defines the lower limit of the stress level. All values that are less than or equal to this value are accumulated in a bin.

- **Maximum Stress Range**: defines the upper limit of the stress level. Values that exceed the maximum are counted in the last bin.
- **Number of Mean Stress Bins**: defines the number of bins for the mean values. The value must be at least one. Two separate bins are added in the first/last column of the Halfcycle Count Matrix, where mean values are kept that are less than or equal to or, respectively, greater than the defined limits.
- **Minimum negative Mean Stress**: defines the lower limit of the mean values. All values that are less than or equal to this value are accumulated in a bin.
- **Maximum positive Mean Stress**: defines the upper limit of the mean values. Values that exceed the maximum are counted in the last bin.
- **Mean Stress Correction**: defines the correction mode to be used. The correction according to Goodman and Gerber is available as well as the option not to make a correction.
- **Use UTS Correction**: if active, a UTS correction of the stress matrix is performed.
- **UTS**: defines the tensile strength of the monitored material.
- **K1**: defines the gradient of the Wöhler curve in the range `N = 1..NC1.`
- **K2**: defines the gradient of the Wöhler curve starting at `N >= NC2.`
- **SRI**: defines the "Stress Range Intercept".
- **NC1**: defines the transition point for the UTS correction.
- **NC2**: defines the transition point between K1 and K2.
- **Woehler Curve**: definition of the Wöhler curve.
- **Additional Damage**: defines the constant damage from the beginning. The total damage is thus calculated from the sum of fDamage and the accumulated damage with respect to the configured Wöhler curve.
- **Additional Halfcycles**: defines the initial value of the counted half cycles at the beginning. The total number of cycles is calculated from the sum of nCycles and the current number of input data.
- **Input Length**: defines the length of the input data array for block processing.

Output values

- **Damage**: indicates the calculated fatigue damage with respect to the defined Wöhler curve.
- **HalfCycles**: indicates the number of half cycles counted.
- **Range Bins**: output array of length "Number of Stress Range Bins" with summed half cycles along the middle stress.
- **Mean Bins**: output array of length "Number of Mean Stress Bins" with summed half cycles along the absolute stress.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.9 Instantaneous Frequency


Instanteneous Frequency

Input		FFT ength	512	Output	
Input 00	<Empty> -	Window Length	400	Output 00	
Reset	<Empty> -	Sample Rate	10000		
Enable Execution	<Empty> -	Number of Channels	1	Cnt Results	*Cnt Results @ tccm_algorithm*
		Magnitude Threshold	2,3E-308	New Result	*New Result @ tccm_algorithm*

Calculation of the instantaneous frequency of a time signal.

The documentation of the corresponding PLC function block can be found here: FB_CMA_InstantaneousFrequency

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

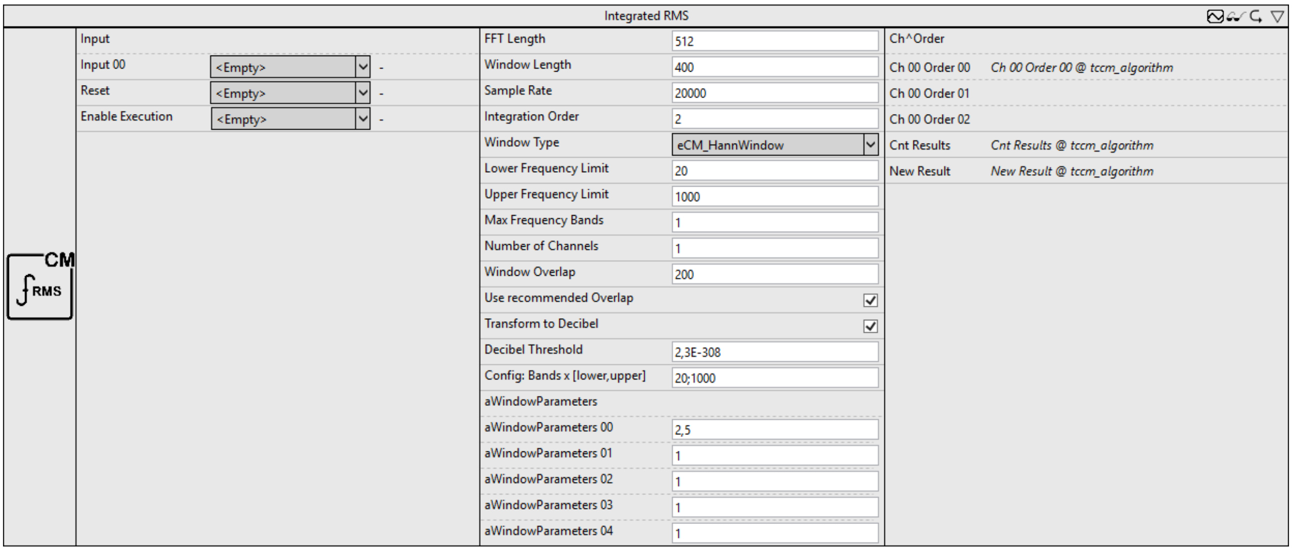
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.

- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Sample Rate**: sampling rate of the incoming time signal. The value is used for the scaling of the result in Hz.
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Magnitude Threshold**: defines the limit value for the numerical calculability of the instantaneous frequency.

Output values

- **Output**: output array of half window length containing the calculated instantaneous frequency.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.10 Integrated RMS



Input		
Input 00	<Empty>	-
Reset	<Empty>	-
Enable Execution	<Empty>	-

FFT Length	512
Window Length	400
Sample Rate	20000
Integration Order	2
Window Type	eCM_HannWindow
Lower Frequency Limit	20
Upper Frequency Limit	1000
Max Frequency Bands	1
Number of Channels	1
Window Overlap	200
Use recommended Overlap	☑
Transform to Decibel	☑
Decibel Threshold	2,3E-308
Config: Bands x [lower,upper]	20;1000
aWindowParameters	
aWindowParameters 00	2,5
aWindowParameters 01	1
aWindowParameters 02	1
aWindowParameters 03	1
aWindowParameters 04	1

Ch^Order	
Ch 00 Order 00	*Ch 00 Order 00 @ tccm_algorithm*
Ch 00 Order 01	
Ch 00 Order 02	
Cnt Results	*Cnt Results @ tccm_algorithm*
New Result	*New Result @ tccm_algorithm*

Calculates (optionally integrated) RMS values for single- and multi-channel real-valued time series.

The documentation of the corresponding PLC function block can be found here: FB_CMA_IntegratedRMS

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

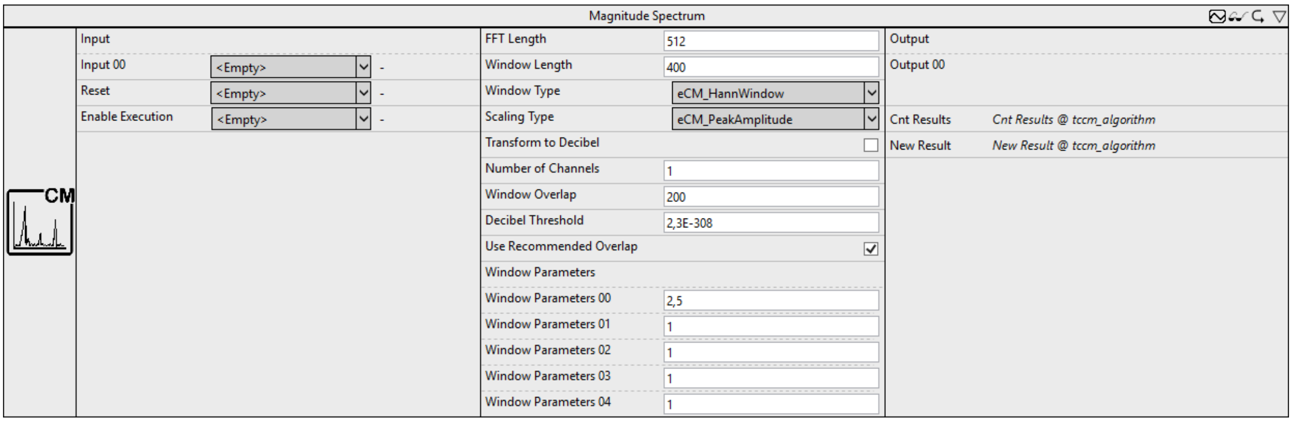
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Sample Rate**: sampling rate of the incoming time signal. The value is used for the scaling of the result in Hz.
- **Max Integration Order**: is the maximum order of integration. This must be an integer between zero and two. The number of the values determined per channel is (`Order+1`).
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Lower Frequency Limit**: lower limit of the considered frequency interval. The lower cut-off frequency must be at least the sampling rate divided by the FFT-length.
- **Upper Frequency Limit**: upper limit of the considered frequency interval. The upper cut-off frequency must be no greater than half the sampling rate and greater than the lower cut-off frequency.
- **Max Frequency Bands**: this value specifies the maximum number of frequency bands for which the RMS value is calculated.
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.

- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Config: MaxBands x [fmin, fmax]**: definition of the configurable lower and upper limit of the frequency bands.
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: output arrays corresponding to the number of frequency bands with channel and order specific RMS values.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.11 Magnitude Spectrum



Magnitude Spectrum

Input		FFT Length	512	Output	
Input 00	<Empty>	Window Length	400	Output 00	
Reset	<Empty>	Window Type	eCM_HannWindow		
Enable Execution	<Empty>	Scaling Type	eCM_PeakAmplitude	Cnt Results	*Cnt Results @ tccm_algorithm*
		Transform to Decibel	☐	New Result	*New Result @ tccm_algorithm*
		Number of Channels	1		
		Window Overlap	200		
		Decibel Threshold	2,3E-308		
		Use Recommended Overlap	☑		
		Window Parameters			
		Window Parameters 00	2,5		
		Window Parameters 01	1		
		Window Parameters 02	1		
		Window Parameters 03	1		
		Window Parameters 04	1		

Calculates the magnitude spectrum (also referred to as amplitude spectrum) of a real-valued input signal.

The documentation of the corresponding PLC function block can be found here: FB_CMA_MagnitudeSpectrum

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

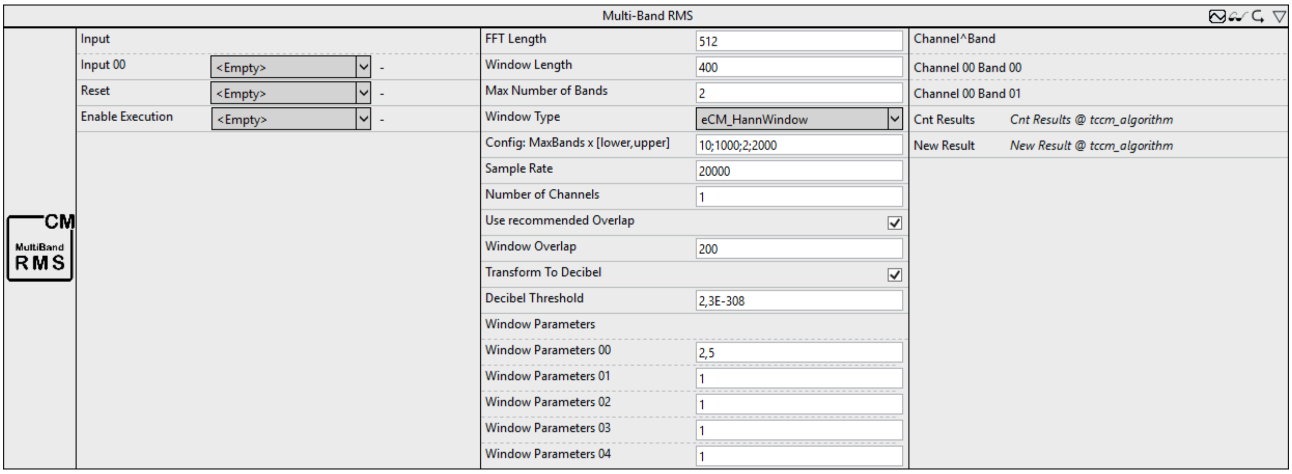
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Scaling Type**: Enables the selection of the scaling (of the type E_CM_ScalingType) to be used, in case absolute scaling is required. The default value is `eCM_DiracScaling`. When selecting the scaling the type of signal should be considered: either deterministic signals or wide-band signals with stochastic portion. Both types require different scalings.

- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: output array of length `N / 2 + 1` (for FFT length `N`), which contains the calculated spectral lines of the magnitude spectrum.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.12 Multi Band RMS



Multi-Band RMS

Input		
Input 00	<Empty>	-
Reset	<Empty>	-
Enable Execution	<Empty>	-

FFT Length	512
Window Length	400
Max Number of Bands	2
Window Type	eCM_HannWindow
Config: MaxBands x [lower,upper]	10;1000;2;2000
Sample Rate	20000
Number of Channels	1
Use recommended Overlap	☑
Window Overlap	200
Transform To Decibel	☑
Decibel Threshold	2,3E-308
Window Parameters	
Window Parameters 00	2,5
Window Parameters 01	1
Window Parameters 02	1
Window Parameters 03	1
Window Parameters 04	1

Channel^Band	
Channel 00 Band 00	
Channel 00 Band 01	
Cnt Results	*Cnt Results @ tccm_algorithm*
New Result	*New Result @ tccm_algorithm*

Calculated RMS value for single- and multi-channel real-valued time series for configurable frequency bands.

The documentation of the corresponding PLC function block can be found here: FB_CMA_MultiBandRMS

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

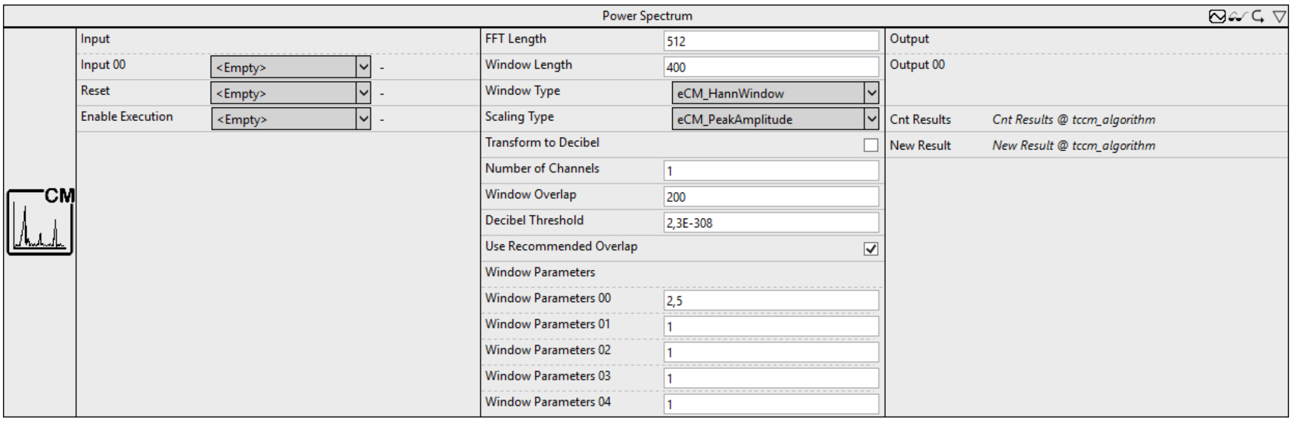
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Max Number of Bands**: this value specifies the maximum number of frequency bands for which the RMS value is calculated.
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Config: MaxBands x [fmin, fmax]**: definition of the configurable lower and upper limit of the frequency bands.

- **Sample Rate**: sampling rate of the incoming time signal. The value is used for the scaling of the result in Hz.
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: channel and band specific RMS values.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.13 Power Spectrum



Power Spectrum

Input		
Input 00	<Empty>	-
Reset	<Empty>	-
Enable Execution	<Empty>	-

FFT Length	512
Window Length	400
Window Type	eCM_HannWindow
Scaling Type	eCM_PeakAmplitude
Transform to Decibel	□
Number of Channels	1
Window Overlap	200
Decibel Threshold	2,3E-308
Use Recommended Overlap	☑
Window Parameters	
Window Parameters 00	2,5
Window Parameters 01	1
Window Parameters 02	1
Window Parameters 03	1
Window Parameters 04	1

Output	
Output 00	
Cnt Results	*Cnt Results @ tccm_algorithm*
New Result	*New Result @ tccm_algorithm*

Calculation of the power spectrum of a real-valued input signal, and optional decibel scaling.

The documentation of the corresponding PLC function block can be found here: FB_CMA_PowerSpectrum

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

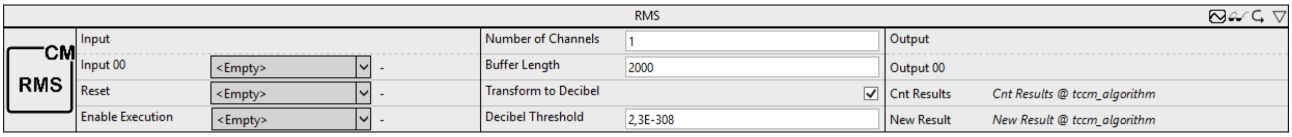
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Scaling Type**: Enables the selection of the scaling (of the type E_CM_ScalingType) to be used, in case absolute scaling is required. The default value is `eCM_DiracScaling`. When selecting the scaling the type of signal should be considered: either deterministic signals or wide-band signals with stochastic portion. Both types require different scalings.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: output array of length `N / 2 + 1` (for FFT length `N`), which contains the calculated spectral lines of the magnitude spectrum.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.14 RMS



RMS

Input			Number of Channels	1	Output	
Input 00	<Empty>	-	Buffer Length	2000	Output 00	
Reset	<Empty>	-	Transform to Decibel	☑	Cnt Results	*Cnt Results @ tccm_algorithm*
Enable Execution	<Empty>	-	Decibel Threshold	2,3E-308	New Result	*New Result @ tccm_algorithm*

Calculates the temporal RMS value for single and multi-channel real-valued signals.

The documentation of the corresponding PLC function block can be found here: FB_CMA_RMS

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

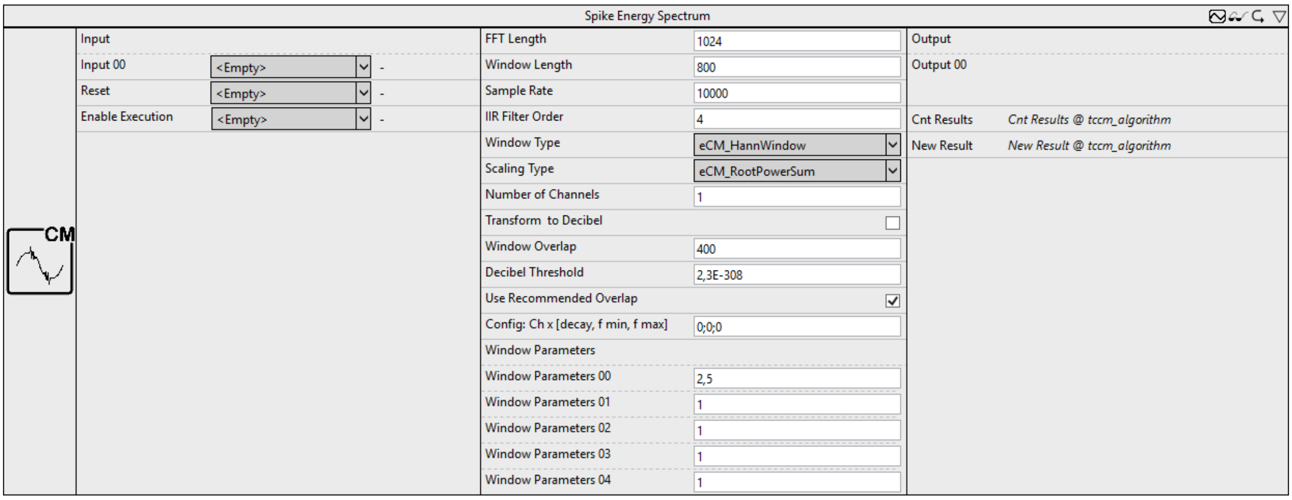
Configuration options

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Buffer Length**: is the number of input values per channel held in the internal buffer.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)

Output values

- **Output**: RMS value of the associated input data stream.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.15 Spike Energy Spectrum



Spike Energy Spectrum

Input		
Input 00	<Empty>	-
Reset	<Empty>	-
Enable Execution	<Empty>	-

FFT Length	1024
Window Length	800
Sample Rate	10000
IIR Filter Order	4
Window Type	eCM_HannWindow
Scaling Type	eCM_RootPowerSum
Number of Channels	1
Transform to Decibel	□
Window Overlap	400
Decibel Threshold	2,3E-308
Use Recommended Overlap	☑
Config: Ch x [decay, f min, f max]	0;0;0
Window Parameters	
Window Parameters 00	2,5
Window Parameters 01	1
Window Parameters 02	1
Window Parameters 03	1
Window Parameters 04	1

Output	
Output 00	
Cnt Results	*Cnt Results @ tccm_algorithm*
New Result	*New Result @ tccm_algorithm*

Analysis of peak energy of high-frequency signal components.

The documentation of the corresponding PLC function block can be found here: FB_CMA_SpikeEnergySpectrum

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

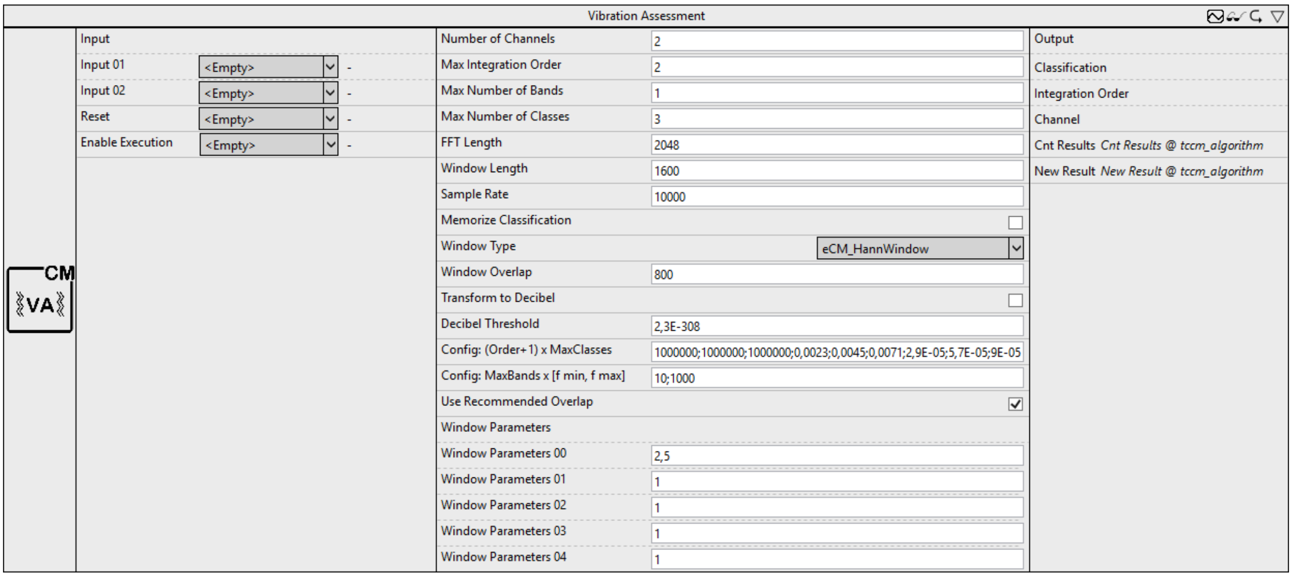
Configuration options

- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Sample Rate**: sampling rate of the incoming time signal. The value is used for the scaling of the result in Hz.
- **IIR Filter Order**: defines the order of the IIR filter used.
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Scaling Type**: Enables the selection of the scaling (of the type E_CM_ScalingType) to be used, in case absolute scaling is required. The default value is `eCM_DiracScaling`. When selecting the scaling the type of signal should be considered: either deterministic signals or wide-band signals with stochastic portion. Both types require different scalings.
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).
- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Config: Ch x [decay, fmin, fmax]**: definition of the configurable parameters: the decay time and the considered frequency band (by lower and upper limits). The decay time should optimally be chosen so that the peak energy can completely decay, i.e. 'decay time' > 1 / 'error frequency'.
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: Output array of length `N / 2 + 1` (for FFT length `N`), which contains the calculated spectral lines of the spike energy spectrum.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.16 Vibration Assessment



Vibration Assessment

Input		
Input 01	<Empty>	-
Input 02	<Empty>	-
Reset	<Empty>	-
Enable Execution	<Empty>	-

Number of Channels	2
Max Integration Order	2
Max Number of Bands	1
Max Number of Classes	3
FFT Length	2048
Window Length	1600
Sample Rate	10000
Memorize Classification	□
Window Type	eCM_HannWindow
Window Overlap	800
Transform to Decibel	□
Decibel Threshold	2,3E-308
Config: (Order+1) x MaxClasses	1000000;1000000;1000000;0,0023;0,0045;0,0071;2,9E-05;5,7E-05;9E-05
Config: MaxBands x [f min, f max]	10;1000
Use Recommended Overlap	☑
Window Parameters	
Window Parameters 00	2,5
Window Parameters 01	1
Window Parameters 02	1
Window Parameters 03	1
Window Parameters 04	1

Output
Classification
Integration Order
Channel
Cnt Results *Cnt Results @ tccm_algorithm*
New Result *New Result @ tccm_algorithm*

Vibration assessment of real-valued input signals based on ISO 10816-3.

The documentation of the corresponding PLC function block can be found here: FB_CMA_VibrationAssessment

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

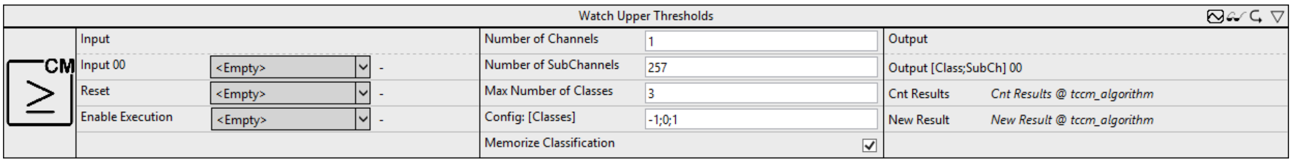
- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Max Integration Order**: is the maximum order of integration. This must be an integer between zero and two. The number of the values determined per channel is (`Order+1`).
- **Max Number of Bands**: this value specifies the maximum number of frequency bands for which the RMS value is calculated.
- **Max Number of Classes**: defines the maximum number of classes that will be configured. The value must be at least one.
- **FFT Length**: is the length of the FFT. It must be greater than one and an integral power of two.
- **Window Length**: is the length of the analysis window in samples. The length must be greater than one and an even number.
- **Sample Rate**: sampling rate of the incoming time signal. The value is used for the scaling of the result in Hz.
- **Memorize Classification**: if selected, the function block recalculates the number of the highest category and the corresponding channel at each step. Otherwise, the result values are stored when a limit value is exceeded until the reset is executed or a channel reaches a higher category.
- **WindowType**: Defines the used window function (of the type E_CM_WindowType). A good default value is the window type `eCM_HannWindow`.
- **Window Overlap**: defines the number of overlapping samples. This must be greater than or equal to zero.
- **Transform to Decibel**: is a Boolean value that indicates whether the result of the FFT is to be transformed to the decibel scale, according to transformation x -> 20 * log10(x).

- **Decibel Threshold**: is a very small floating point value greater than zero. Values that are less than this number are replaced with this value before any transformation to the decibel scale. (The purpose is the avoidance of value range errors. The logarithm of zero is not defined and strives infinitely towards minus for the limit value of small arguments. The same applies to the argument of the number zero, arg(0). The smallest possible value is 2.3e-308)
- **Config: (Order+1) x MaxClasses**: definition of the configurable threshold values with respect to the integration order.
- **Config: MaxBands x [fmin, fmax]**: definition of the configurable lower and upper limit of the frequency bands.
- **Use Recommended Overlap**: if selected, a recommended overlap is calculated internally (see F_CM_CalculateRecommendedOverlap).
- **Window Parameters**: contains the free parameters of selected window functions. When using `eCM_KaiserWindow`, the first entry defines the parameter beta; if `eCM_FlatTopWindow` is used, all parameters are used. See section Window functions.

Output values

- **Output**: the result is a one-dimensional array that contains three values for each frequency band: the highest calculated classification (in the range -1..'Max Number of Classes'), the associated integration order (in the range 0..'Max Integration Order') and the channel (in the range 1..'Number of Channels').
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

6.4.14.2.1.17 Watch Upper Thresholds



Watch Upper Thresholds

Input		Number of Channels	1	Output	
Input 00	<Empty> -	Number of SubChannels	257	Output [Class;SubCh] 00	
Reset	<Empty> -	Max Number of Classes	3	Cnt Results	*Cnt Results @ tccm_algorithm*
Enable Execution	<Empty> -	Config: [Classes]	-1;0;1	New Result	*New Result @ tccm_algorithm*
		Memorize Classification	☑		

Configurable threshold value monitoring of multi-channel data.

The documentation of the corresponding PLC function block can be found here: FB_CMA_WatchUpperThresholds

The documentation of the TwinCAT 3 Condition Monitoring PLC library can be found here: Overview.

Configuration options

- **Number of Channels**: defines the number of independent channels. This must be greater than zero.
- **Number of SubChannels**: defines the number of independent subchannels. This value is automatically adjusted to the length of the linked input array. If multiple channels are used, all lengths must match.
- **Max Number of Classes**: defines the maximum number of classes that will be configured. The value must be at least one.
- **Config: [classes]**: definition of the configurable threshold values.
- **Memorize Classification**: if selected, the function block recalculates the number of the highest category and the corresponding channel at each step. Otherwise, the result values are stored when a limit value is exceeded until the reset is executed or a channel reaches a higher category.

Output values

- **Output**: output array of length two for each input data stream. The values of the tuple describe the classification as well as the subchannel in which this class was identified.
- **Cnt Results**: specifies the number of output arrays calculated.
- **New Result**: is a Boolean value that indicates whether a new result was obtained in the current cycle.

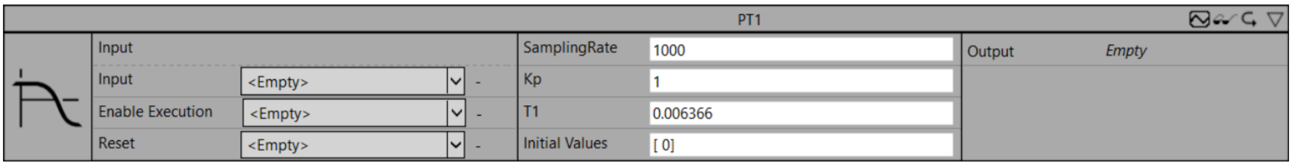
6.4.14.2.2 Filter

The library **TwinCAT Filter** offers algorithms for the implementation of digital filters.

Algorithms Overview

Algorithm	Description	PLC function block
PT1 [▶ 317]	Implementation of a first-order delay element (PT1 element).	FB_FTR_PT1
IIRCoeff [▶ 318]	Implementation of an IIR (Infinitive Impulse Response) filter via filter coefficients.	FB_FTR_IIRCoeff
IIRSpec [▶ 318]	Implementation of an IIR (Infinitive Impulse Response) filter via filter specifications.	FB_FTR_IIRSpec
PT2 [▶ 319]	Implementation of a second-order delay element (PT2 element).	FB_FTR_PT2
PT3 [▶ 319]	Implementation of a third-order delay element (PT3 element).	FB_FTR_PT3
PTn [▶ 320]	Implementation of a delay element of nth order with equal time constants.	FB_FTR_PTn
IIRSos [▶ 320]	Implementation of an IIR (Infinitive Impulse Response) filter via filter coefficients.	FB_FTR_IIRSos
Notch [▶ 320]	Implementation of a band-stop filter with narrow bandwidth.	FB_FTR_Notch
LeadLag [▶ 321]	Implementation of a first-order phase correction element.	FB_FTR_LeadLag
PT2oscillation [▶ 321]	Implementation of a second-order oscillatory delay element.	FB_FTR_PT2oscillation
PTt [▶ 322]	Implementation of a dead time element (PTt element).	FB_FTR_PTt
Median [▶ 322]	Implementation of a median filter.	FB_FTR_Median
ActualValue [▶ 323]	Implementation of a plausibility check and filtering of a measured input value.	FB_FTR_ActualValue
Gaussian [▶ 323]	Implementation of a Gaussian filter.	FB_FTR_Gaussian

6.4.14.2.2.1 PT1



PT1

Input
Input <Empty>
Enable Execution <Empty>
Reset <Empty>

SamplingRate 1000
Kp 1
T1 0.006366
Initial Values [0]

Output Empty

Implementation of a first-order delay element (PT1 element).

The documentation of the corresponding PLC function block can be found here: FB_FTR_PT1

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

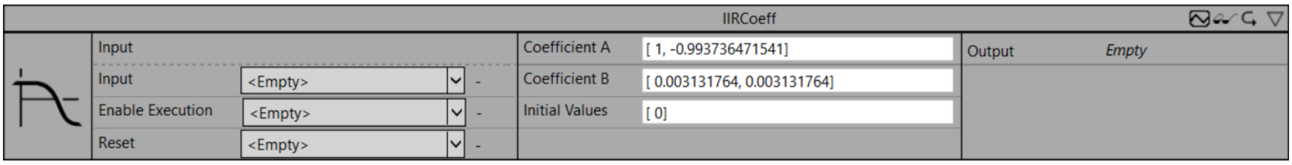
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Kp**: gain factor (greater than zero).
- **T1**: time constant T_1 in seconds (greater than zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.2 IIRCoeff



IIRCoeff

Input
- Input: <Empty>
- Enable Execution: <Empty>
- Reset: <Empty>

Coefficient A: [1, -0.993736471541]
Coefficient B: [0.003131764, 0.003131764]
Initial Values: [0]

Output: Empty

Implementation of an Infinite Impulse Response filter (IIR).

The documentation of the corresponding PLC function block can be found here: FB_FTR_IIRCoeff

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

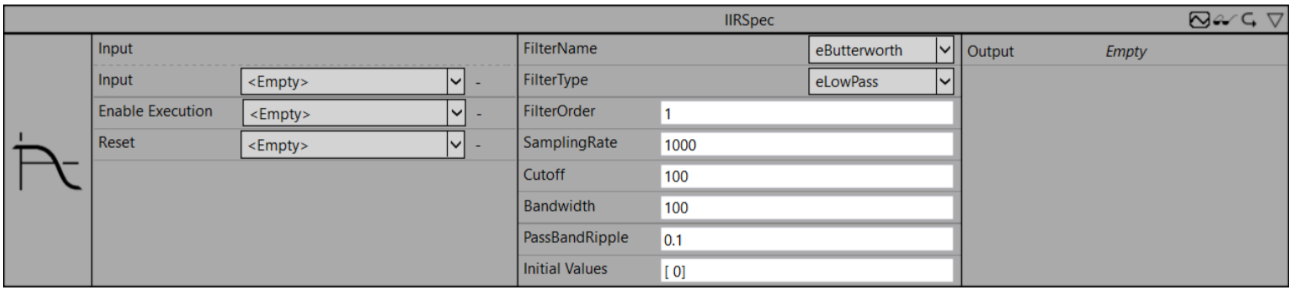
Configuration parameters

- **Coefficient A**: the coefficients a_k (denominator) [$a_{0,}$ $a_{1,}$ $a_{2,}$..., a_N] can be chosen freely.
- **Coefficient B**: the coefficients b_k (numerator) [$b_{0,}$ $b_{1,}$ $b_{2,}$..., b_M] can be chosen freely.
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.3 IIRSpec



IIRSpec

Input
- Input: <Empty>
- Enable Execution: <Empty>
- Reset: <Empty>

FilterName: eButterworth
FilterType: eLowPass
FilterOrder: 1
SamplingRate: 1000
Cutoff: 100
Bandwidth: 100
PassBandRipple: 0.1
Initial Values: [0]

Output: Empty

Implementation of an Infinite Impulse Response filter (IIR).

The documentation of the corresponding PLC function block can be found here: FB_FTR_IIRSpec

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

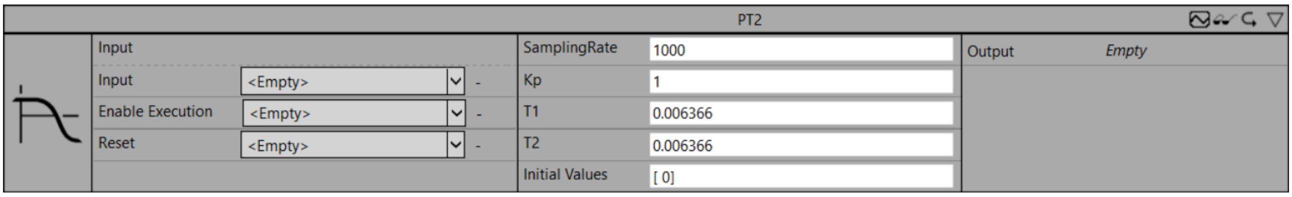
Configuration parameters

- **FilterName**: describes the filter implementation (Butterworth, Chebyshev).
- **FilterType**: describes the filter type (high-pass, low-pass, band-pass, band-stop).
- **FilterOrder**: filter order (max. 20 for high-pass and low-pass, max. 10 for band-pass and band-stop).
- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Cutoff**: cut-off frequency in Hz (greater than 0 and less than SamplingRate/2).
- **Bandwidth**: bandwidth in Hz with respect to band-pass and band-stop.
- **PassBandRipple**: passband ripple of the amplitude response in the passband of the filter in dB (greater than 0).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.4 PT2


PT2

Input		SamplingRate	1000	Output	Empty
Input	<Empty>	Kp	1		
Enable Execution	<Empty>	T1	0.006366		
Reset	<Empty>	T2	0.006366		
		Initial Values	[0]		

Implementation of a second-order delay element (PT2 element).

The documentation of the corresponding PLC function block can be found here: FB_FTR_PT2

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

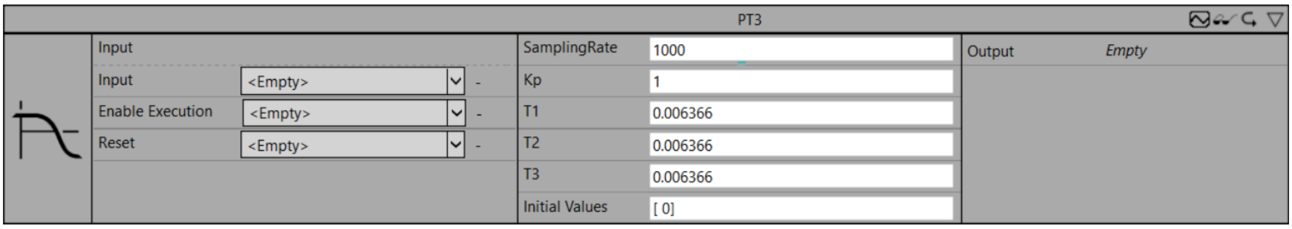
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Kp**: gain factor (greater than zero).
- **T1**: time constant T_1 in seconds (greater than zero).
- **T2**: time constant T_2 in seconds (greater than zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.5 PT3


PT3

Input		SamplingRate	1000	Output	Empty
Input	<Empty>	Kp	1		
Enable Execution	<Empty>	T1	0.006366		
Reset	<Empty>	T2	0.006366		
		T3	0.006366		
		Initial Values	[0]		

Implementation of a third-order delay element (PT3 element).

The documentation of the corresponding PLC function block can be found here: FB_FTR_PT3

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

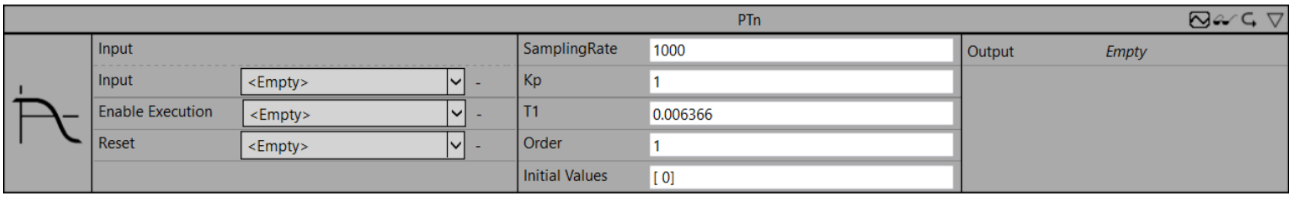
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Kp**: gain factor (greater than zero).
- **T1**: time constant T_1 in seconds (greater than zero).
- **T2**: time constant T_2 in seconds (greater than zero).
- **T3**: time constant T_3 in seconds (greater than zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.6 PTn



PTn

Input			SamplingRate	1000	Output	Empty
Input	<Empty>	-	Kp	1		
Enable Execution	<Empty>	-	T1	0.006366		
Reset	<Empty>	-	Order	1		
			Initial Values	[0]		

Implementation of a delay element of nth order with equal time constants.

The documentation of the corresponding PLC function block can be found here: FB_FTR_PTn

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

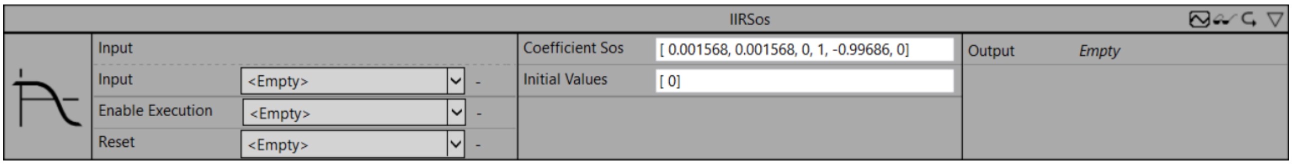
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Kp**: gain factor (greater than zero).
- **T1**: time constant T_1 in seconds (greater than zero).
- **Order**: order of the filter (1..10).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.7 IIRSos



IIRSos

Input			Coefficient Sos	[0.001568, 0.001568, 0, 1, -0.99686, 0]	Output	Empty
Input	<Empty>	-	Initial Values	[0]		
Enable Execution	<Empty>	-				
Reset	<Empty>	-				

Implementation of an Infinite Impulse Response filter (IIR).

The documentation of the corresponding PLC function block can be found here: FB_FTR_IIRSos.

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

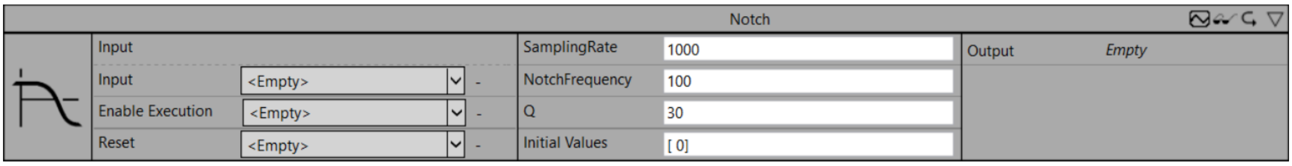
Configuration parameters

- **Coefficient Sos**: the coefficients $[b_{01}, b_{11}, b_{21}, a_{01}, a_{11}, a_{21}, b_{02}, b_{12}, b_{22}, a_{02}, a_{12}, a_{22}, \ldots, b_{0M}, b_{1M}, b_{2M}, a_{0M}, a_{1M}, a_{2M}]$ are freely selectable.
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.8 Notch



Notch

Input			SamplingRate	1000	Output	Empty
Input	<Empty>	-	NotchFrequency	100		
Enable Execution	<Empty>	-	Q	30		
Reset	<Empty>	-	Initial Values	[0]		

Implementation of a band-stop filter with narrow bandwidth.

The documentation of the corresponding PLC function block can be found here: FB_FTR_Notch

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

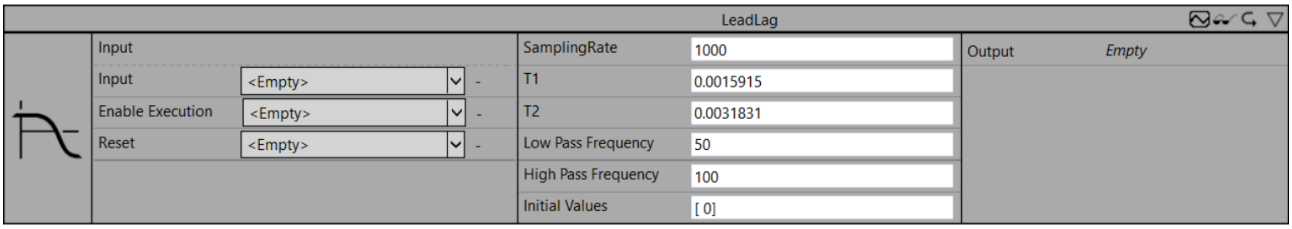
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **NotchFrequency**: notch frequency in Hz (greater than 0 and less than SamplingRate/2).
- **Q**: Q-factor = notch frequency/bandwidth (greater than zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.9 LeadLag



LeadLag

Input			
Input	<Empty>	-	
Enable Execution	<Empty>	-	
Reset	<Empty>	-	

SamplingRate	1000
T1	0.0015915
T2	0.0031831
Low Pass Frequency	50
High Pass Frequency	100
Initial Values	[0]

Output	Empty

Implementation of a first-order phase correction element.

The documentation of the corresponding PLC function block can be found here: FB_FTR_LeadLag.

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

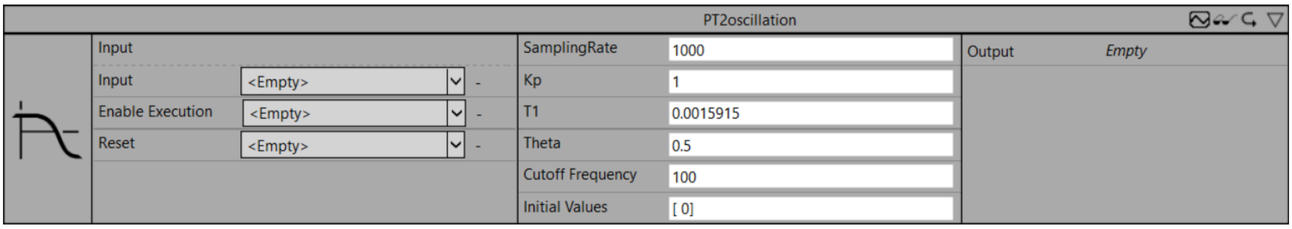
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **T1**: time constant T_1 in seconds (greater than zero).
- **T2**: time constant T_2 in seconds (greater than zero).
- **Low Pass Frequency (optional)**: low-pass frequency in Hz (greater than 0 and less than SamplingRate/2).
- **High Pass Frequency (optional)**: high-pass frequency in Hz (greater than 0 and less than SamplingRate/2).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.10 PT2oscillation



PT2oscillation

Input			
Input	<Empty>	-	
Enable Execution	<Empty>	-	
Reset	<Empty>	-	

SamplingRate	1000
Kp	1
T1	0.0015915
Theta	0.5
Cutoff Frequency	100
Initial Values	[0]

Output	Empty

Implementation of a second-order oscillatory delay element.

The documentation of the corresponding PLC function block can be found here: FB_FTR_PT2oscillation

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

Configuration parameters

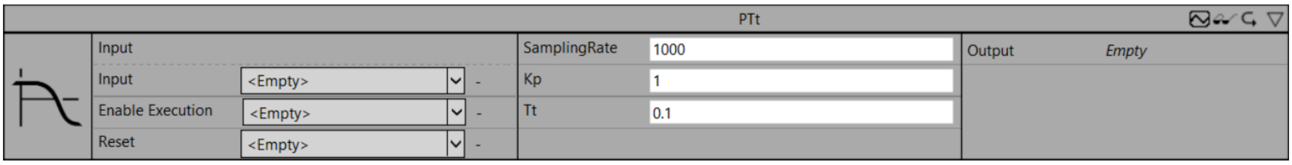
- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Kp**: gain factor (greater than zero).

- **T1**: time constant T_1 in seconds (greater than zero).
- **Theta**: damping factor (greater than zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.11 PTt



PTt

Input

Input <Empty> -

Enable Execution <Empty> -

Reset <Empty> -

SamplingRate 1000

Kp 1

Tt 0.1

Output *Empty*

Implementation of a dead time element (PTt element).

The documentation of the corresponding PLC function block can be found here: FB_FTR_PTt

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

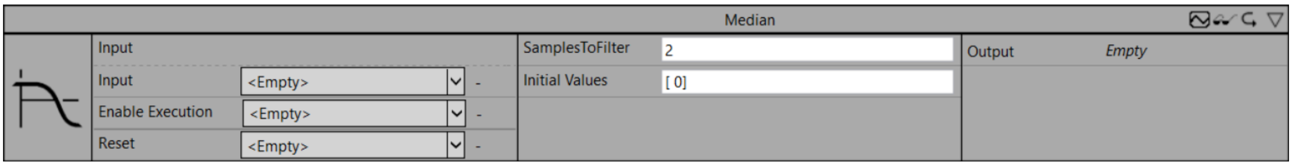
Configuration parameters

- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Kp**: gain factor (greater than zero).
- **Tt**: dead time T_t in seconds (greater than zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.12 Median



Median

Input

Input <Empty> -

Enable Execution <Empty> -

Reset <Empty> -

SamplesToFilter 2

Initial Values [0]

Output *Empty*

Implementation of a median filter.

The documentation of the corresponding PLC function block can be found here: FB_FTR_Median

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

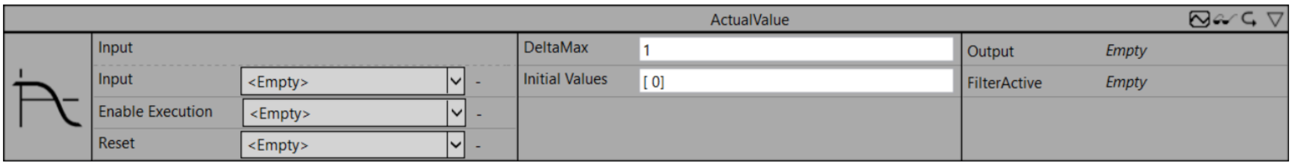
Configuration parameters

- **SamplesToFilter**: number of samples for calculating the sliding average value (often referred to as window size).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.2.13 ActualValue


ActualValue

Input		DeltaMax	1	Output	Empty
Input	<Empty> -	Initial Values	[0]	FilterActive	Empty
Enable Execution	<Empty> -				
Reset	<Empty> -				

Implementation of a plausibility check and filtering of a measured input value.

The documentation of the corresponding PLC function block can be found here: FB_FTR_ActualValue

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

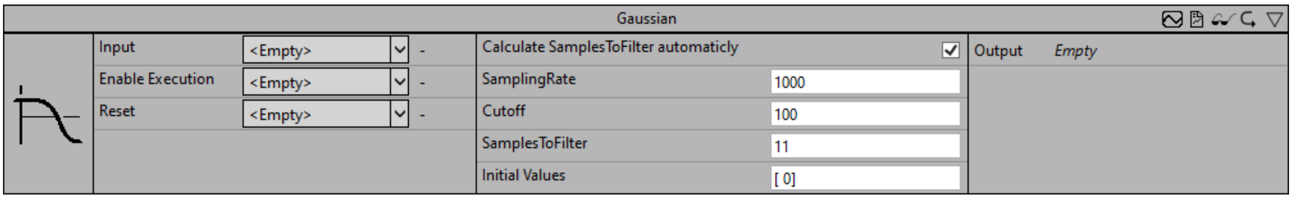
Configuration parameters

- **DeltaMax**: maximum difference between two input values in sequence (greater than or equal to zero).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.
- **FilterActive**: indicates which elements of the input signal have been changed.

6.4.14.2.2.14 Gaussian


Gaussian

Input	<Empty> -	Calculate SamplesToFilter automaticly	☑	Output	Empty
Enable Execution	<Empty> -	SamplingRate	1000		
Reset	<Empty> -	Cutoff	100		
		SamplesToFilter	11		
		Initial Values	[0]		

Implementation of an Infinite Impulse Response filter (IIR).

The documentation of the corresponding PLC function block can be found here: FB_FTR_IIRSpec

The documentation of the TwinCAT 3 Filter PLC library can be found here: Overview

Configuration parameters

- **Calculate SamplesToFilter automatically**: Is a boolean value that indicates whether SamplesToFilter should be calculated automatically.
- **SamplingRate**: sampling rate f_s in Hz (greater than zero).
- **Cutoff**: cut-off frequency in Hz (greater than 0 and less than SamplingRate/2).
- **SamplesToFilter**: number of samples for calculating the sliding average value (often referred to as window size).
- **Initial Values (optional)**: initial values define the internal state of the filter.

Output values

- **Output**: manipulated output signal.

6.4.14.2.3 Power Monitoring

Beckhoff offers various terminals for implementing power grid analysis and monitoring in its I/O portfolio. In a single- or three-phase grid, power measurement terminals such as the EL3403 or EL3413 directly provide RMS values for current and voltage, as well as active, reactive and apparent power. The EL3773 and EL3783 power monitoring terminals can also be used to log the raw data for current and voltage, so that grid events can be detected even more accurately, and various values can be calculated in the controller itself.

The TwinCAT 3 Power Monitoring Function is a PLC library for evaluating raw current and voltage data provided by the power monitoring terminals.

The library provides function blocks for calculating RMS values for current, voltage and power. These can be output as instantaneous or average values. The function block also offers maximum and minimum values. Frequency and frequency spectra can be determined, such as e.g., harmonics in the grid and their load in the form of the Total Harmonic Distortion (THD). Furthermore, the library offers function blocks which can be used to determine frequencies and rotary fields.

General algorithms

Algorithm	Description	PLC function block
Scaling [▶ 325]	Scaling of the input data for current and voltage	FB_PMA_Scaling
Scaling EL3773 [▶ 325]	Scaling of the input data of current and voltage provided as inputs by EL3773 EtherCAT Terminals.	FB_PMA_Scaling_EL3773
Scaling EL3783 [▶ 326]	Scaling of the input data of current and voltage provided as inputs by EL3783 EtherCAT Terminals.	FB_PMA_Scaling_EL3783

Single-phase algorithms

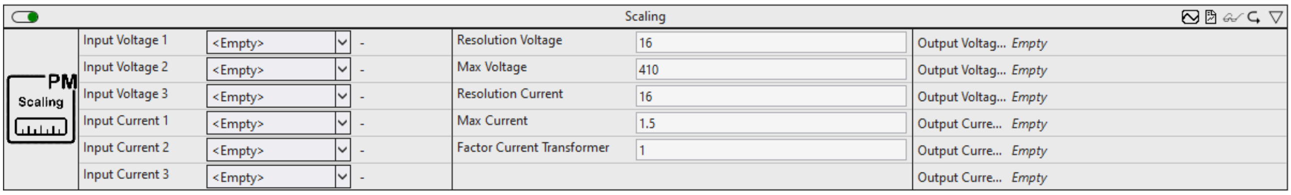
Algorithm	Description	PLC function block
Frequency Period 1Ph [▶ 327]	Calculation of the base frequency of the input signal.	FB_PMA_Frequency_Period_1Ph
Basic Values Period 1Ph [▶ 328]	Calculation of analysis values for the signal curve of current and voltage over time.	FB_PMA_BasicValues_Period_1Ph
Power Values Period 1Ph [▶ 329]	Calculation of characteristic power values of the connected loads based on the signal period.	FB_PMA_PowerValues_Period_1Ph
Harmonics Period 1Ph [▶ 330]	Calculation of the harmonics of current and voltage based on the signal period.	FB_PMA_Harmonics_Period_1Ph
Spectrum 1Ph [▶ 331]	Calculation of the magnitude spectra of the current and voltage values.	FB_PMA_Spectrum_1Ph
Spectrum Quantiles 1Ph [▶ 331]	Calculation of the magnitude spectra of the current and voltage values. In addition, p-quantiles of the spectrum distribution can be calculated.	FB_PMA_Spectrum_Quantiles_1Ph
Power Values 1Ph [▶ 333]	Calculation of power characteristics of the connected loads based on calculations in the frequency range.	FB_PMA_PowerValues_1Ph
Harmonics 1Ph [▶ 334]	Calculation of harmonics of current and voltage.	FB_PMA_Harmonics_1Ph

Three-phase algorithms

Algorithm	Description	PLC function block
Frequency Period 3Ph [▶ 335]	Calculation of the base frequency of the input signal.	FB_PMA_Frequency_Period_3Ph
Basic Values Period 3Ph [▶ 336]	Calculation of analysis values for the signal curve of current and voltage over time.	FB_PMA_BasicValues_Period_3Ph
Power Values Period 3Ph [▶ 337]	Calculation of characteristic power values of the connected loads based on the signal period.	FB_PMA_PowerValues_Period_3Ph
Harmonics Period 3Ph [▶ 338]	Calculation of the harmonics of current and voltage based on the signal period.	FB_PMA_Harmonics_Period_3Ph
Spectrum 3Ph [▶ 339]	Calculation of the magnitude spectra of the current and voltage values.	FB_PMA_Spectrum_3Ph
Spectrum Quantiles 3Ph [▶ 340]	Calculation of the magnitude spectra of the current and voltage values. In addition, p-quantiles of the spectrum distribution can be calculated.	FB_PMA_Spectrum_Quantiles_3Ph
Power Values 3Ph [▶ 341]	Calculation of power characteristics of the connected loads based on calculations in the frequency range.	FB_PMA_PowerValues_3Ph
Harmonics 3Ph [▶ 343]	Calculation of harmonics of current and voltage.	FB_PMA_Harmonics_3Ph

6.4.14.2.3.1 General

6.4.14.2.3.1.1 Scaling



Scaling

PM Scaling

Input Voltage 1	<Empty>	Resolution Voltage	16	Output Voltag...	Empty
Input Voltage 2	<Empty>	Max Voltage	410	Output Voltag...	Empty
Input Voltage 3	<Empty>	Resolution Current	16	Output Voltag...	Empty
Input Current 1	<Empty>	Max Current	1.5	Output Curre...	Empty
Input Current 2	<Empty>	Factor Current Transformer	1	Output Curre...	Empty
Input Current 3	<Empty>			Output Curre...	Empty

Scaling of the input data for current and voltage, which are provided as inputs by EtherCAT Terminals, for example.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Scaling.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Resolution Voltage:** Resolution of the voltage in bits. Value range: 1-32.
- **Max Voltage:** Specifies the maximum amplitude of the input voltage.
- **Resolution Current:** Resolution of the current in bits. Value range: 1-32.
- **Max Current:** Specifies the maximum amplitude of the input current.
- **Factor Current Transformer:** Factor of the current trans former. This is calculated by dividing the input current by the output current.

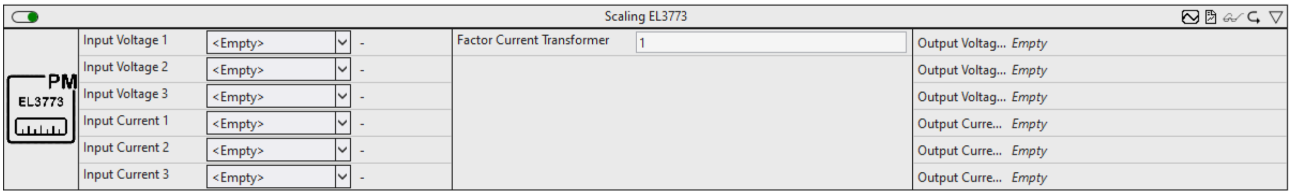
Input values

- **Input Voltage 1 ... Input Voltage 3:** Raw values of the voltage of the individual phases.
- **Input Current 1 ... Input Current 3:** Raw values of the current of the individual phases.

Output values

- **Output Voltage 1 ... Input Voltage 3:** Scaled voltage values of the individual phases.
- **Output Current 1 ... Input Current 3:** Scaled current values of the individual phases.

6.4.14.2.3.1.2 Scaling EL3773



Scaling EL3773

PM EL3773

Input Voltage 1	<Empty>	Factor Current Transformer	1	Output Voltag...	Empty
Input Voltage 2	<Empty>			Output Voltag...	Empty
Input Voltage 3	<Empty>			Output Voltag...	Empty
Input Current 1	<Empty>			Output Curre...	Empty
Input Current 2	<Empty>			Output Curre...	Empty
Input Current 3	<Empty>			Output Curre...	Empty

Scaling of the input data of current and voltage provided as inputs by EL3773 EtherCAT Terminals.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Scaling_EL3773.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Factor Current Transformer:** Factor of the current trans former. This is calculated by dividing the input current by the output current.

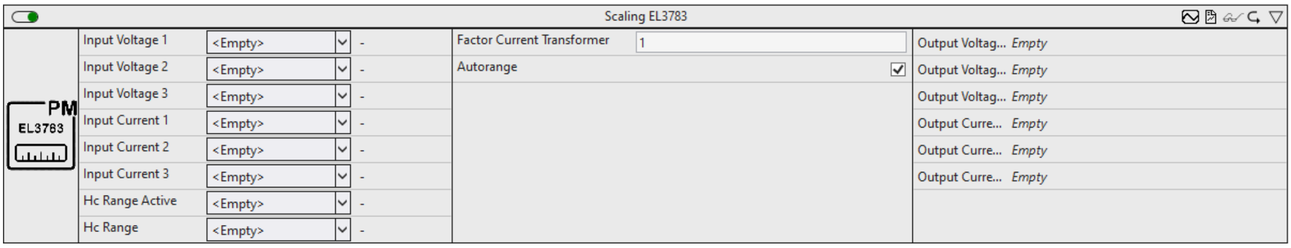
Input values

- **Input Voltage 1 ... Input Voltage 3:** Raw values of the voltage of the individual phases.
- **Input Current 1 ... Input Current 3:** Raw values of the current of the individual phases.

Output values

- **Output Voltage 1 ... Input Voltage 3:** Scaled voltage values of the individual phases.
- **Output Current 1 ... Input Current 3:** Scaled current values of the individual phases.

6.4.14.2.3.1.3 Scaling EL3783



Scaling EL3783

Input Voltage 1	<Empty>	-	Factor Current Transformer	1	Output Voltag... Empty
Input Voltage 2	<Empty>	-	Autorange	☑	Output Voltag... Empty
Input Voltage 3	<Empty>	-			Output Voltag... Empty
Input Current 1	<Empty>	-			Output Curre... Empty
Input Current 2	<Empty>	-			Output Curre... Empty
Input Current 3	<Empty>	-			Output Curre... Empty
Hc Range Active	<Empty>	-			
Hc Range	<Empty>	-			

Scaling of the input data of current and voltage provided as inputs by EL3783 EtherCAT Terminals.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Scaling_EL3783.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Factor Current Transformer:** Factor of the current trans former. This is calculated by dividing the input current by the output current.
- **Autorange:** Automatic range selection activated in the EtherCAT Terminal.

Input values

- **Input Voltage 1 ... Input Voltage 3:** Raw values of the voltage of the individual phases.
- **Input Current 1 ... Input Current 3:** Raw values of the current of the individual phases.
- **Hc Range Active:** *TRUE* if the Auto-Range mode is active on the terminal.
- **Hc Range:** The current measuring range information of the EL3783 in Auto-Range mode.
- **Use 5A range (optional):** If the value is *TRUE*, the 5A measuring range of the EL3783 is used. If it is *FALSE*, the 1 A measurement range is used. Cannot be combined with Auto-Range mode.

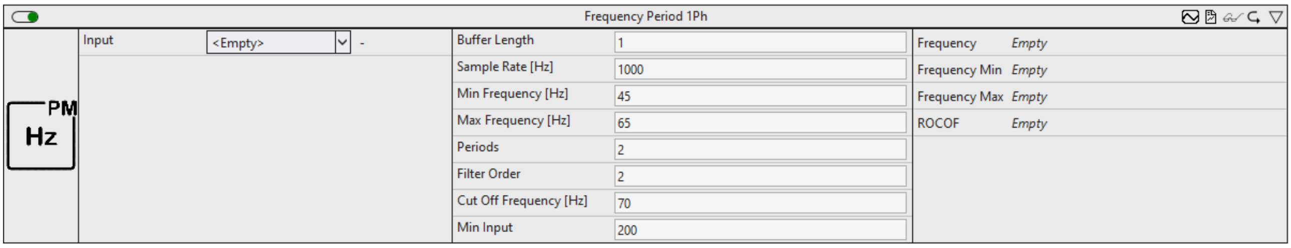
Output values

- **Output Voltage 1 ... Input Voltage 3:** Scaled voltage values of the individual phases.
- **Output Current 1 ... Input Current 3:** Scaled current values of the individual phases.

6.4.14.2.3.2 Single-phase

6.4.14.2.3.2.1 Based on the signal period

6.4.14.2.3.2.1.1 Frequency Period 1Ph



Frequency Period 1Ph

Input	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Min Frequency [Hz]	45
Max Frequency [Hz]	65
Periods	2
Filter Order	2
Cut Off Frequency [Hz]	70
Min Input	200

Frequency	*Empty*
Frequency Min	*Empty*
Frequency Max	*Empty*
ROCOF	*Empty*

The *Frequency Period* module calculates the base frequency of the input signal. To do this, the signal is first filtered with a Butterworth low-pass filter. The zero crossings of the input signal are then determined from the filtered values, and the frequency is calculated from their difference. The results refer to one or more periods, depending on the configuration. The statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Frequency_Period_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Frequency [Hz]**: Minimum measurement frequency to be expected.
- **Max Frequency [Hz]:** Maximum measurement frequency to be expected.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).
- **Filter Order:** Order of the low-pass filter. The stability of the filter must be considered for the setting. Only values up to the tenth order are allowed.
- **Cut Off Frequency [Hz]:** Cut-off frequency of the low-pass filter.
- **Min Input:** Minimum input value (RMS) over one period. This parameter prevents the calculation of input values that are too small.
- **ROCOF Avg Window Order:** Window size of the sliding average value for the calculation of the ROCOF.

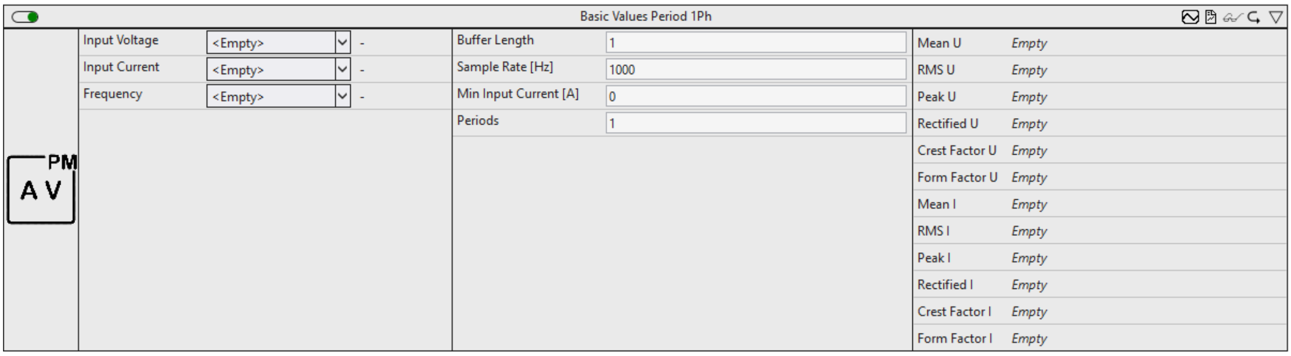
Input values

- **Input:** Input values of current or voltage.
- **Reset Statistics (optional):** *TRUE* resets the min and max values of the outputs.

Output values

- **Frequency:** Frequency determined by two or more zero crossings.
- **Frequency Min:** Smallest value of the calculated frequency. Can be reset via the *Reset Statistics* input.
- **Frequency Max:** Largest value of the calculated frequency. Can be reset via the *Reset Statistics* input.
- **ROCOF:** Rate of change of frequency (ROCOF).

6.4.14.2.3.2.1.2 Basic Values Period 1Ph



Basic Values Period 1Ph

Input Voltage	<Empty>	-
Input Current	<Empty>	-
Frequency	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Min Input Current [A]	0
Periods	1

Mean U	Empty
RMS U	Empty
Peak U	Empty
Rectified U	Empty
Crest Factor U	Empty
Form Factor U	Empty
Mean I	Empty
RMS I	Empty
Peak I	Empty
Rectified I	Empty
Crest Factor I	Empty
Form Factor I	Empty

The *Basic Values Period* module calculates analysis values for the signal curve of current and voltage over time. These include the mean value, the RMS value, the peak value, the rectified value, the crest factor and the form factor for current and voltage. The results refer to a configurable number of signal periods. The period duration refers to the frequency specified at the input at the start of the period. The optional statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_BasicValues_Period_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).

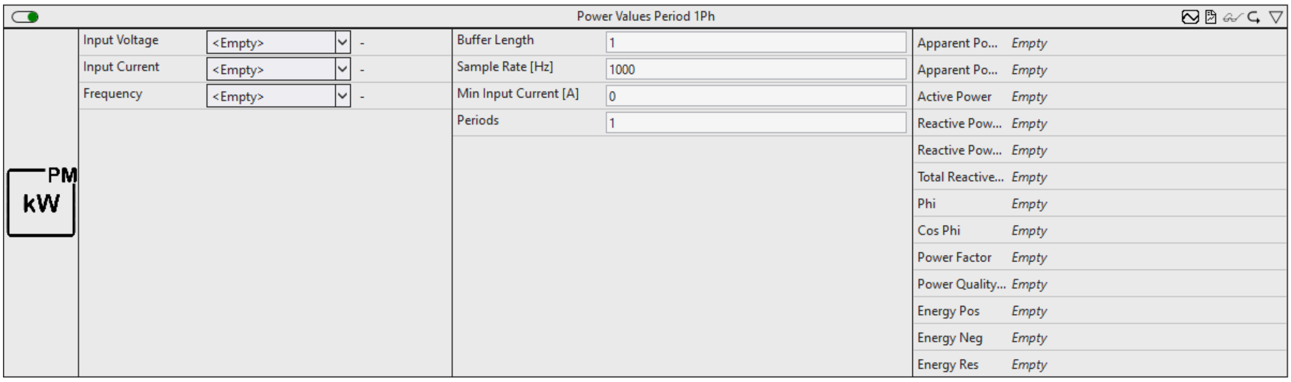
Input values

- **Frequency:** Current frequency of the input signal. Is used to determine the length of the period at the beginning of a period. The output of the *Frequency Period1 Ch* module can be used.
- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Mean U:** Mean voltage value over n periods
- **RMS U:** RMS value of the voltage over n periods.
- **Peak U:** Peak value of the voltage over n periods.
- **Rectified U:** Rectified voltage value over n periods.
- **Crest Factor U:** Crest factor of the voltage (peak value/RMS value).
- **Form Factor U:** Form factor of the voltage (RMS value/rectified value).
- **Mean I:** Mean value of the current over n periods
- **RMS I:** RMS value of the current over n periods.
- **Peak I:** Peak value of the current over n periods.
- **Rectified I:** Rectified value of the current over n periods.
- **Crest Factor I:** Crest factor of the current (peak value/RMS value).
- **Form Factor I:** Form factor of the current (RMS value/rectified value).

6.4.14.2.3.2.1.3 Power Values Period 1Ph



Input Voltage <Empty> -
Input Current <Empty> -
Frequency <Empty> -

Buffer Length 1
Sample Rate [Hz] 1000
Min Input Current [A] 0
Periods 1

Apparent Po... Empty
Apparent Po... Empty
Active Power Empty
Reactive Pow... Empty
Reactive Pow... Empty
Total Reactive... Empty
Phi Empty
Cos Phi Empty
Power Factor Empty
Power Quality... Empty
Energy Pos Empty
Energy Neg Empty
Energy Res Empty

The *Power Values Period* module calculates the power values of the connected loads. These include the fundamental components and the phase shift angle. For this purpose, only the first harmonics of the input signal are used for the calculation in addition to the signal curve in the time range. The advantage of these algorithms is the high dynamics of the calculations. The results refer to a configurable number of signal periods. The period duration refers to the frequency specified at the input at the start of the period. The statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_PowerValues_Period_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).

Input values

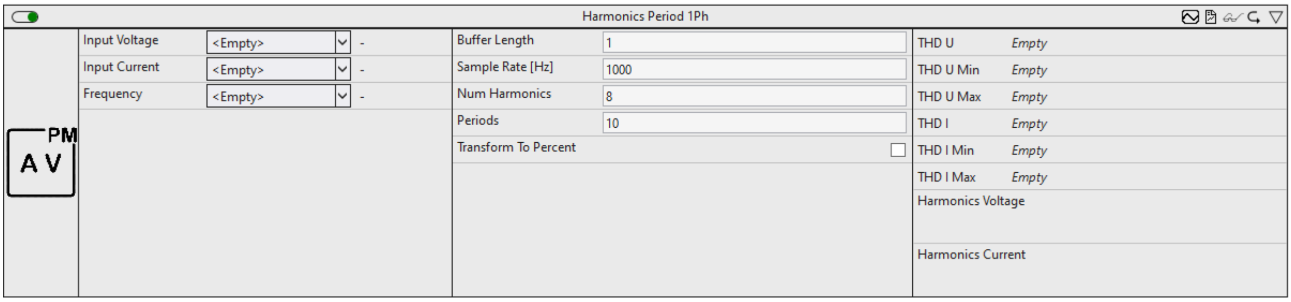
- **Frequency:** Current frequency of the input signal. Is used to determine the length of the period at the beginning of a period. The output of the *Frequency Period1 Ch* module can be used.
- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Apparent Power:** Total apparent power.
- **Apparent Power 1:** Fundamental apparent power.
- **Active Power:** Active power.
- **Reactive Power d:** Distortion reactive power.
- **Reactive Power 1:** Fundamental shift reactive power.
- **Total Reactive Power:** Total reactive power.
- **Phi:** Phase shift angle.
- **Cos Phi:** CosPhi (active power/fundamental apparent power).
- **Power Factor:** Power factor (active power/total apparent power).
- **Power Quality Factor:** Represents the quality of the power supply simplified in a value range between 0 and 1. The frequency, the RMS value of the voltage and the THD of the voltage are also included.
- **Energy Pos:** Energy in positive direction.

- **Energy Neg:** Energy in the negative direction.
- **Energy Res:** Resulting energy.

6.4.14.2.3.2.1.4 Harmonics Period 1Ph



Harmonics Period 1Ph

Input Voltage	<Empty>	-
Input Current	<Empty>	-
Frequency	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Num Harmonics	8
Periods	10
Transform To Percent	□

THD U	*Empty*
THD U Min	*Empty*
THD U Max	*Empty*
THD I	*Empty*
THD I Min	*Empty*
THD I Max	*Empty*
Harmonics Voltage	
Harmonics Current	

The *Harmonics Period* module calculates the harmonics of current and voltage. In addition, the THD of the input values is calculated from the harmonics. In contrast to the module *Harmonics*, the results refer to a configurable number of signal periods. The period duration refers to the frequency specified at the input at the start of the period. The statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation for the corresponding PLC function block FB_PMA_Harmonics_Period_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).
- **Num Harmonics:** Number of harmonics to be calculated.
- **Transform To Percent:** Boolean value that specifies whether the result should be output as a percentage. The first harmonic corresponds to 100%.

Input values

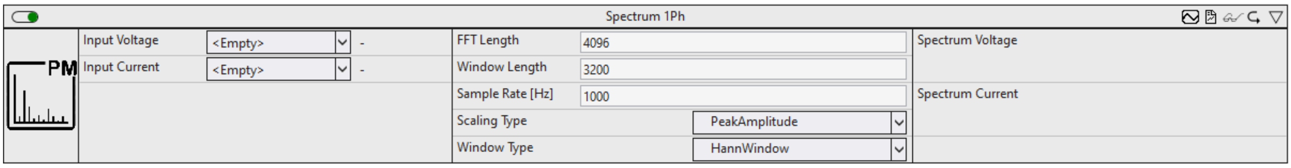
- **Frequency:** Current frequency of the input signal. Is used to determine the length of the period at the beginning of a period. The output of the *Frequency Period1 Ch* module can be used.
- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **THD U:** THD of the voltage. The output is in percent.
- **THD U Min:** Minimum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD U Max:** Maximum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD I:** THD of the current. The output is in percent.
- **THD I Min:** Minimum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD I Max:** Maximum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **Harmonics Voltage:** Harmonics of the voltage.
- **Harmonics Current:** Harmonics of the current.

6.4.14.2.3.2.2 *Based on the frequency range*

6.4.14.2.3.2.2.1 Spectrum 1Ph


Spectrum 1Ph

PM

Input Voltage <Empty> -

Input Current <Empty> -

FFT Length 4096

Window Length 3200

Sample Rate [Hz] 1000

Scaling Type PeakAmplitude

Window Type HannWindow

Spectrum Voltage

Spectrum Current

The *Spectrum* module calculates the magnitude spectra of the current and voltage values. These are suitable for analyzing the input signals in the frequency range. The size of the input buffer is half the window length.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Spectrum_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Scaling Type:** Enables selection of the scaling used if absolute scaling is required. See: E_PMA_ScalingType
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Transform To Decibel (optional):** Boolean value indicating whether the result of the FFT should be transformed to the decibel scale, corresponding to the transformation x → 20 * log10(x).
- **Decibel Threshold (optional):** Very small floating-point value greater than zero. Values smaller than this number are replaced with this value before any transformation to the decibel scale, since the logarithm of zero is not defined mathematically. The smallest possible value is 3.75e-324.

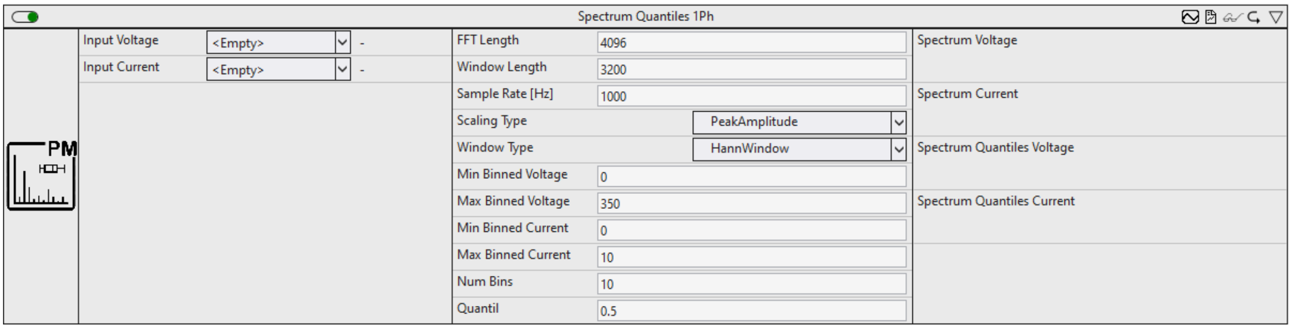
Input values

- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Spectrum Voltage:** Magnitude spectrum of the voltage values.
- **Spectrum Current:** Magnitude spectrum of the current values.

6.4.14.2.3.2.2.2 Spectrum Quantiles 1Ph


Spectrum Quantiles 1Ph

PM

Input Voltage <Empty> -

Input Current <Empty> -

FFT Length 4096

Window Length 3200

Sample Rate [Hz] 1000

Scaling Type PeakAmplitude

Window Type HannWindow

Min Binned Voltage 0

Max Binned Voltage 350

Min Binned Current 0

Max Binned Current 10

Num Bins 10

Quantil 0.5

Spectrum Voltage

Spectrum Current

Spectrum Quantiles Voltage

Spectrum Quantiles Current

The *Spectrum Quantiles* module calculates the magnitude spectra of the current and voltage values in the same way as the *Spectrum* module. In addition, p-quantiles of the spectrum distribution can be calculated.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Spectrum_Quantiles_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Scaling Type:** Enables selection of the scaling used if absolute scaling is required. See: E_PMA_ScalingType
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Transform To Decibel (optional):** Boolean value indicating whether the result of the FFT should be transformed to the decibel scale, corresponding to the transformation x → 20 * log10(x).
- **Decibel Threshold (optional):** Very small floating-point value greater than zero. Values smaller than this number are replaced with this value before any transformation to the decibel scale, since the logarithm of zero is not defined mathematically. The smallest possible value is 3.75e-324.
- **Min Binned Voltage:** Lower limit value for which the output data of the spectrum calculation of the voltage are counted in the regular histogram bins.
- **Max Binned Voltage:** Upper limit value for which the output data of the spectrum calculation of the voltage are counted in the regular histogram bins. *Max Binned Voltage* must be greater than *Min Binned Voltage*.
- **Min Binned Current:** Lower limit value for which the output data of the spectrum calculation of the current are counted in the regular histogram bins.
- **Max Binned Current:** Upper limit value for which the output data of the spectrum calculation of the current are counted in the regular histogram bins. *Max Binned Current* must be greater than *Min Binned Current*.
- **Num Bins:** Number of bins in a histogram. The minimum number is 1. In many cases, values between ten and twenty are a sensible choice. The two special bins for values below *Min Binned* and above *Max Binned* are not included in this value.
- **Quantile:** Indicates the quantile limit. It must be between 0.0 and 1.0. For example, 0.2 corresponds to the 20% quantile.

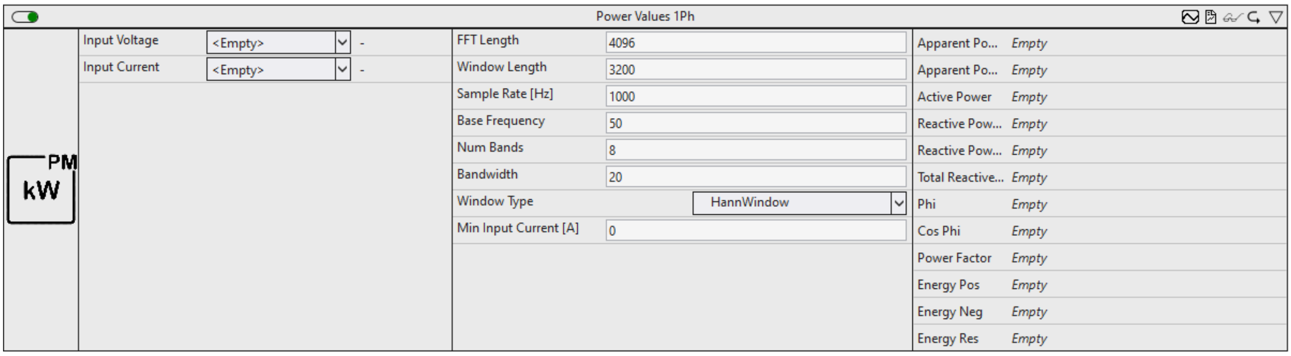
Input values

- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Spectrum Voltage:** Magnitude spectrum of the voltage values.
- **Spectrum Current:** Magnitude spectrum of the current values.
- **Spectrum Quantiles Voltage:** P-quantiles of the distribution of *Spectrum Voltage*
- **Spectrum Quantiles Current:** P-Quantiles of the distribution of *Spectrum Current*

6.4.14.2.3.2.2.3 Power Values 1Ph



Power Values 1Ph

Input Voltage	<Empty>	-
Input Current	<Empty>	-

FFT Length	4096
Window Length	3200
Sample Rate [Hz]	1000
Base Frequency	50
Num Bands	8
Bandwidth	20
Window Type	HannWindow
Min Input Current [A]	0

Apparent Po...	*Empty*
Apparent Po...	*Empty*
Active Power	*Empty*
Reactive Pow...	*Empty*
Reactive Pow...	*Empty*
Total Reactive...	*Empty*
Phi	*Empty*
Cos Phi	*Empty*
Power Factor	*Empty*
Energy Pos	*Empty*
Energy Neg	*Empty*
Energy Res	*Empty*

The *Power Values* module calculates the power values of the connected loads. These include the fundamental components and the phase shift angle. Internally, the individual harmonics and their phase angle are determined for the calculations.

Alternatively, the *Power Values Period* module can be used for the calculation. It uses simpler calculation methods for enhanced dynamics.

In the Beckhoff Information System you will find the documentation for the corresponding PLC function block FB_PMA_PowerValues_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Base Frequency**: Frequency of the first harmonic in Hz.
- **Num Bands**: Specifies the number of bands for which the RMS is calculated.
- **Bandwidth**: Total bandwidth of a single RMS band.
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.

Input values

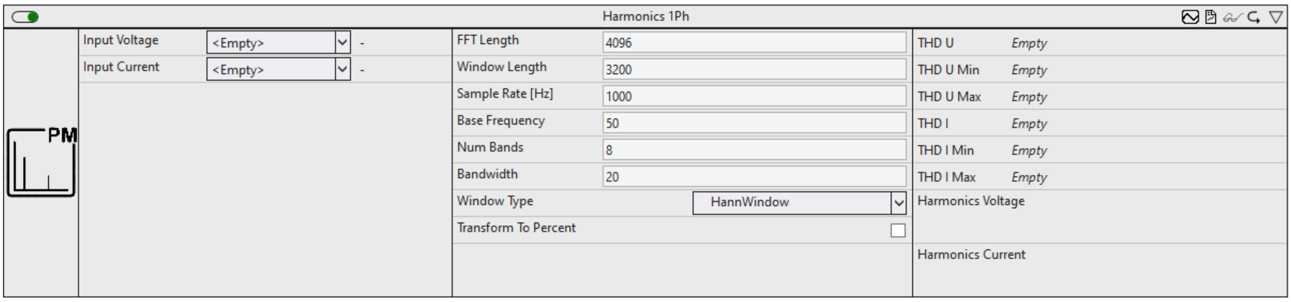
- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Apparent Power:** Total apparent power.
- **Apparent Power 1:** Fundamental apparent power.
- **Active Power:** Active power.
- **Reactive Power d:** Distortion reactive power.
- **Reactive Power 1:** Fundamental shift reactive power.
- **Total Reactive Power:** Total reactive power.
- **Phi:** Phase shift angle.
- **Cos Phi:** CosPhi (active power/fundamental apparent power).

- **Power Factor:** Power factor (active power/total apparent power).
- **Power Quality Factor:** Represents the quality of the power supply simplified in a value range between 0 and 1. The frequency, the RMS value of the voltage and the THD of the voltage are also included.
- **Energy Pos:** Energy in positive direction.
- **Energy Neg:** Energy in the negative direction.
- **Energy Res:** Resulting energy.

6.4.14.2.3.2.2.4 Harmonics 1Ph



Harmonics 1Ph

Input Voltage	<Empty>	-	
Input Current	<Empty>	-	
FFT Length	4096	THD U	Empty
Window Length	3200	THD U Min	Empty
Sample Rate [Hz]	1000	THD U Max	Empty
Base Frequency	50	THD I	Empty
Num Bands	8	THD I Min	Empty
Bandwidth	20	THD I Max	Empty
Window Type	HannWindow	Harmonics Voltage	
Transform To Percent	□		
		Harmonics Current	

The *Harmonics* module calculates the harmonics of current and voltage. In addition, the THD of the input values is calculated from the harmonics.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Harmonics_1Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Num Harmonics:** Number of harmonics to be calculated.
- **Transform To Percent:** Boolean value that specifies whether the result should be output as a percentage. The first harmonic corresponds to 100%.
- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Base Frequency**: Frequency of the first harmonic in Hz.
- **Num Bands**: Specifies the number of bands for which the RMS is calculated.
- **Bandwidth**: Total bandwidth of a single RMS band.
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Transform To Percent:** Boolean value that specifies whether the result should be output as a percentage. The first harmonic corresponds to 100%.

Input values

- **Input Voltage:** Input values of the voltage.
- **Input Current:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

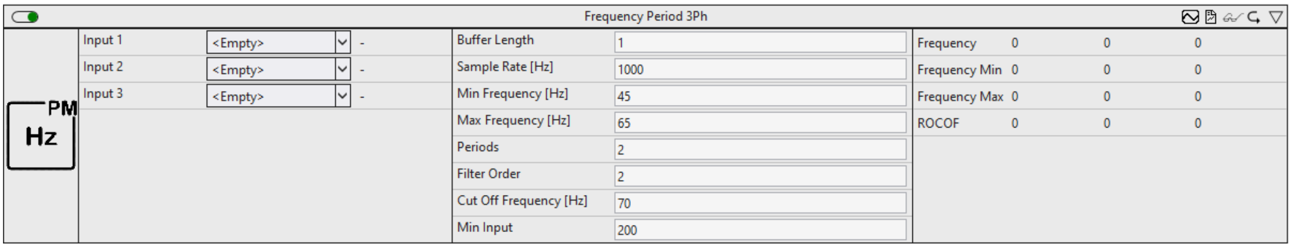
- **THD U:** THD of the voltage. The output is in percent.
- **THD U Min:** Minimum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD U Max:** Maximum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.

- **THD I:** THD of the current. The output is in percent.
- **THD I Min:** Minimum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD I Max:** Maximum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **Harmonics Voltage:** Harmonics of the voltage.
- **Harmonics Current:** Harmonics of the current.

6.4.14.2.3.3 Three-phase

6.4.14.2.3.3.1 Based on the signal period

6.4.14.2.3.3.1.1 Frequency Period 3Ph



Frequency Period 3Ph

Input 1	<Empty>	-
Input 2	<Empty>	-
Input 3	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Min Frequency [Hz]	45
Max Frequency [Hz]	65
Periods	2
Filter Order	2
Cut Off Frequency [Hz]	70
Min Input	200

Frequency	0	0	0
Frequency Min	0	0	0
Frequency Max	0	0	0
ROCOF	0	0	0

The *Frequency Period* module calculates the base frequency of the input signal. To do this, the signal is first filtered with a Butterworth low-pass filter. The zero crossings of the input signal are then determined from the filtered values, and the frequency is calculated from their difference. The results refer to one or more periods, depending on the configuration. The statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Frequency_Period_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Frequency [Hz]**: Minimum measurement frequency to be expected.
- **Max Frequency [Hz]:** Maximum measurement frequency to be expected.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).
- **Filter Order:** Order of the low-pass filter. The stability of the filter must be considered for the setting. Only values up to the tenth order are allowed.
- **Cut Off Frequency [Hz]:** Cut-off frequency of the low-pass filter.
- **Min Input:** Minimum input value (RMS) over one period. This parameter prevents the calculation of input values that are too small.
- **ROCOF Avg Window Order:** Window size of the sliding average value for the calculation of the ROCOF.

Input values

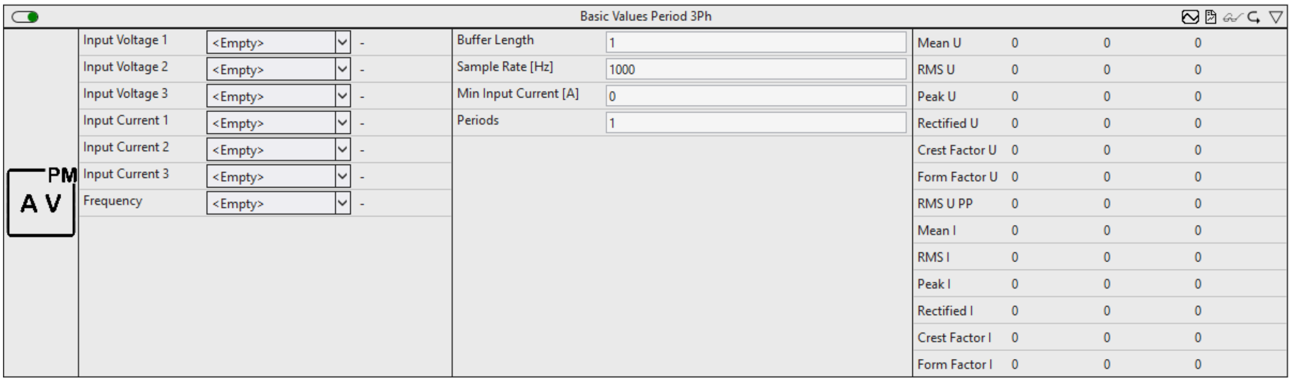
- **Input 1 .. Input 3:** Input values of current or voltage.
- **Reset Statistics (optional):** *TRUE* resets the min and max values of the outputs.

Output values

- **Frequency:** Frequency determined by two or more zero crossings.
- **Frequency Min:** Smallest value of the calculated frequency. Can be reset via the *Reset Statistics* input.

- **Frequency Max:** Largest value of the calculated frequency. Can be reset via the *Reset Statistics* input.
- **ROCOF:** Rate of change of frequency (ROCOF).

6.4.14.2.3.3.1.2 Basic Values Period 3Ph



Basic Values Period 3Ph

Input Voltage 1	<Empty>	-
Input Voltage 2	<Empty>	-
Input Voltage 3	<Empty>	-
Input Current 1	<Empty>	-
Input Current 2	<Empty>	-
Input Current 3	<Empty>	-
Frequency	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Min Input Current [A]	0
Periods	1

Mean U	0	0	0
RMS U	0	0	0
Peak U	0	0	0
Rectified U	0	0	0
Crest Factor U	0	0	0
Form Factor U	0	0	0
RMS U PP	0	0	0
Mean I	0	0	0
RMS I	0	0	0
Peak I	0	0	0
Rectified I	0	0	0
Crest Factor I	0	0	0
Form Factor I	0	0	0

The *Basic Values Period* module calculates analysis values for the signal curve of current and voltage over time. These include the mean value, the RMS value, the peak value, the rectified value, the crest factor and the form factor for current and voltage. The results refer to a configurable number of signal periods. The period duration refers to the frequency specified at the input at the start of the period. The optional statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_BasicValues_Period_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).

Input values

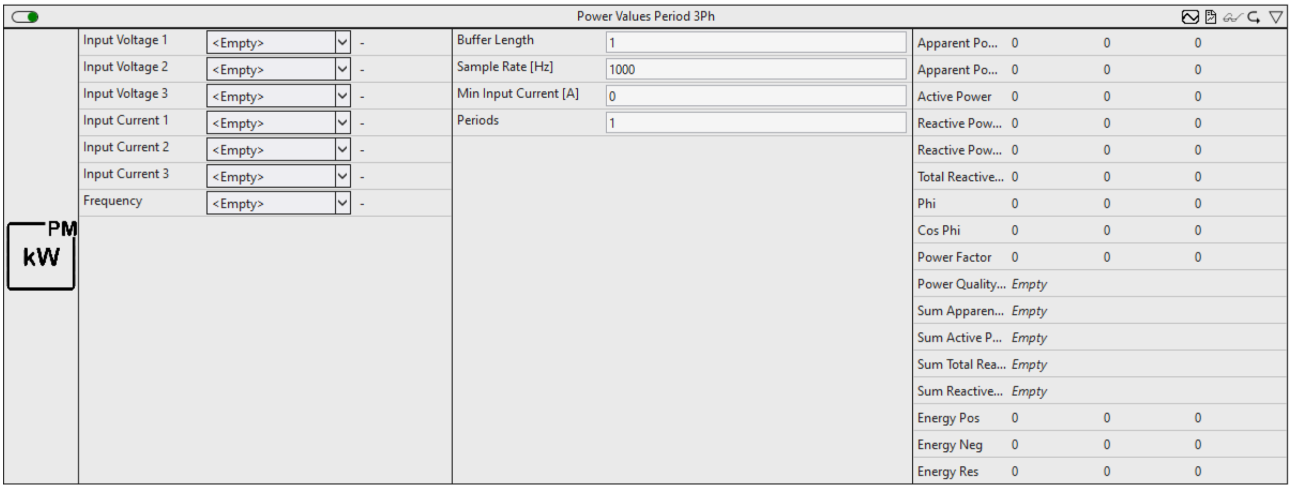
- **Frequency:** Current frequency of the input signal. Is used to determine the length of the period at the beginning of a period. The output of the *Frequency Period1 Ch* module can be used.
- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Mean U:** Mean voltage value over n periods
- **RMS U:** RMS value of the voltage over n periods.
- **Peak U:** Peak value of the voltage over n periods.
- **Rectified U:** Rectified voltage value over n periods.
- **Crest Factor U:** Crest factor of the voltage (peak value/RMS value).
- **Form Factor U:** Form factor of the voltage (RMS value/rectified value).
- **Mean I:** Mean value of the current over n periods
- **RMS I:** RMS value of the current over n periods.
- **Peak I:** Peak value of the current over n periods.
- **Rectified I:** Rectified value of the current over n periods.

- **Crest Factor I:** Crest factor of the current (peak value/RMS value).
- **Form Factor I:** Form factor of the current (RMS value/rectified value).

6.4.14.2.3.3.1.3 Power Values Period 3Ph



Power Values Period 3Ph

Input Voltage 1	<Empty>	-
Input Voltage 2	<Empty>	-
Input Voltage 3	<Empty>	-
Input Current 1	<Empty>	-
Input Current 2	<Empty>	-
Input Current 3	<Empty>	-
Frequency	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Min Input Current [A]	0
Periods	1

Apparent Po...	0	0	0
Apparent Po...	0	0	0
Active Power	0	0	0
Reactive Pow...	0	0	0
Reactive Pow...	0	0	0
Total Reactive...	0	0	0
Phi	0	0	0
Cos Phi	0	0	0
Power Factor	0	0	0
Power Quality...	*Empty*		
Sum Apparen...	*Empty*		
Sum Active P...	*Empty*		
Sum Total Rea...	*Empty*		
Sum Reactive...	*Empty*		
Energy Pos	0	0	0
Energy Neg	0	0	0
Energy Res	0	0	0

The *Power Values Period* module calculates the power values of the connected loads. These include the fundamental components and the phase shift angle. For this purpose, only the first harmonics of the input signal are used for the calculation in addition to the signal curve in the time range. The advantage of these algorithms is the high dynamics of the calculations. The results refer to a configurable number of signal periods. The period duration refers to the frequency specified at the input at the start of the period. The statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_PowerValues_Period_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).

Input values

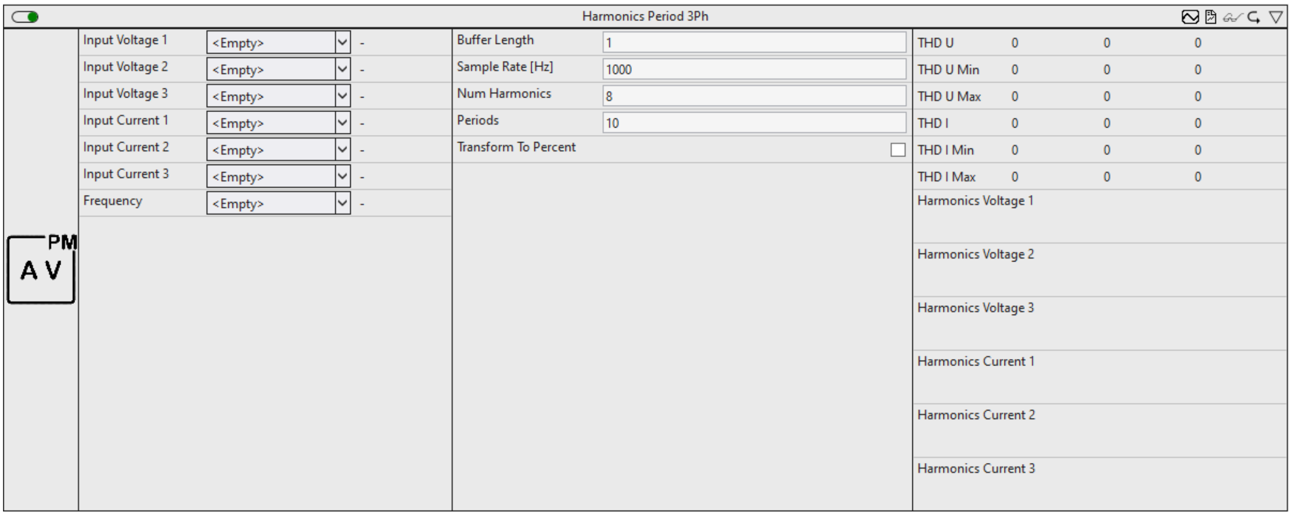
- **Frequency:** Current frequency of the input signal. Is used to determine the length of the period at the beginning of a period. The output of the *Frequency Period1 Ch* module can be used.
- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Apparent Power:** Total apparent power.
- **Apparent Power 1:** Fundamental apparent power.
- **Active Power:** Active power.
- **Reactive Power d:** Distortion reactive power.
- **Reactive Power 1:** Fundamental shift reactive power.
- **Total Reactive Power:** Total reactive power.
- **Phi:** Phase shift angle.

- **Cos Phi:** CosPhi (active power/fundamental apparent power).
- **Power Factor:** Power factor (active power/total apparent power).
- **Power Quality Factor:** Represents the quality of the power supply simplified in a value range between 0 and 1. The frequency, the RMS value of the voltage and the THD of the voltage are also included.
- **Energy Pos:** Energy in positive direction.
- **Energy Neg:** Energy in the negative direction.
- **Energy Res:** Resulting energy.
- **Sum Apparent Power:** Sum of the total apparent power of all phases.
- **Sum Active Power:** Sum of the active power of all phases.
- **Sum Total Reactive Power:** Sum of the total reactive power of all phases.
- **Sum Reactive Power 1 Res:** Sum of the values of the fundamental shift reactive power of all phases.

6.4.14.2.3.3.1.4 Harmonics Period 3Ph



Harmonics Period 3Ph

Input Voltage 1	<Empty>	-
Input Voltage 2	<Empty>	-
Input Voltage 3	<Empty>	-
Input Current 1	<Empty>	-
Input Current 2	<Empty>	-
Input Current 3	<Empty>	-
Frequency	<Empty>	-

Buffer Length	1
Sample Rate [Hz]	1000
Num Harmonics	8
Periods	10
Transform To Percent	□

THD U	0	0	0
THD U Min	0	0	0
THD U Max	0	0	0
THD I	0	0	0
THD I Min	0	0	0
THD I Max	0	0	0
Harmonics Voltage 1			
Harmonics Voltage 2			
Harmonics Voltage 3			
Harmonics Current 1			
Harmonics Current 2			
Harmonics Current 3			

The *Harmonics Period* module calculates the harmonics of current and voltage. In addition, the THD of the input values is calculated from the harmonics. In contrast to the module *Harmonics*, the results refer to a configurable number of signal periods. The period duration refers to the frequency specified at the input at the start of the period. The statistical results refer to the entire runtime or the time at which the statistical results were last reset.

In the Beckhoff Information System you will find the documentation for the corresponding PLC function block FB_PMA_Harmonics_Period_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Buffer Length:** Length of the input buffer.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.
- **Periods:** Number of periods that influence the calculation. (Period length = sample rate/frequency).
- **Num Harmonics:** Number of harmonics to be calculated.
- **Transform To Percent:** Boolean value that specifies whether the result should be output as a percentage. The first harmonic corresponds to 100%.

Input values

- **Frequency:** Current frequency of the input signal. Is used to determine the length of the period at the beginning of a period. The output of the *Frequency Period1 Ch* module can be used.

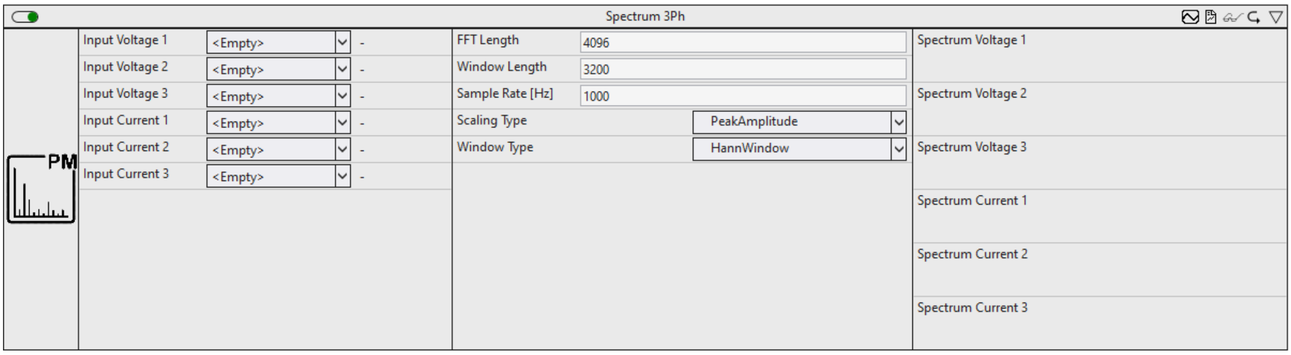
- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **THD U:** THD of the voltage. The output is in percent.
- **THD U Min:** Minimum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD U Max:** Maximum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD I:** THD of the current. The output is in percent.
- **THD I Min:** Minimum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD I Max:** Maximum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **Harmonics Voltage:** Harmonics of the voltage.
- **Harmonics Current:** Harmonics of the current.

6.4.14.2.3.3.2 Based on the frequency range

6.4.14.2.3.3.2.1 Spectrum 3Ph



Spectrum 3Ph

Input Voltage 1 <Empty> -
Input Voltage 2 <Empty> -
Input Voltage 3 <Empty> -
Input Current 1 <Empty> -
Input Current 2 <Empty> -
Input Current 3 <Empty> -

FFT Length 4096
Window Length 3200
Sample Rate [Hz] 1000
Scaling Type PeakAmplitude
Window Type HannWindow

Spectrum Voltage 1
Spectrum Voltage 2
Spectrum Voltage 3
Spectrum Current 1
Spectrum Current 2
Spectrum Current 3

The *Spectrum* module calculates the magnitude spectra of the current and voltage values. These are suitable for analyzing the input signals in the frequency range. The size of the input buffer is half the window length.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Spectrum_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Scaling Type:** Enables selection of the scaling used if absolute scaling is required. See: E_PMA_ScalingType
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Transform To Decibel (optional):** Boolean value indicating whether the result of the FFT should be transformed to the decibel scale, corresponding to the transformation x → 20 * log10(x).
- **Decibel Threshold (optional):** Very small floating-point value greater than zero. Values smaller than this number are replaced with this value before any transformation to the decibel scale, since the logarithm of zero is not defined mathematically. The smallest possible value is 3.75e-324.

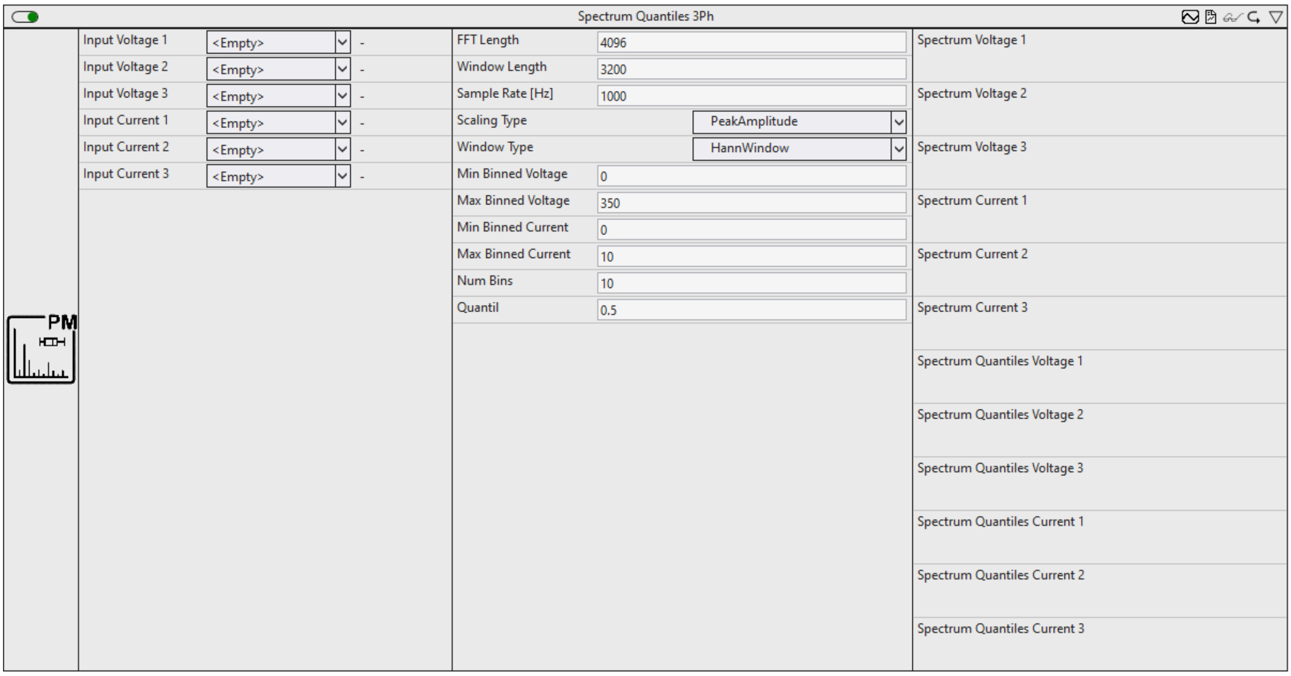
Input values

- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Spectrum Voltage 1 .. Spectrum Voltage 3:** Magnitude spectrum of the voltage values.
- **Spectrum Current 1 ... Spectrum Current 3:** Magnitude spectrum of the current values.

6.4.14.2.3.3.2.2 Spectrum Quantiles 3Ph



Spectrum Quantiles 3Ph

Input Voltage 1	<Empty>	-
Input Voltage 2	<Empty>	-
Input Voltage 3	<Empty>	-
Input Current 1	<Empty>	-
Input Current 2	<Empty>	-
Input Current 3	<Empty>	-

FFT Length	4096
Window Length	3200
Sample Rate [Hz]	1000
Scaling Type	PeakAmplitude
Window Type	HannWindow
Min Binned Voltage	0
Max Binned Voltage	350
Min Binned Current	0
Max Binned Current	10
Num Bins	10
Quantil	0.5

PM
Spectrum Voltage 1
Spectrum Voltage 2
Spectrum Voltage 3
Spectrum Current 1
Spectrum Current 2
Spectrum Current 3
Spectrum Quantiles Voltage 1
Spectrum Quantiles Voltage 2
Spectrum Quantiles Voltage 3
Spectrum Quantiles Current 1
Spectrum Quantiles Current 2
Spectrum Quantiles Current 3

The *Spectrum Quantiles* module calculates the magnitude spectra of the current and voltage values in the same way as the *Spectrum* module. In addition, p-quantiles of the spectrum distribution can be calculated.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Spectrum_Quantiles_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Scaling Type:** Enables selection of the scaling used if absolute scaling is required. See: E_PMA_ScalingType
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Transform To Decibel (optional):** Boolean value indicating whether the result of the FFT should be transformed to the decibel scale, corresponding to the transformation x → 20 * log10(x).
- **Decibel Threshold (optional):** Very small floating-point value greater than zero. Values smaller than this number are replaced with this value before any transformation to the decibel scale, since the logarithm of zero is not defined mathematically. The smallest possible value is 3.75e-324.

- **Min Binned Voltage:** Lower limit value for which the output data of the spectrum calculation of the voltage are counted in the regular histogram bins.
- **Max Binned Voltage:** Upper limit value for which the output data of the spectrum calculation of the voltage are counted in the regular histogram bins. *Max Binned Voltage* must be greater than *Min Binned Voltage*.
- **Min Binned Current:** Lower limit value for which the output data of the spectrum calculation of the current are counted in the regular histogram bins.
- **Max Binned Current:** Upper limit value for which the output data of the spectrum calculation of the current are counted in the regular histogram bins. *Max Binned Current* must be greater than *Min Binned Current*.
- **Num Bins:** Number of bins in a histogram. The minimum number is 1. In many cases, values between ten and twenty are a sensible choice. The two special bins for values below *Min Binned* and above *Max Binned* are not included in this value.
- **Quantile:** Indicates the quantile limit. It must be between 0.0 and 1.0. For example, 0.2 corresponds to the 20% quantile.

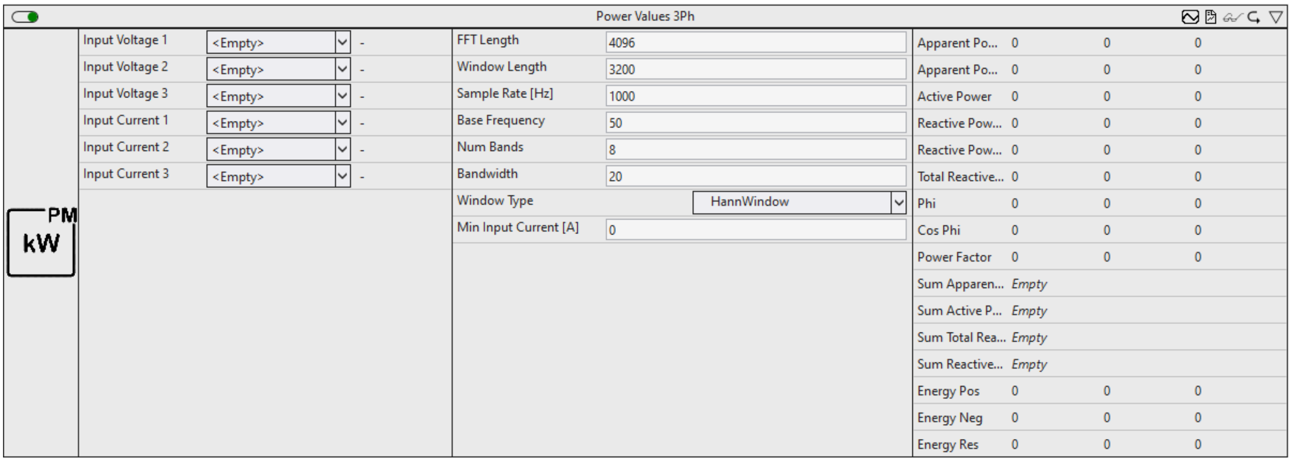
Input values

- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Spectrum Voltage 1 .. Spectrum Voltage 3:** Magnitude spectrum of the voltage values.
- **Spectrum Current 1 ... Spectrum Current 3:** Magnitude spectrum of the current values
- **Spectrum Quantiles Voltage 1 ... Spectrum Quantiles Voltage 3:** P-quantiles of the distribution of *Spectrum Voltage*
- **Spectrum Quantiles Current 1 ... Spectrum Quantiles Current 3:** P-Quantiles of the distribution of *Spectrum Current*

6.4.14.2.3.3.2.3 Power Values 3Ph



Input Voltage 1	<Empty>	-
Input Voltage 2	<Empty>	-
Input Voltage 3	<Empty>	-
Input Current 1	<Empty>	-
Input Current 2	<Empty>	-
Input Current 3	<Empty>	-

FFT Length	4096
Window Length	3200
Sample Rate [Hz]	1000
Base Frequency	50
Num Bands	8
Bandwidth	20
Window Type	HannWindow
Min Input Current [A]	0

Apparent Po...	0	0	0
Apparent Po...	0	0	0
Active Power	0	0	0
Reactive Pow...	0	0	0
Reactive Pow...	0	0	0
Total Reactive...	0	0	0
Phi	0	0	0
Cos Phi	0	0	0
Power Factor	0	0	0
Sum Apparen...	*Empty*		
Sum Active P...	*Empty*		
Sum Total Rea...	*Empty*		
Sum Reactive...	*Empty*		
Energy Pos	0	0	0
Energy Neg	0	0	0
Energy Res	0	0	0

The *Power Values* module calculates the power values of the connected loads. These include the fundamental components and the phase shift angle. Internally, the individual harmonics and their phase angle are determined for the calculations.

Alternatively, the *Power Values Period* module can be used for the calculation. It uses simpler calculation methods for enhanced dynamics.

In the Beckhoff Information System you will find the documentation for the corresponding PLC function block FB_PMA_PowerValues_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Base Frequency**: Frequency of the first harmonic in Hz.
- **Num Bands**: Specifies the number of bands for which the RMS is calculated.
- **Bandwidth**: Total bandwidth of a single RMS band.
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Min Input Current [A]:** Minimum input value (RMS) of the current. This parameter prevents the calculation of input values that are too small.

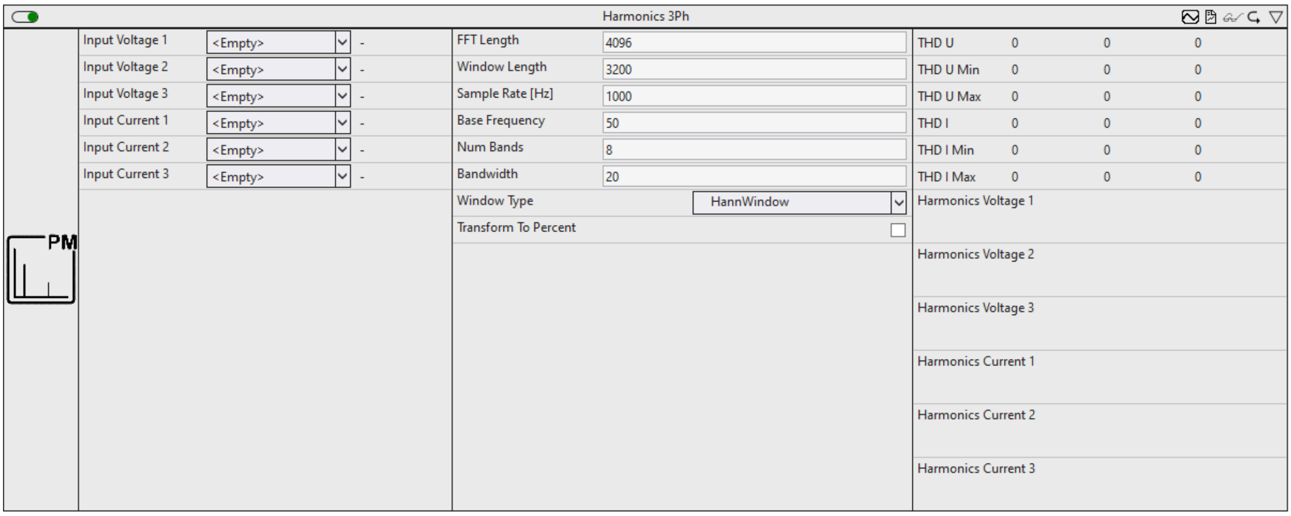
Input values

- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **Apparent Power:** Total apparent power.
- **Apparent Power 1:** Fundamental apparent power.
- **Active Power:** Active power.
- **Reactive Power d:** Distortion reactive power.
- **Reactive Power 1:** Fundamental shift reactive power.
- **Total Reactive Power:** Total reactive power.
- **Phi:** Phase shift angle.
- **Cos Phi:** CosPhi (active power/fundamental apparent power).
- **Power Factor:** Power factor (active power/total apparent power).
- **Power Quality Factor:** Represents the quality of the power supply simplified in a value range between 0 and 1. The frequency, the RMS value of the voltage and the THD of the voltage are also included.
- **Energy Pos:** Energy in positive direction.
- **Energy Neg:** Energy in the negative direction.
- **Energy Res:** Resulting energy.
- **Sum Apparent Power:** Sum of the total apparent power of all phases.
- **Sum Active Power:** Sum of the active power of all phases.
- **Sum Total Reactive Power:** Sum of the total reactive power of all phases.
- **Sum Reactive Power 1 Res:** Sum of the values of the fundamental shift reactive power of all phases.

6.4.14.2.3.3.2.4 Harmonics 3Ph



Harmonics 3Ph

Input Voltage 1	<Empty>	-
Input Voltage 2	<Empty>	-
Input Voltage 3	<Empty>	-
Input Current 1	<Empty>	-
Input Current 2	<Empty>	-
Input Current 3	<Empty>	-

FFT Length	4096
Window Length	3200
Sample Rate [Hz]	1000
Base Frequency	50
Num Bands	8
Bandwidth	20
Window Type	HannWindow
Transform To Percent	□

THD U	0	0	0
THD U Min	0	0	0
THD U Max	0	0	0
THD I	0	0	0
THD I Min	0	0	0
THD I Max	0	0	0
Harmonics Voltage 1			
Harmonics Voltage 2			
Harmonics Voltage 3			
Harmonics Current 1			
Harmonics Current 2			
Harmonics Current 3			

The *Harmonics* module calculates the harmonics of current and voltage. In addition, the THD of the input values is calculated from the harmonics.

In the Beckhoff Information System you will find the documentation of the corresponding PLC function block FB_PMA_Harmonics_3Ph.

In the Beckhoff Information System you will find the documentation of the TF3650 TwinCAT 3 Power Monitoring function with descriptions of the corresponding PLC library.

Configuration parameters

- **Num Harmonics:** Number of harmonics to be calculated.
- **Transform To Percent:** Boolean value that specifies whether the result should be output as a percentage. The first harmonic corresponds to 100%.
- **FFT Length:** Length of the FFT. It must be greater than one and an integral power of two.
- **Window Length:** Length of the analysis window in samples. The length must be greater than one and an even number. It must not be greater than *FFT Length*.
- **Sample Rate [Hz]:** Sample rate (samples per second) of the input signal.
- **Base Frequency**: Frequency of the first harmonic in Hz.
- **Num Bands**: Specifies the number of bands for which the RMS is calculated.
- **Bandwidth**: Total bandwidth of a single RMS band.
- **Window Type:** Defines the window function used. The window type *HannWindow* is a good default value. The windowing can be switched off by the use of the window type *RectangularWindow*. See: E_PMA_WindowType
- **Transform To Percent:** Boolean value that specifies whether the result should be output as a percentage. The first harmonic corresponds to 100%.

Input values

- **Input Voltage 1 .. Input Voltage 3:** Input values of the voltage.
- **Input Current 1 .. Input Current 3:** Input values of the current.
- **Reset Statistics (optional):** `TRUE` resets the min. and max. values of the outputs.

Output values

- **THD U:** THD of the voltage. The output is in percent.
- **THD U Min:** Minimum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD U Max:** Maximum *THD U* value that has occurred. Can be reset via the *Reset Statistics* input.
- **THD I:** THD of the current. The output is in percent.
- **THD I Min:** Minimum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.

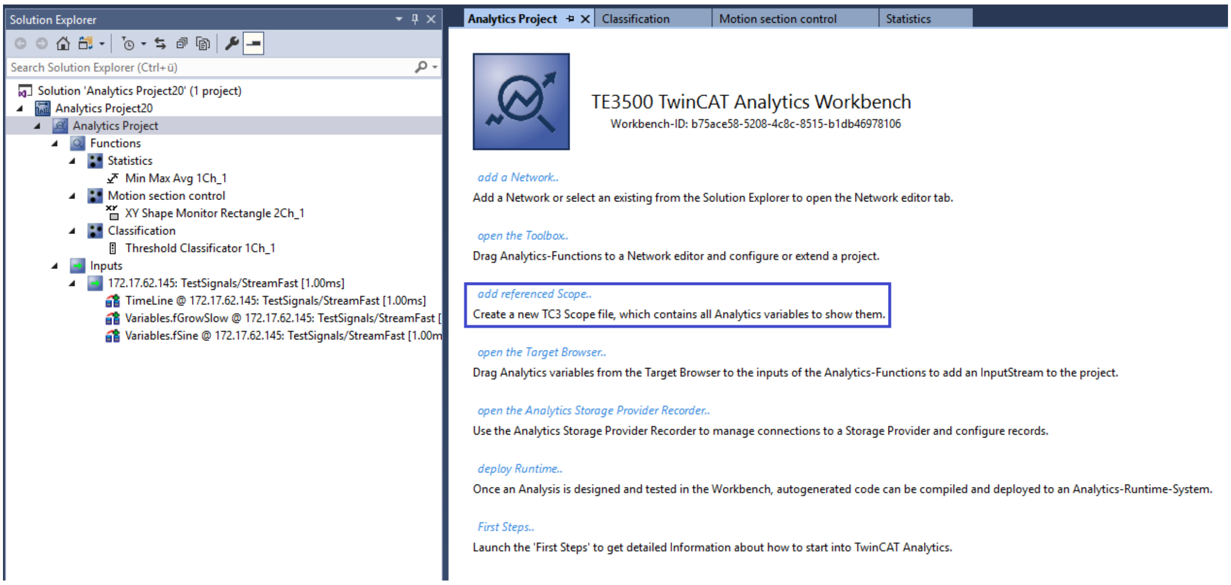
- **THD I Max:** Maximum *THD I* value that has occurred. Can be reset via the *Reset Statistics* input.
- **Harmonics Voltage:** Harmonics of the voltage.
- **Harmonics Current:** Harmonics of the current.

6.5 Interaction with Scope

The TwinCAT Analytics engineering tools offer easy interaction between the Analytics Configurator and Scope View. They allow you to highlight significant values in the data stream and examine them with other process data within the exact cycle. In addition, it is possible to display the algorithm's result data, such as an average or maximum value, in the Scope View.

✓ After configuring the analysis, switch to the **Analytics Project** start page.

1. Click **add referenced Scope…**


Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'Analytics Project20' (1 project)
Analytics Project20
Analytics Project
Functions
Statistics
Min Max Avg 1Ch_1
Motion section control
XY Shape Monitor Rectangle 2Ch_1
Classification
Threshold Classificator 1Ch_1
Inputs
172.17.62.145: TestSignals/StreamFast [1.00ms]
TimeLine @ 172.17.62.145: TestSignals/StreamFast [1.00ms]
Variables.fGrowSlow @ 172.17.62.145: TestSignals/StreamFast [
Variables.fSine @ 172.17.62.145: TestSignals/StreamFast [1.00m
Analytics Project
Classification
Motion section control
Statistics

TE3500 TwinCAT Analytics Workbench

Workbench-ID: b75ace58-5208-4c8c-8515-b1db46978106

add a Network..

Add a Network or select an existing from the Solution Explorer to open the Network editor tab.

open the Toolbox..

Drag Analytics-Functions to a Network editor and configure or extend a project.

add referenced Scope..

Create a new TC3 Scope file, which contains all Analytics variables to show them.

open the Target Browser..

Drag Analytics variables from the Target Browser to the inputs of the Analytics-Functions to add an InputStream to the project.

open the Analytics Storage Provider Recorder..

Use the Analytics Storage Provider Recorder to manage connections to a Storage Provider and configure records.

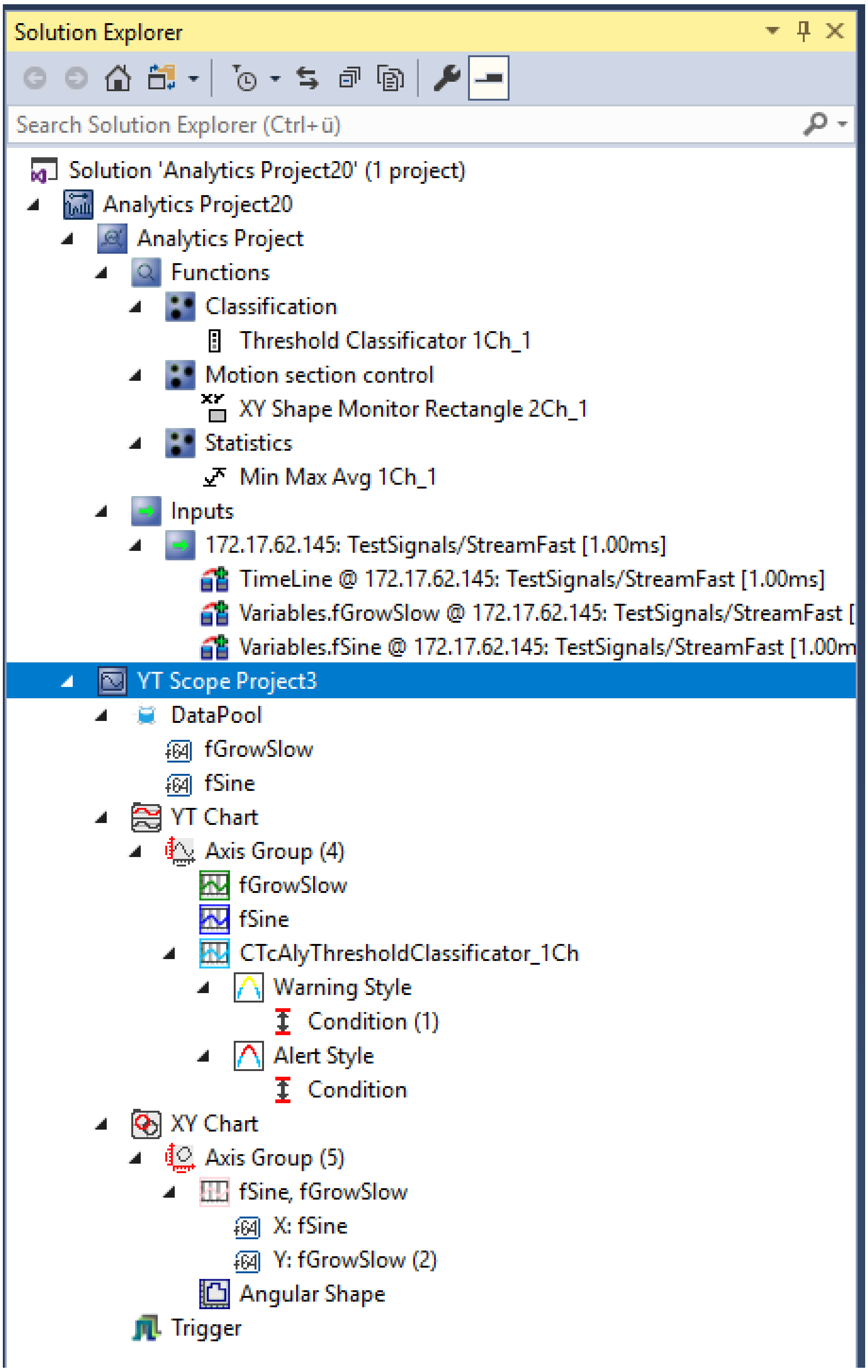
deploy Runtime..

Once an Analysis is designed and tested in the Workbench, autogenerated code can be compiled and deployed to an Analytics-Runtime-System.

First Steps..

Launch the 'First Steps' to get detailed Information about how to start into TwinCAT Analytics.

⇨ An appropriate Scope configuration is automatically added to the project.

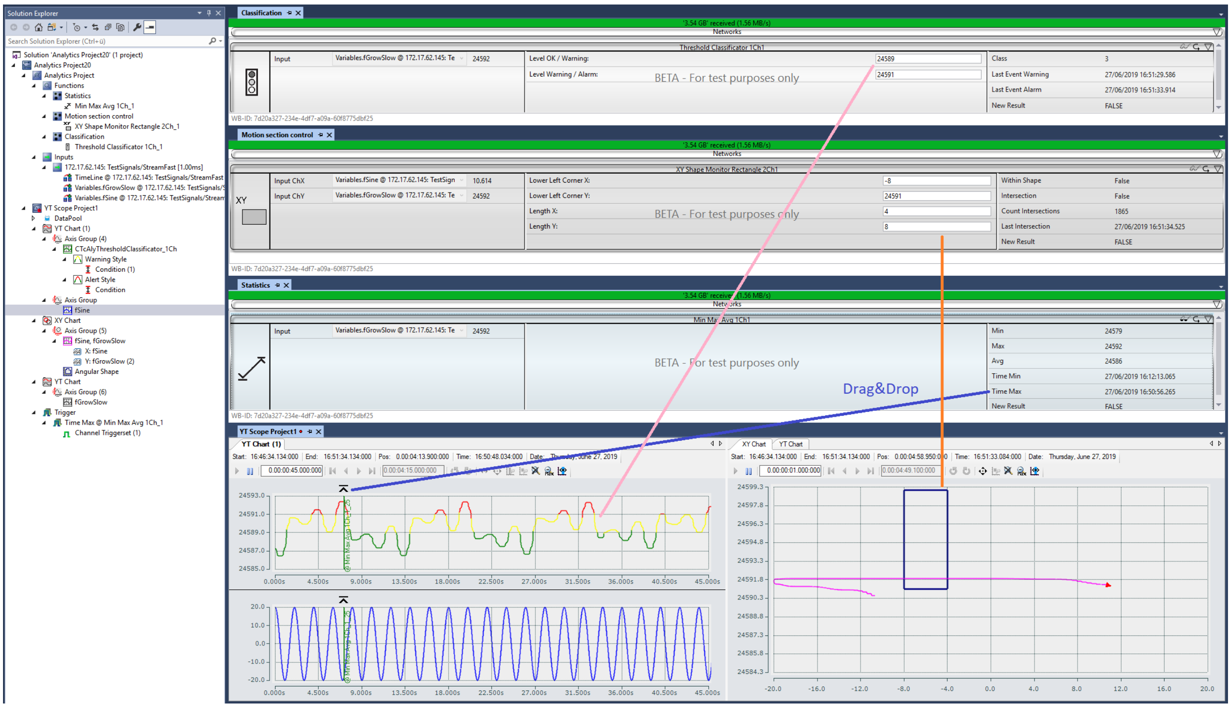


Solution Explorer

Search Solution Explorer (Ctrl+ü)

```
Solution 'Analytics Project20' (1 project)
◢ Analytics Project20
    ◢ Analytics Project
        ◢ Functions
            ◢ Classification
                Threshold Classificator 1Ch_1
            ◢ Motion section control
                XY Shape Monitor Rectangle 2Ch_1
            ◢ Statistics
                Min Max Avg 1Ch_1
        ◢ Inputs
            ◢ 172.17.62.145: TestSignals/StreamFast [1.00ms]
                TimeLine @ 172.17.62.145: TestSignals/StreamFast [1.00ms]
                Variables.fGrowSlow @ 172.17.62.145: TestSignals/StreamFast [
                Variables.fSine @ 172.17.62.145: TestSignals/StreamFast [1.00m
    ◢ YT Scope Project3
        ◢ DataPool
            fGrowSlow
            fSine
        ◢ YT Chart
            ◢ Axis Group (4)
                fGrowSlow
                fSine
                ◢ CTcAlyThresholdClassificator_1Ch
                    ◢ Warning Style
                        Condition (1)
                    ◢ Alert Style
                        Condition
        ◢ XY Chart
            ◢ Axis Group (5)
                ◢ fSine, fGrowSlow
                    X: fSine
                    Y: fGrowSlow (2)
                Angular Shape
        Trigger
```

2. There are numerous possibilities to visualize the data from the various algorithms. Any timestamps of an algorithm output can be dragged and dropped into the graphs. The Scope marks the position of the event with a colored marker (see blue line in the image below).



Classification — '3.54 GB' received (1.56 MB/s) — Networks

Threshold Classificator 1Ch1
Input: Variables.fGrowSlow @ 172.17.62.145: Te — 24592
Level OK / Warning: 24589
Level Warning / Alarm: 24591
Class: 3
Last Event Warning: 27/06/2019 16:51:29.586
Last Event Alarm: 27/06/2019 16:51:33.914
New Result: FALSE
WB-ID: 7d20a327-234e-4df7-a09a-60f8775dbf25

Motion section control — XY Shape Monitor Rectangle 2Ch1
Input ChX: Variables.fSine @ 172.17.62.145: TestSign — 10.614
Input ChY: Variables.fGrowSlow @ 172.17.62.145: Te — 24592
Lower Left Corner X: -8
Lower Left Corner Y: 24591
Length X: 4
Length Y: 8
Within Shape: False
Intersection: False
Count Intersections: 1865
Last Intersection: 27/06/2019 16:51:34.525
New Result: FALSE

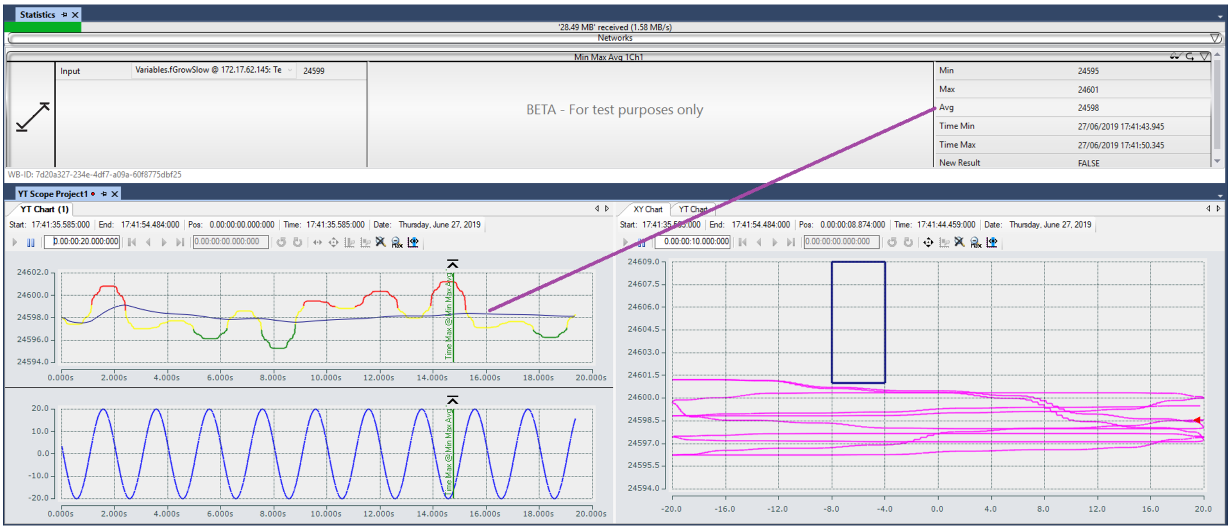
Statistics — Min Max Avg 1Ch1
Input: Variables.fGrowSlow @ 172.17.62.145: Te — 24592
Min: 24579
Max: 24592
Avg: 24586
Time Min: 27/06/2019 16:12:13.065
Time Max: 27/06/2019 16:50:56.265
New Result: FALSE

YT Scope Project1 — YT Chart (1): Start: 16:46:34.134:000 End: 16:51:34.134:000 Pos: 0.00:04:13.900:000 Time: 16:50:48.034:000 Date: Thursday, June 27, 2019

XY Chart: Start: 16:46:34.134:000 End: 16:51:34.134:000 Pos: 0.00:04:58.950:000 Time: 16:51:33.084:000 Date: Thursday, June 27, 2019

3. For each type of Shape algorithm, the SW creates an XY diagram in the Scope configuration including a shape with the given definition (see orange line in the image above). For the Threshold classification algorithm, a dynamic color change from channel color to yellow (warning) and to red (alarm) (pink line) also occurs automatically in the Scope View.

4. In addition, you can drag and drop the algorithm's result values into a graph before or during recording to add a new channel showing, for example, the average, minimum, or maximum value.



Statistics — '28.49 MB' received (1.58 MB/s) — Networks

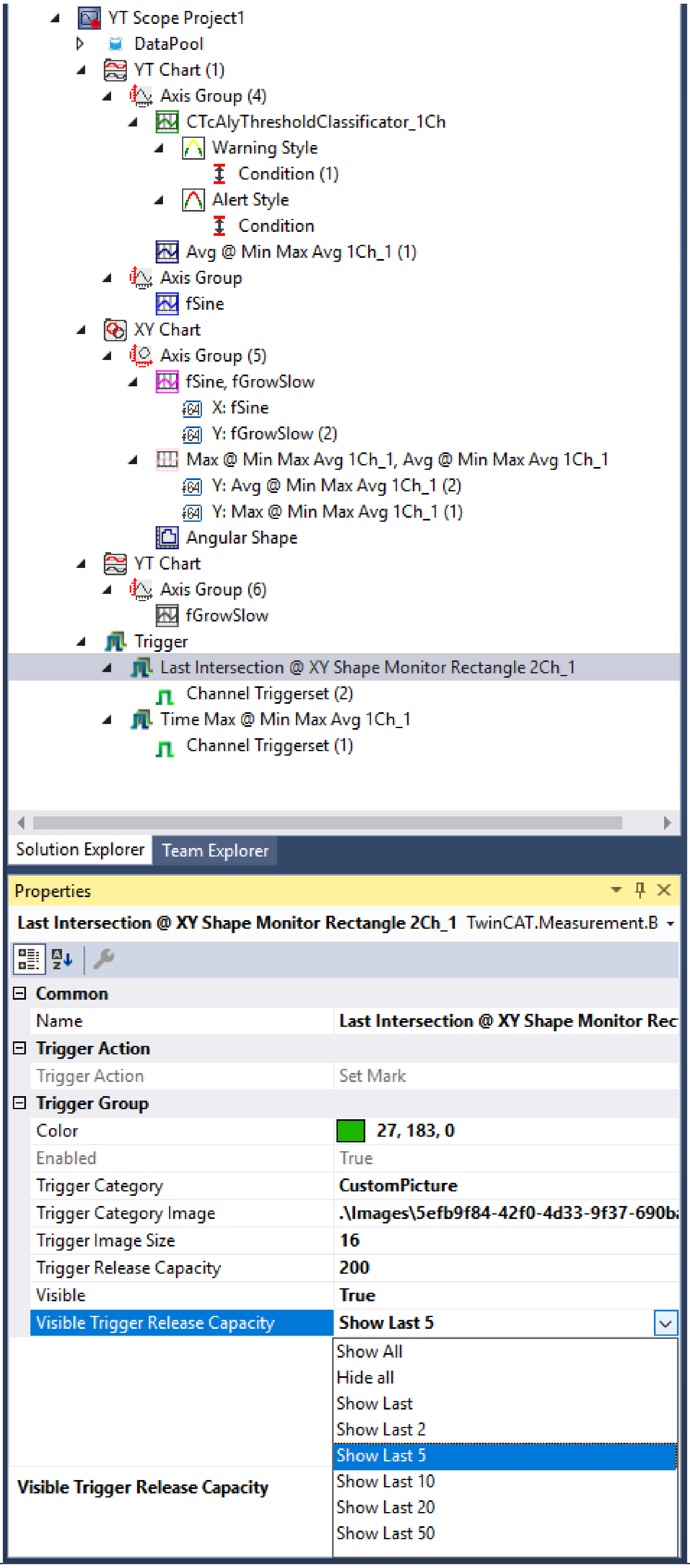
Min Max Avg 1Ch1
Input: Variables.fGrowSlow @ 172.17.62.145: Te — 24599
Min: 24595
Max: 24601
Avg: 24598
Time Min: 27/06/2019 17:41:43.945
Time Max: 27/06/2019 17:41:50.345
New Result: FALSE
WB-ID: 7d20a327-234e-4df7-a09a-60f8775dbf25

YT Scope Project1 — YT Chart (1): Start: 17:41:35.585:000 End: 17:41:54.484:000 Pos: 0.00:00:00.000:000 Time: 17:41:35.585:000 Date: Thursday, June 27, 2019

XY Chart: Start: 17:41:35.585:000 End: 17:41:54.484:000 Pos: 0.00:00:08.874:000 Time: 17:41:44.459:000 Date: Thursday, June 27, 2019

5. Click the appropriate trigger group in the Scope View tree to control the number of events/markers displayed.

6. In the Properties window you can set a number for Visible Trigger Release Capacity.



Last Intersection @ XY Shape Monitor Rectangle 2Ch_1 TwinCAT.Measurement.B

Common	
Name	**Last Intersection @ XY Shape Monitor Rec**
Trigger Action	
Trigger Action	Set Mark
Trigger Group	
Color	🟩 **27, 183, 0**
Enabled	True
Trigger Category	**CustomPicture**
Trigger Category Image	**.\Images\5efb9f84-42f0-4d33-9f37-690b**
Trigger Image Size	**16**
Trigger Release Capacity	**200**
Visible	**True**
Visible Trigger Release Capacity	**Show Last 5**

Visible Trigger Release Capacity

⇨ You can choose between:

- All
- Hide All
- Show Last
- Show Last 2
- Show Last 5
- Show Last 10
- Show Last 20
- Show Last 50

Scope configuration stored in network template

A created Scope configuration can be saved together with the associated network in a network template [▸ 348], in order to automatically obtain the same Scope configuration when a network is used again.

Time relationship between Analytics configuration and Scope View

Note that the recording time in Scope View may differ from that in Analytics. Especially if the ring buffer is selected in the Scope View, it could be possible that some significant values of your analysis are in a past Scope recording!

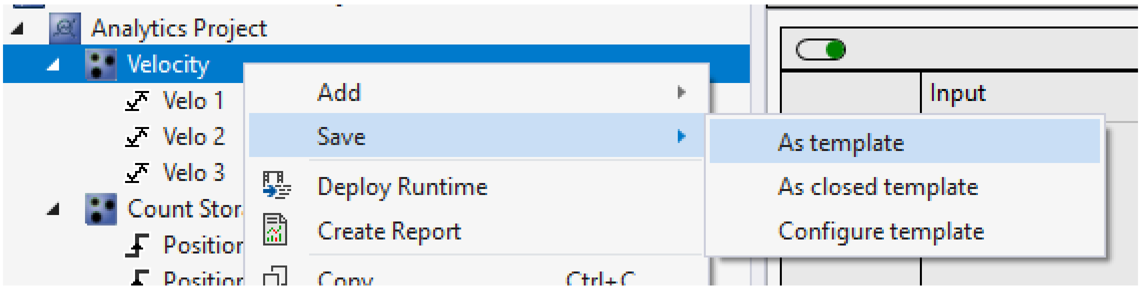
6.5.1 Scope configuration in the network template

In addition to the configuration of a network, the configuration of the connected Scope project can also be saved in a network template. However, only those parts of the Scope can be saved that have a connection to the selected network.

Depending on whether the Scope configuration to be saved is still to be adjusted manually, the template can be saved in two ways.

Save without manual configuration

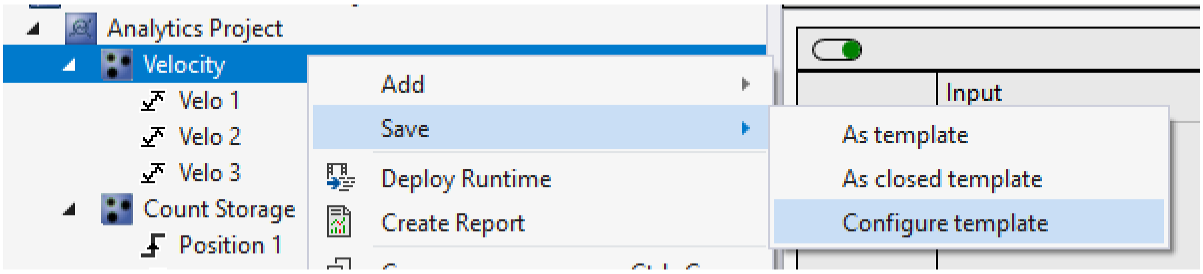
If you want to save the Scope configuration in the template without any manual adjustment, select the context menu items **Save-As template** and **Save-As closed template.**



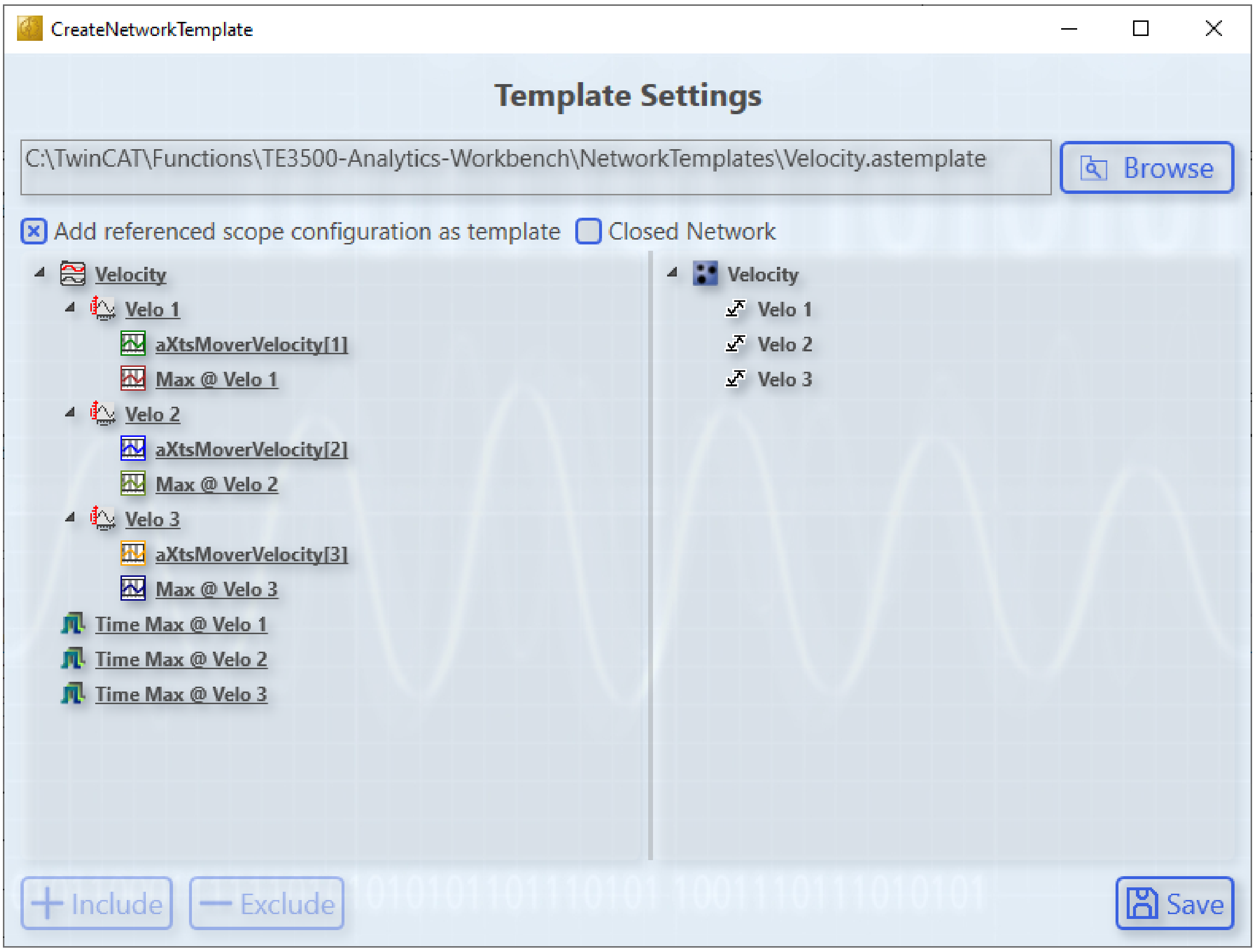
In the following dialog, specify the location and name of the template. At this point you can use the file type **Save as type** to specify whether the Scope configuration should also be saved (**Scope Network Template**) or whether only the network (**Network Template**) should be saved.

Save with manual configuration

If the Scope configuration is to be checked and, if necessary, adjusted before saving, select the lowest entry in the context menu **Configure template**.



To control and adjust the Scope configuration, a new window opens, showing all Scope components on the left and all Analytics components on the right:



C:\TwinCAT\Functions\TE3500-Analytics-Workbench\NetworkTemplates\Velocity.astemplate

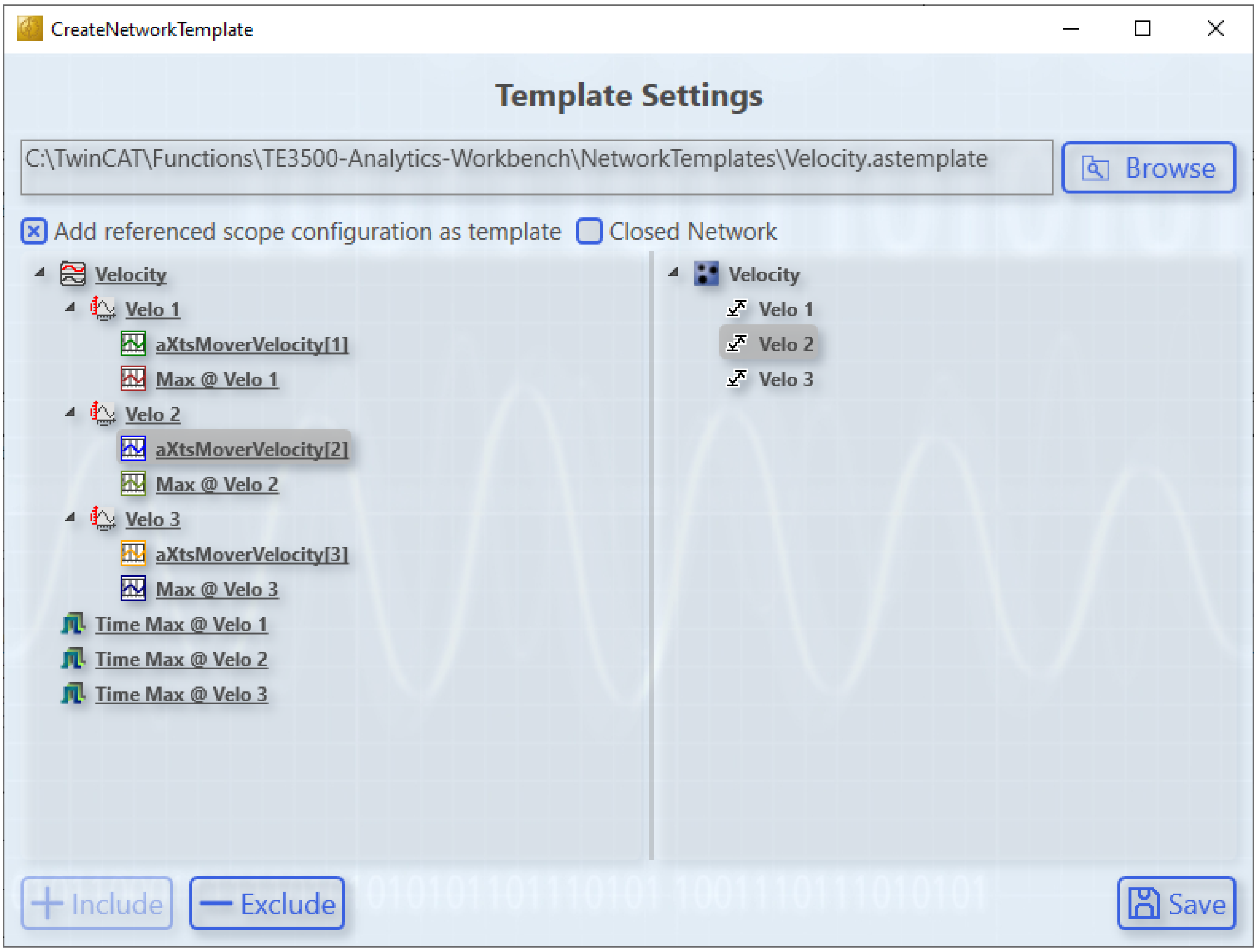
☒ Add referenced scope configuration as template □ Closed Network

In the upper line you can set the location and name of the template. Directly below you can make general settings that affect the template.

- **Add referenced scope configuration as template**
  Activate this setting to save the scope configuration in the template.
- **Closed Network**
  This setting determines in which format the template [▸ 67] should be saved. This setting is only visible if at least one network input and one network output are configured in the network.

In the two tree views below, the configurations are displayed in order to still edit the Scope configuration if necessary. Here you can exclude Scope components from the template but also add them again.
All components which are added to the template are underlined in the view.

When elements are selected in the Scope tree, all Analytics components that have a direct dependency on the Scope component are also selected.



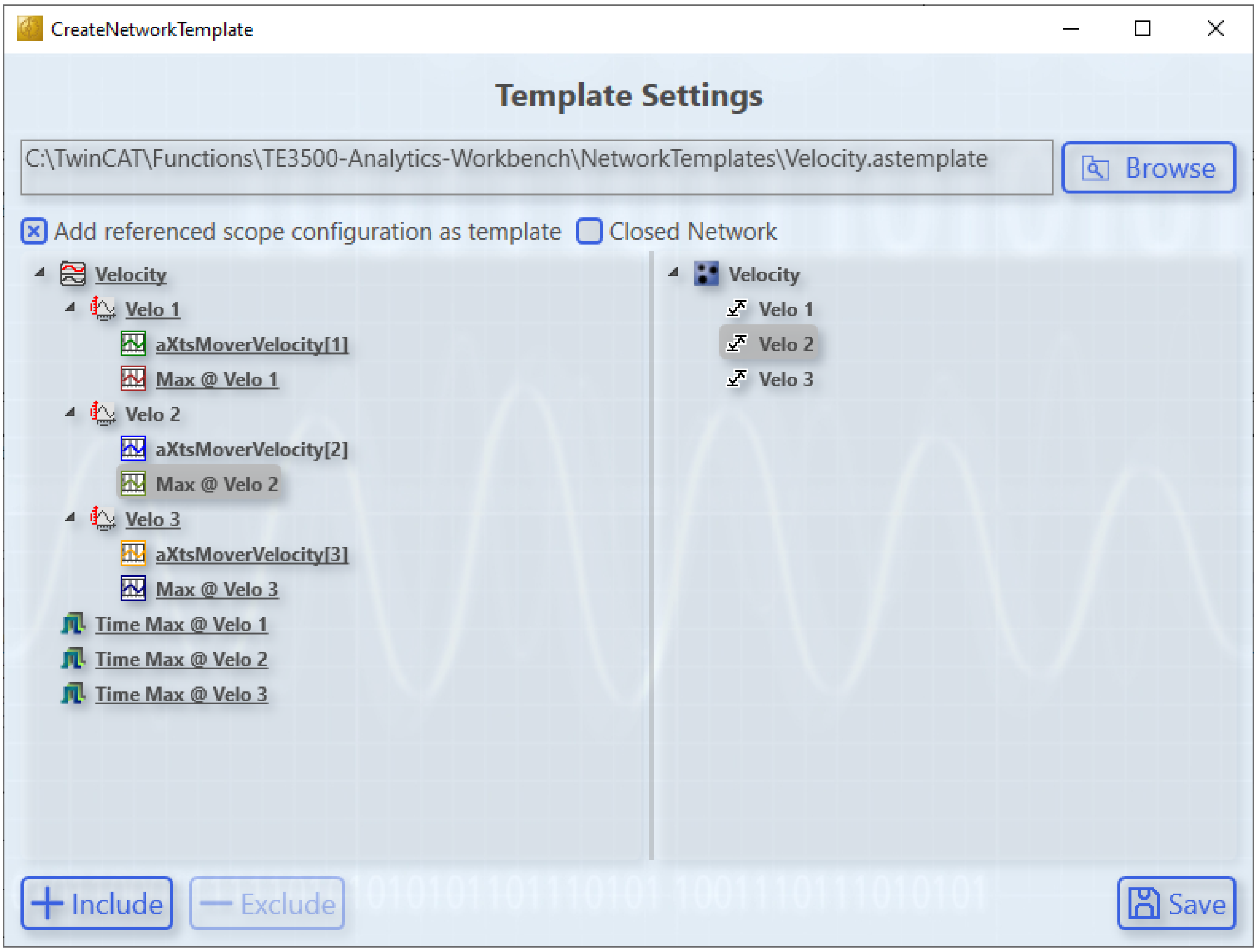
CreateNetworkTemplate

Template Settings

C:\TwinCAT\Functions\TE3500-Analytics-Workbench\NetworkTemplates\Velocity.astemplate

☒ Add referenced scope configuration as template ☐ Closed Network

In the example above, selecting **aXtsMoverVelocity[2**] automatically selects the module **Velo 2** because the channel indicates the module's input variable.
If a Scope component is selected that is still underlined and thus stored in the template, you can use the **Exclude** button to remove it from the template.
If you remove an element from the template, all elements that lie below it are also automatically removed. In addition, the element above is taken out if every element below it is also taken out.



Identical to removing elements, you can select Scope components to add them back to the template using the button **Include**.
It should be noted that all overlying elements are also automatically added back to the template, even if they were previously deselected.

Adding Scope configurations from the template

To use a template with the Scope configuration, you need to add the template to the Analytics project. If there is already a referenced Scope project for the project, the template configuration is added directly there. If no referenced Scope project exists yet, the template configuration will be added when the Scope project is created later.
Before the configuration is added to the project, a query opens where you can also cancel the automatic addition.

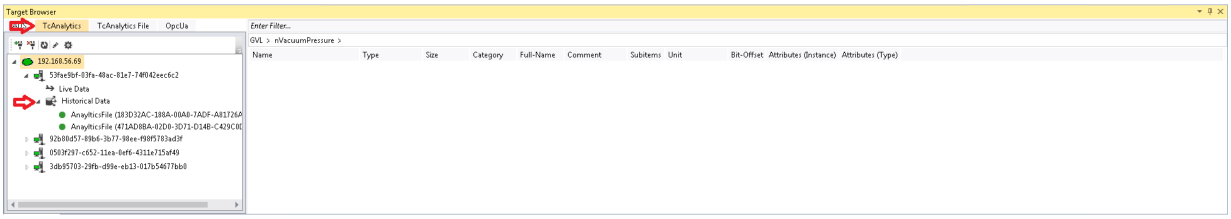
6.6 Working with Historical Data

Historical Data can be analysed with the Analytics Workbench or the Analytics Service Tool. To see your recorded data, you need the TwinCAT Target Browser.

Selection of data from the TwinCAT Target Browser

The historical data can be pulled directly from the Target Browser to an input of an analysis algorithm.

1. First, you need to click **TcAnalytics** in the left corner of Target Browser. There you can see your configured broker, which lists live and historical data from your various devices. This should look like the following figure.



Target Browser

ADS | TcAnalytics | TcAnalytics File | OpcUa

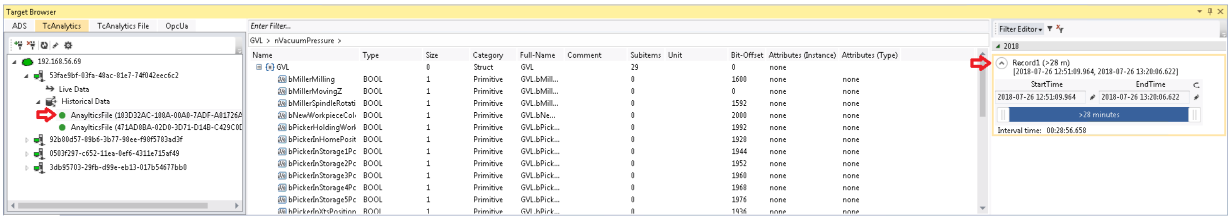
- 192.168.56.69
  - 53fae9bf-03fa-48ac-81e7-74f042eec6c2
    - Live Data
    - Historical Data
      - AnayIticsFile (183D32AC-188A-00A0-7ADF-A81726A
      - AnayIticsFile (471AD8BA-02D0-3D71-D14B-C429C0D
  - 92b80d57-89b6-3b77-98ee-f98f5783ad3f
  - 0503f297-c652-11ea-0ef6-4311e715af49
  - 3db95703-29fb-d99e-eb13-017b54677bb0

Enter Filter...

GVL > nVacuumPressure >

Name	Type	Size	Category	Full-Name	Comment	Subitems	Unit	Bit-Offset	Attributes (Instance)	Attributes (Type)

2. Go to the historical stream you created and select the recording to be analyzed. All your records are listed in the **Record** window on the right. The last recording is selected by default.



Target Browser

ADS | TcAnalytics | TcAnalytics File | OpcUa

- 192.168.56.69
  - 53fae9bf-03fa-48ac-81e7-74f042eec6c2
    - Live Data
    - Historical Data
      - AnayIticsFile (183D32AC-188A-00A0-7ADF-A81726A
      - AnayIticsFile (471AD8BA-02D0-3D71-D14B-C429C0D
  - 92b80d57-89b6-3b77-98ee-f98f5783ad3f
  - 0503f297-c652-11ea-0ef6-4311e715af49
  - 3db95703-29fb-d99e-eb13-017b54677bb0

Enter Filter...

GVL > nVacuumPressure >

Name	Type	Size	Category	Full-Name	Comment	Subitems	Unit	Bit-Offset	Attributes (Instance)	Attributes (Type)
GVL		0	Struct	GVL		29		0	none	
bMillerMilling	BOOL	1	Primitive	GVL.bMill...		0		1600	none	none
bMillerMovingZ	BOOL	1	Primitive	GVL.bMill...		0		0	none	none
bMillerSpindleRotati	BOOL	1	Primitive	GVL.bMill...		0		1592	none	none
bNewWorkpieceCol	BOOL	1	Primitive	GVL.bNe...		0		2000	none	none
bPickerHoldingWorl	BOOL	1	Primitive	GVL.bPick...		0		1992	none	none
bPickerInHomePosit	BOOL	1	Primitive	GVL.bPick...		0		1928	none	none
bPickerInStorage1Pc	BOOL	1	Primitive	GVL.bPick...		0		1944	none	none
bPickerInStorage2Pc	BOOL	1	Primitive	GVL.bPick...		0		1952	none	none
bPickerInStorage3Pc	BOOL	1	Primitive	GVL.bPick...		0		1960	none	none
bPickerInStorage4Pc	BOOL	1	Primitive	GVL.bPick...		0		1968	none	none
bPickerInStorage5Pc	BOOL	1	Primitive	GVL.bPick...		0		1976	none	none
bPickerInXtsPosition	BOOL	1	Primitive	GVL.bPick...		0		1936	none	none

Filter Editor

2018

Record1 (>28 m)
[2018-07-26 12:51:09.964, 2018-07-26 13:20:06.622]

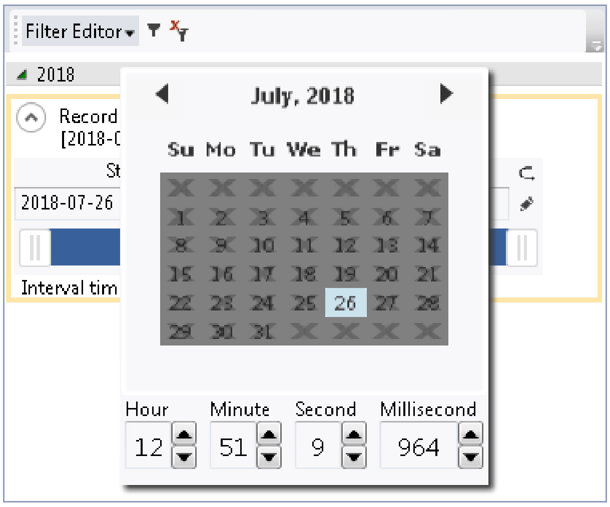
StartTime: 2018-07-26 12:51:09.964

EndTime: 2018-07-26 13:20:06.622

>28 minutes

Interval time: 00:28:56.658

3. When you record live, the time range of the recording is updated every few seconds. The entire time range of a recording is used by default. You can also edit the start and end time to analyze your desired data area. This can be done with a slider, text fields or in a graphical calendar view. If you click on the symbol to the right of the text fields, the calendar view will be displayed.



Filter Editor

2018

Record
[2018-(

2018-07-26

Interval tim

July, 2018

Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

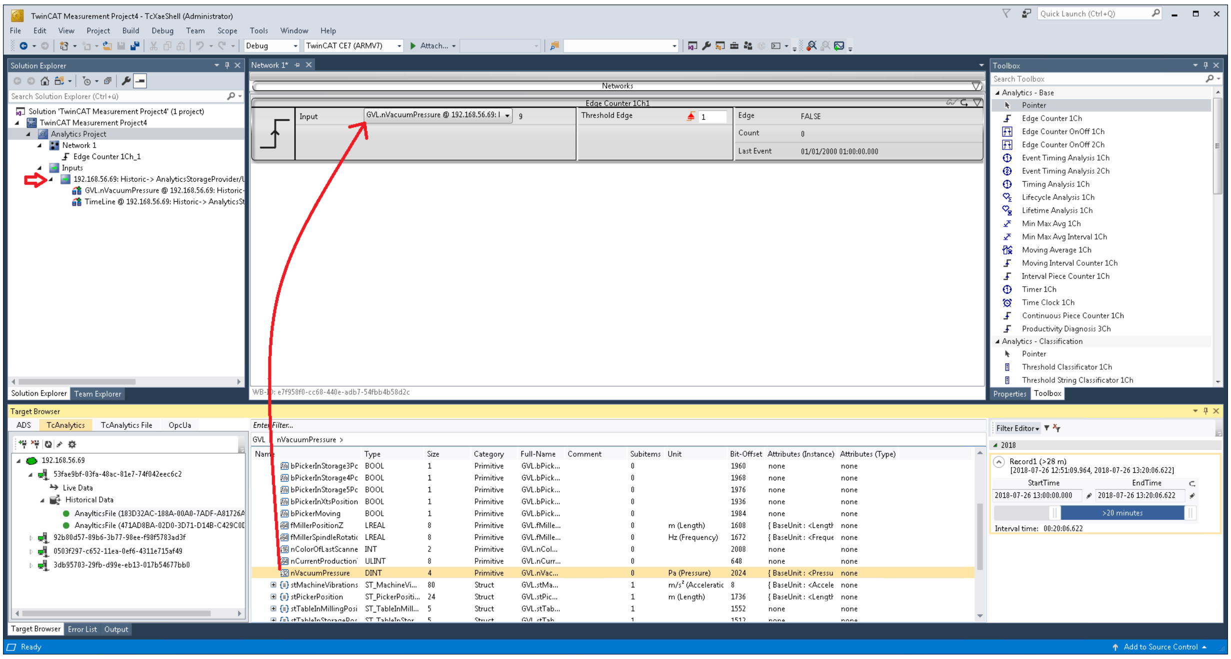
Hour: 12

Minute: 51

Second: 9

Millisecond: 964

4. After these steps, you can drag and drop a symbol to an input of an algorithm just as you do with the symbols of the live data.



Solution 'TwinCAT Measurement Project4' (1 project)
- TwinCAT Measurement Project4
  - Analytics Project
    - Network 1
      - Edge Counter 1Ch_1
    - Inputs
      - 192.168.56.69: Historic-> AnalyticsStorageProvider/
        - GVL.nVacuumPressure @ 192.168.56.69: Historic-
        - TimeLine @ 192.168.56.69: Historic-> AnalyticsSt

Edge Counter 1Ch1

Input	GVL.nVacuumPressure @ 192.168.56.69: I	9	
Threshold Edge	1	Edge	FALSE
		Count	0
		Last Event	01/01/2000 01:00:00.000

WB-ID: e7f958f0-cc68-440e-adb7-54fbb4b58d2c

Toolbox
- Analytics - Base
  - Pointer
  - Edge Counter 1Ch
  - Edge Counter OnOff 1Ch
  - Edge Counter OnOff 2Ch
  - Event Timing Analysis 1Ch
  - Event Timing Analysis 2Ch
  - Timing Analysis 1Ch
  - Lifecycle Analysis 1Ch
  - Lifetime Analysis 1Ch
  - Min Max Avg 1Ch
  - Min Max Avg Interval 1Ch
  - Moving Average 1Ch
  - Moving Interval Counter 1Ch
  - Interval Piece Counter 1Ch
  - Timer 1Ch
  - Time Clock 1Ch
  - Continuous Piece Counter 1Ch
  - Productivity Diagnosis 3Ch
- Analytics - Classification
  - Pointer
  - Threshold Classificator 1Ch
  - Threshold String Classificator 1Ch

Target Browser (TcAnalytics)
- 192.168.56.69
  - 53fae9bf-03fa-48ac-81e7-74f042eec6c2
    - Live Data
    - Historical Data
      - AnaylticsFile (183D32AC-188A-00A0-7ADF-A81726A
      - AnaylticsFile (471AD8BA-02D0-3D71-D14B-C429C0D
  - 92b80d57-89b6-3b77-98ee-f98f5783ad3f
  - 0503f297-c652-11ea-0ef6-4311e715af49
  - 3db95703-29fb-d99e-eb13-017b54677bb0

GVL nVacuumPressure >

Name	Type	Size	Category	Full-Name	Comment	Subitems	Unit	Bit-Offset	Attributes (Instance)	Attributes (Type)
bPickerInStorage3Pc	BOOL	1	Primitive	GVL.bPick...		0		1960	none	none
bPickerInStorage4Pc	BOOL	1	Primitive	GVL.bPick...		0		1968	none	none
bPickerInStorage5Pc	BOOL	1	Primitive	GVL.bPick...		0		1976	none	none
bPickerInXtsPosition	BOOL	1	Primitive	GVL.bPick...		0		1936	none	none
bPickerMoving	BOOL	1	Primitive	GVL.bPick...		0		1984	none	none
fMillerPositionZ	LREAL	8	Primitive	GVL.fMille...		0	m (Length)	1608	{ BaseUnit : <Length	none
fMillerSpindleRotatic	LREAL	8	Primitive	GVL.fMille...		0	Hz (Frequency)	1672	{ BaseUnit : <Freque	none
nColorOfLastScanne	INT	2	Primitive	GVL.nCol...		0		2008	none	none
nCurrentProduction	ULINT	8	Primitive	GVL.nCurr...		0		648	none	none
nVacuumPressure	DINT	4	Primitive	GVL.nVac...		0	Pa (Pressure)	2024	{ BaseUnit : <Pressu	none
stMachineVibrations	ST_MachineVi...	80	Struct	GVL.stMa...		1	m/s² (Acceleratic	8	{ BaseUnit : <Accele	none
stPickerPosition	ST_PickerPositi...	24	Struct	GVL.stPic...		1	m (Length)	1736	{ BaseUnit : <Length	none
stTableInMillingPosi	ST_TableInMill...	5	Struct	GVL.stTab...		1		1552	none	none
stTableInStoragePos	ST_TableInStor	5	Struct	GVL.stTab...		1		1512	none	none

2018
Record1 (>28 m)
[2018-07-26 12:51:09.964, 2018-07-26 13:20:06.622]

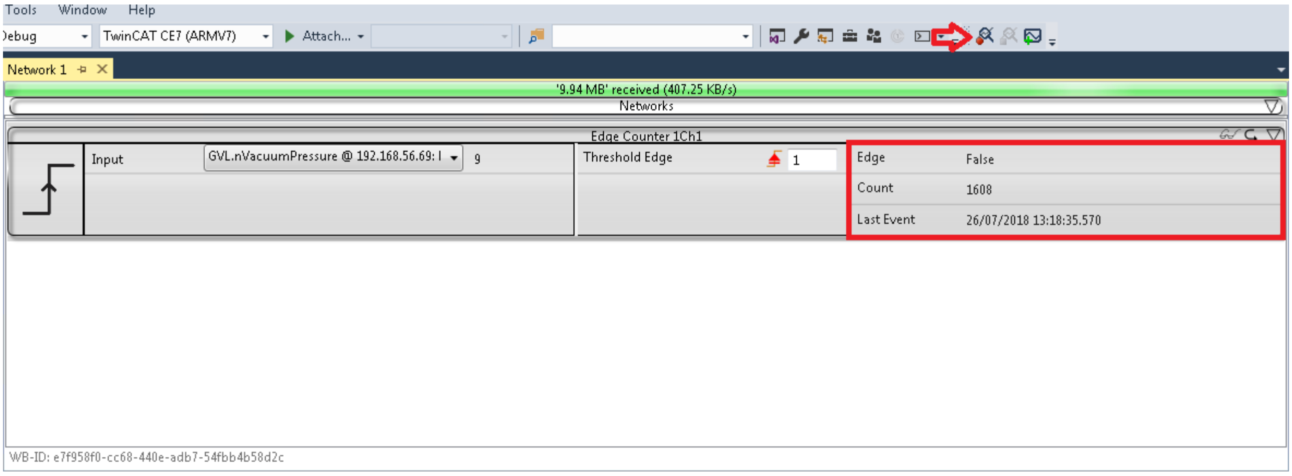
StartTime	EndTime
2018-07-26 13:00:00.000	2018-07-26 13:20:06.622

>20 minutes
Interval time: 00:20:06.622

⇨ A new input source for your historical stream is then generated and can be displayed in the Solution Explorer of your Visual Studio®. First, the dragged symbol and a timestamp of the current device time are listed under this stream. Also new drawn symbols of this stream are listed there.

Analyse your historical data in the Analytics Configurator

To analyse your historical data press on the Start Analytics button. In contrast to analysing live data, a green progress bar appears. The speed of your analysis depends on your record length, the amount and size of your symbols as well as on your broadband speed to the broker. The analysis stops automatically when the progress bar ends. The results will remain visible.



'9.94 MB' received (407.25 KB/s)

Edge Counter 1Ch1

Input	GVL.nVacuumPressure @ 192.168.56.69: I	9	
Threshold Edge	1	Edge	False
		Count	1608
		Last Event	26/07/2018 13:18:35.570

WB-ID: e7f958f0-cc68-440e-adb7-54fbb4b58d2c

6.7 Runtime deployment

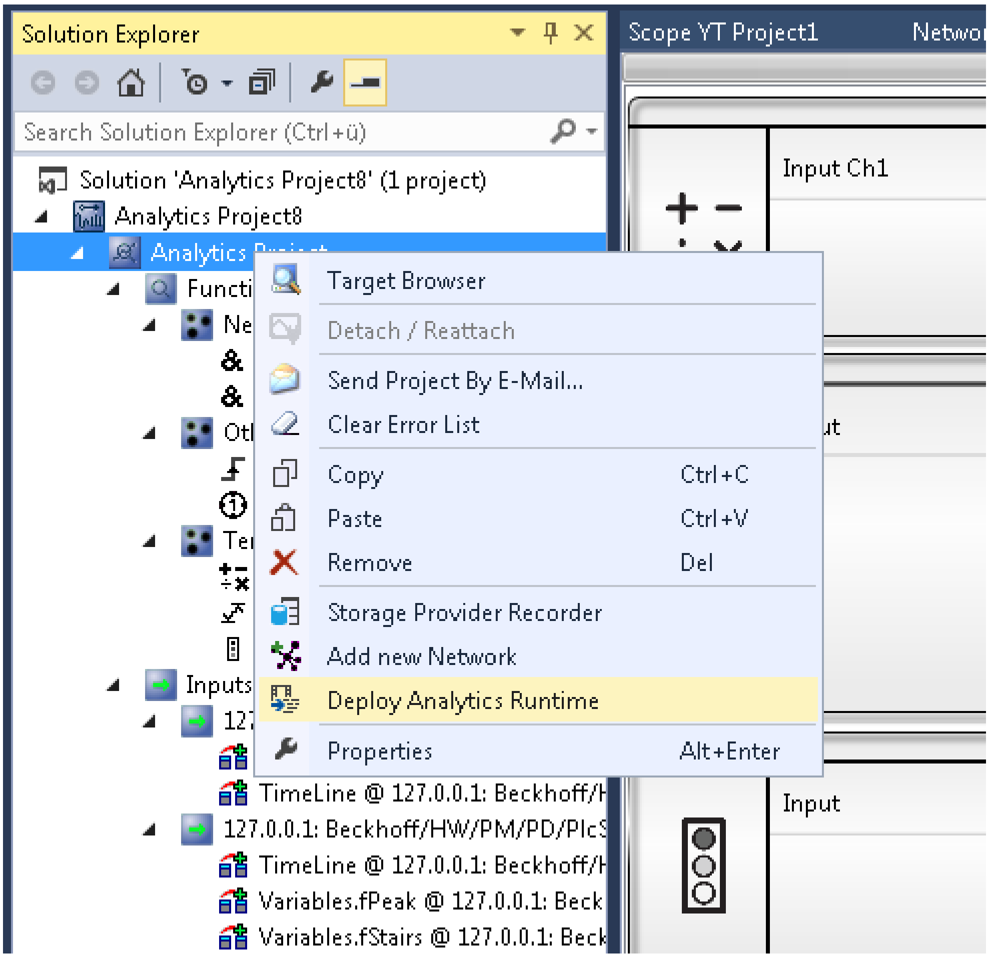
PLC code can be generated with all modules and parameters configured in the TwinCAT Analytics Workbench Configurator. This code can be downloaded into a TwinCAT Analytics runtime to realize 24/7 data analysis.

NOTICE

Compatibility of automatically generated PLC code

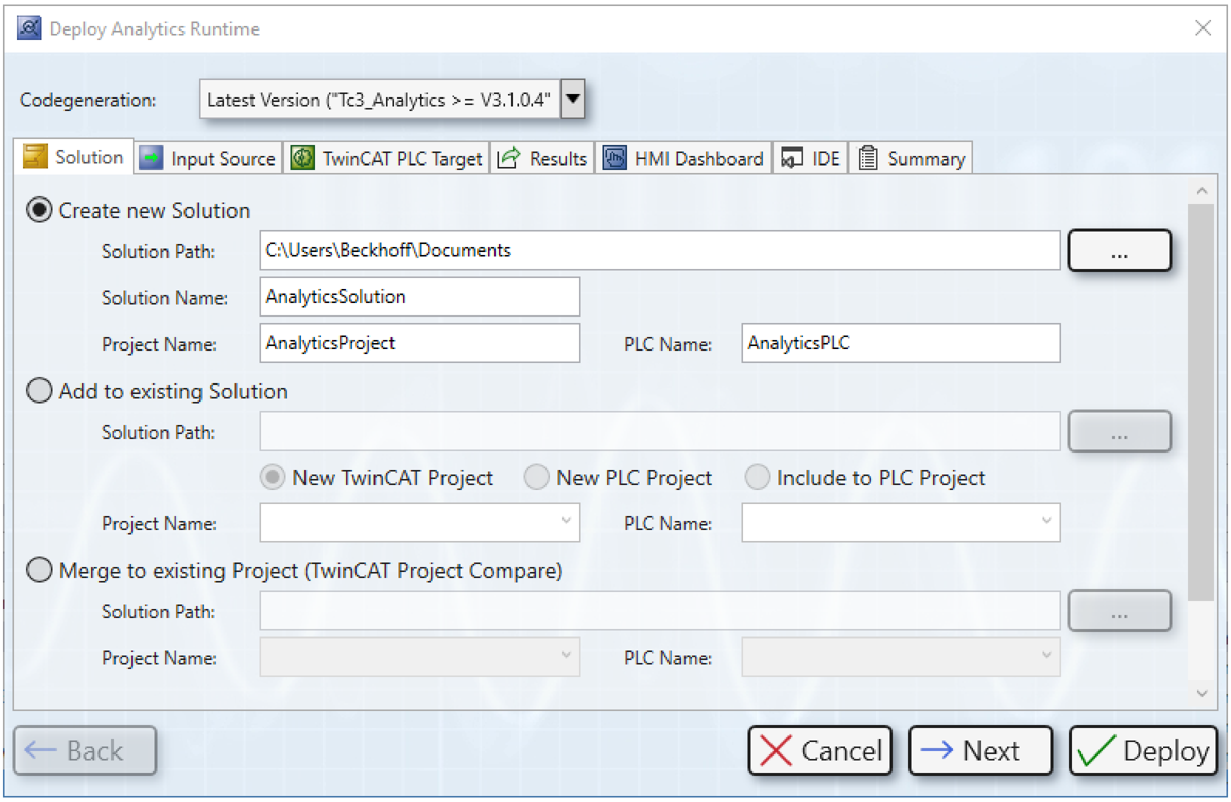
The automatically generated PLC code is based on the TwinCAT Analytics Library. Interfaces of the base models of the library are compatible with earlier versions after release of the library. The automatically generated code itself is only a sample code! This code generation may change from version to version. As far as possible, this is solved by the code generation version.

✓ After configuration, click on the **Deploy Analytics Runtime** command in the context menu.


Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'Analytics Project8' (1 project)
Analytics Project8
Analytics Project
Functi
Ne
Ot
Te
Inputs
127
TimeLine @ 127.0.0.1: Beckhoff/H
127.0.0.1: Beckhoff/HW/PM/PD/PlcS
TimeLine @ 127.0.0.1: Beckhoff/H
Variables.fPeak @ 127.0.0.1: Beck
Variables.fStairs @ 127.0.0.1: Beck
Scope YT Project1
Netwo
Input Ch1
ut
Input
Target Browser
Detach / Reattach
Send Project By E-Mail...
Clear Error List
Copy Ctrl+C
Paste Ctrl+V
Remove Del
Storage Provider Recorder
Add new Network
Deploy Analytics Runtime
Properties Alt+Enter

✓ The Deploy Wizard starts and it is possible to set up the entire required configuration step by step for use.

1. On the first tab **Solution**, select whether a new Solution is to be created (**Create new Solution**), whether the analysis PLC code is to be integrated or added to an existing Solution as a new project, new PLC or to an existing PLC (**Add to existing Solution**), or whether the new project is to be added to an existing solution using the TwinCAT Project Compare Tool (**Merge to existing Project**).



Deploy Analytics Runtime

Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4"

● Create new Solution

Solution Path: C:\Users\Beckhoff\Documents

Solution Name: AnalyticsSolution

Project Name: AnalyticsProject PLC Name: AnalyticsPLC

○ Add to existing Solution

Solution Path:

● New TwinCAT Project ○ New PLC Project ○ Include to PLC Project

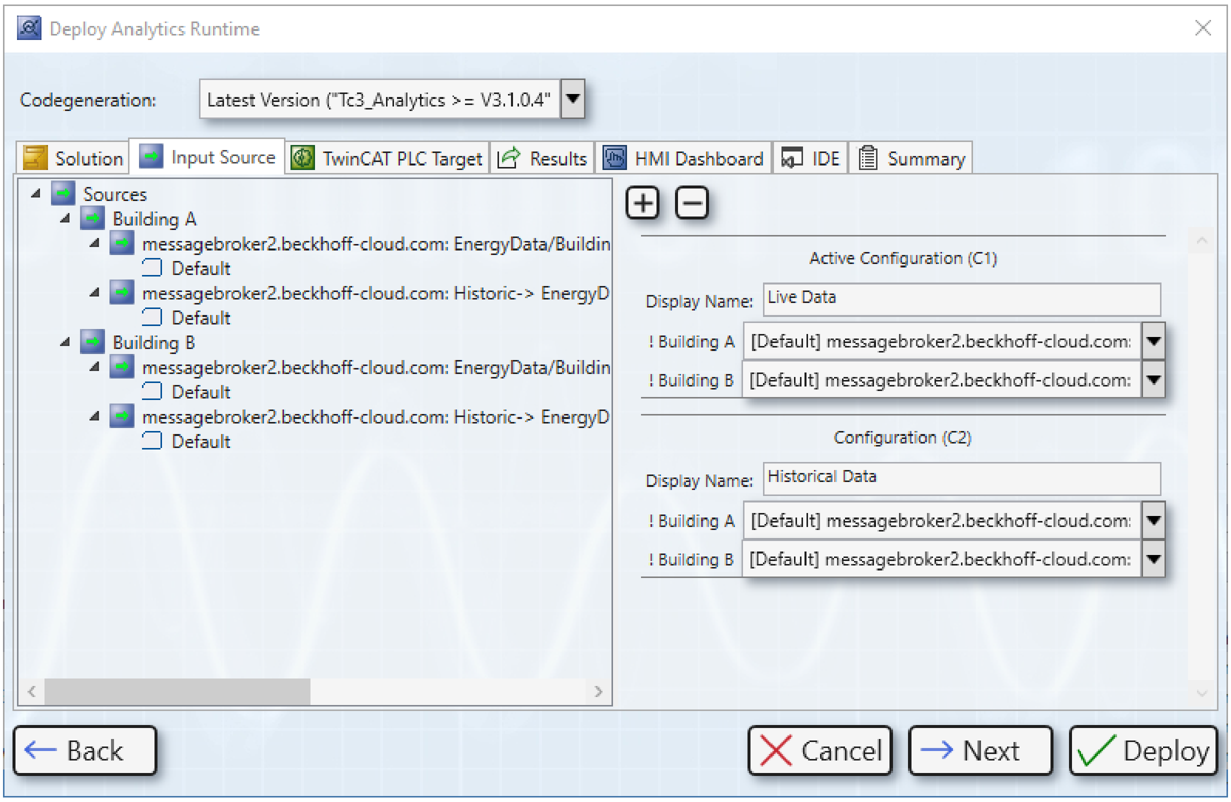
Project Name: PLC Name:

○ Merge to existing Project (TwinCAT Project Compare)

Solution Path:

Project Name: PLC Name:

2. Select several data source configurations on the second tab **Input Source**. This is necessary, for example, if both live data and historical data are to be analyzed.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4"

- Sources
  - Building A
    - messagebroker2.beckhoff-cloud.com: EnergyData/Buildin
      - Default
    - messagebroker2.beckhoff-cloud.com: Historic-> EnergyD
      - Default
  - Building B
    - messagebroker2.beckhoff-cloud.com: EnergyData/Buildin
      - Default
    - messagebroker2.beckhoff-cloud.com: Historic-> EnergyD
      - Default

Active Configuration (C1)

Display Name: Live Data

! Building A [Default] messagebroker2.beckhoff-cloud.com:

! Building B [Default] messagebroker2.beckhoff-cloud.com:

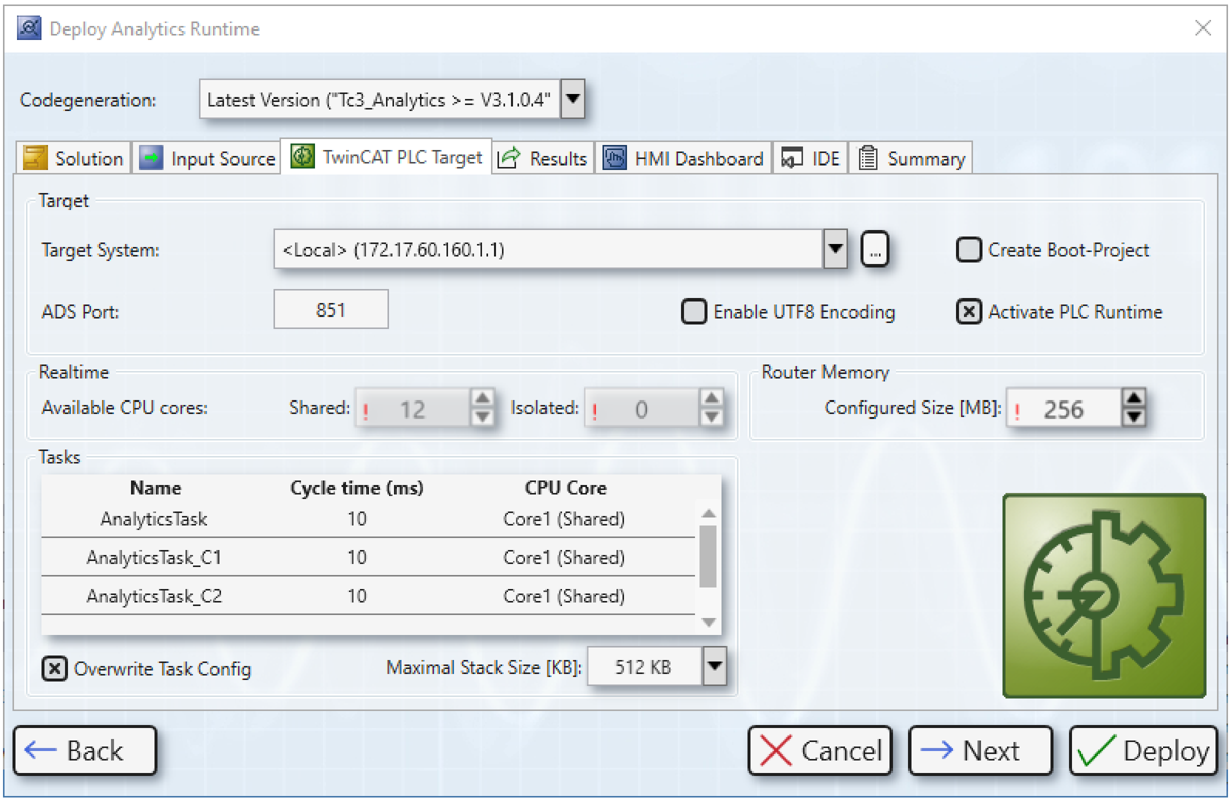
Configuration (C2)

Display Name: Historical Data

! Building A [Default] messagebroker2.beckhoff-cloud.com:

! Building B [Default] messagebroker2.beckhoff-cloud.com:

3. On the third tab **TwinCAT PLC Target** PLC-specific parameters such as target system, task cycle time or task assignment to corresponding CPU cores can be specified.



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4")

Target

Target System: <Local> (172.17.60.160.1.1)

□ Create Boot-Project

ADS Port: 851

□ Enable UTF8 Encoding

☒ Activate PLC Runtime

Realtime

Available CPU cores: Shared: 12 Isolated: 0

Router Memory

Configured Size [MB]: 256

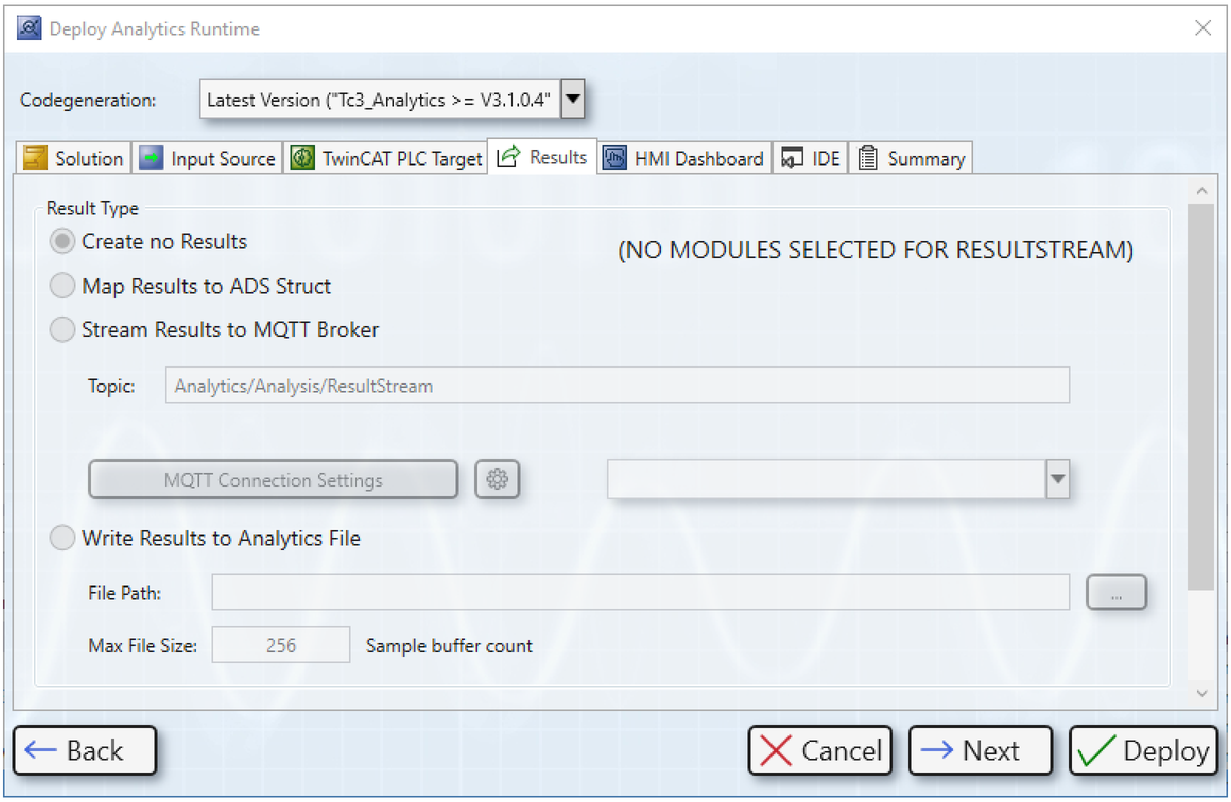
Tasks

Name	Cycle time (ms)	CPU Core
AnalyticsTask	10	Core1 (Shared)
AnalyticsTask_C1	10	Core1 (Shared)
AnalyticsTask_C2	10	Core1 (Shared)

☒ Overwrite Task Config

Maximal Stack Size [KB]: 512 KB

4. If you have set the **PLC Result** property of some functions in the configurator, the **Result** tab is enabled. Specify where the results are streamed or saved.



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4")

Result Type

● Create no Results

(NO MODULES SELECTED FOR RESULTSTREAM)

○ Map Results to ADS Struct

○ Stream Results to MQTT Broker

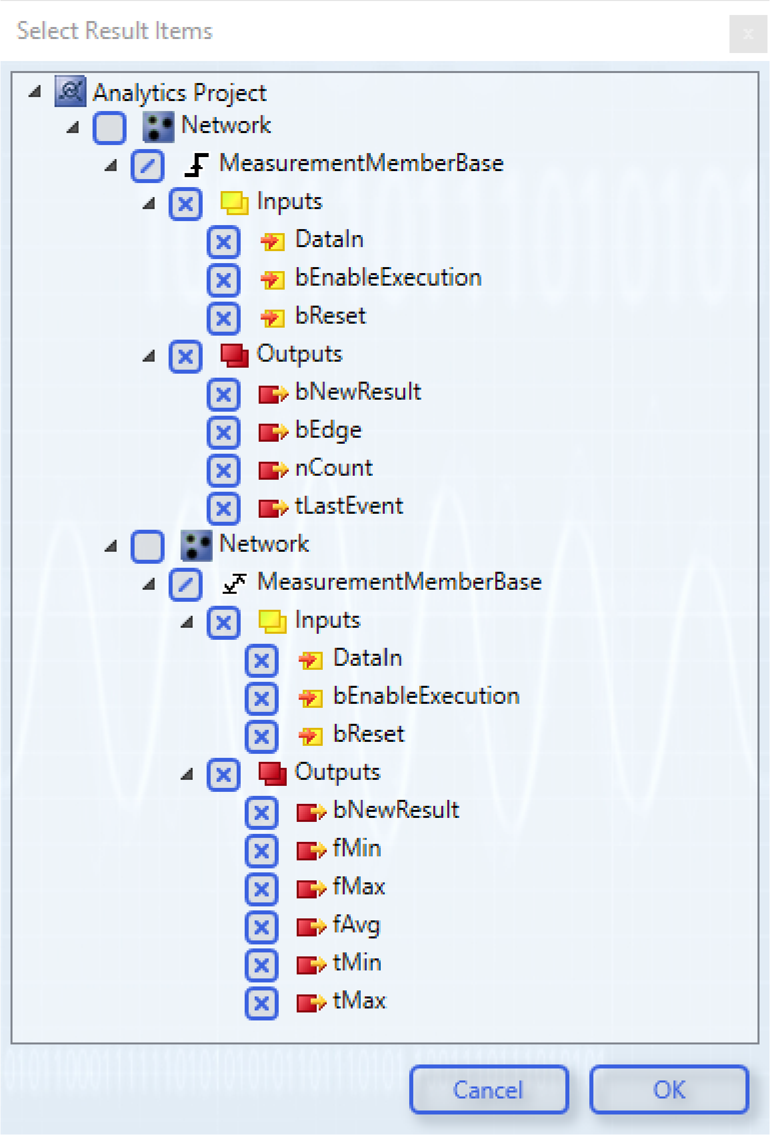
Topic: Analytics/Analysis/ResultStream

○ Write Results to Analytics File

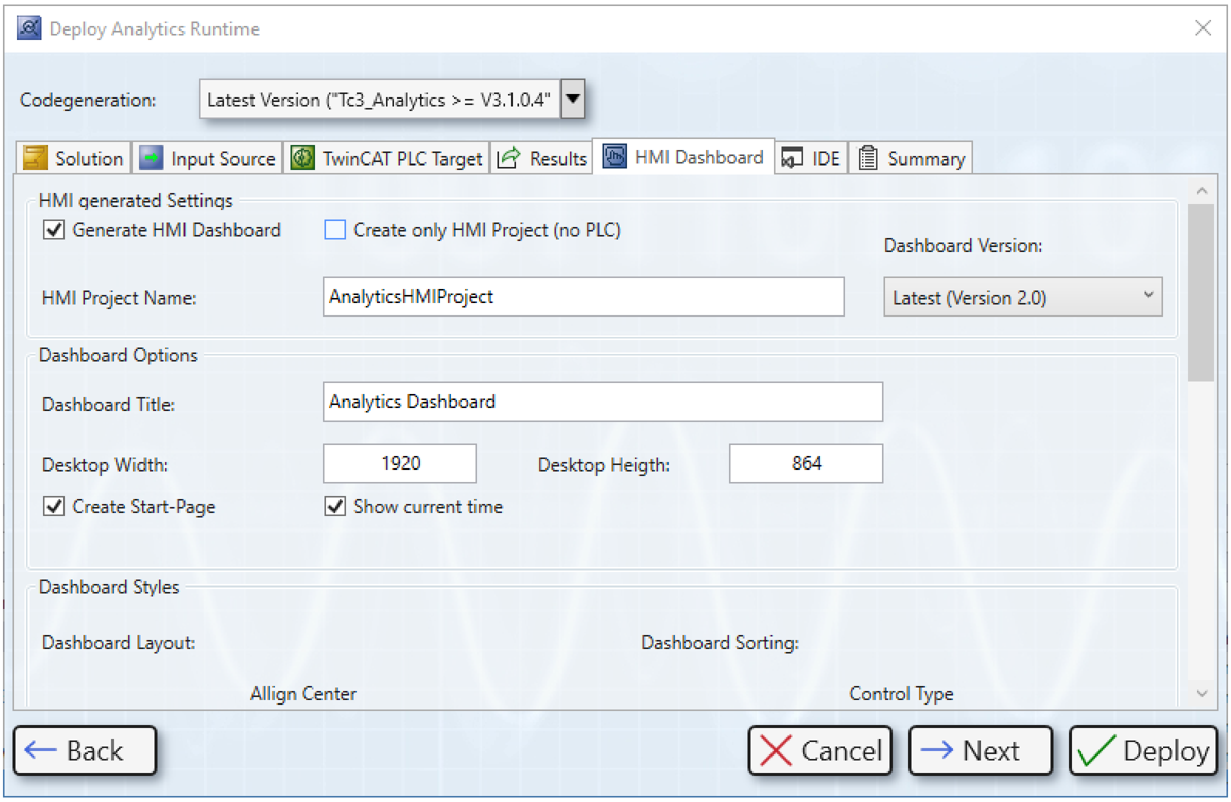
File Path:

Max File Size: 256 Sample buffer count

5. By clicking **Select Result Items** it is possible to select only the desired values.



6. If you have assigned HMI Controls to one or more functions, the **HMI Dashboard** tab is enabled. Here you can make various settings to generate a customized dashboard that suits your needs.



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4")

HMI generated Settings

☑ Generate HMI Dashboard ☐ Create only HMI Project (no PLC)

Dashboard Version: Latest (Version 2.0)

HMI Project Name: AnalyticsHMIProject

Dashboard Options

Dashboard Title: Analytics Dashboard

Desktop Width: 1920 Desktop Heigth: 864

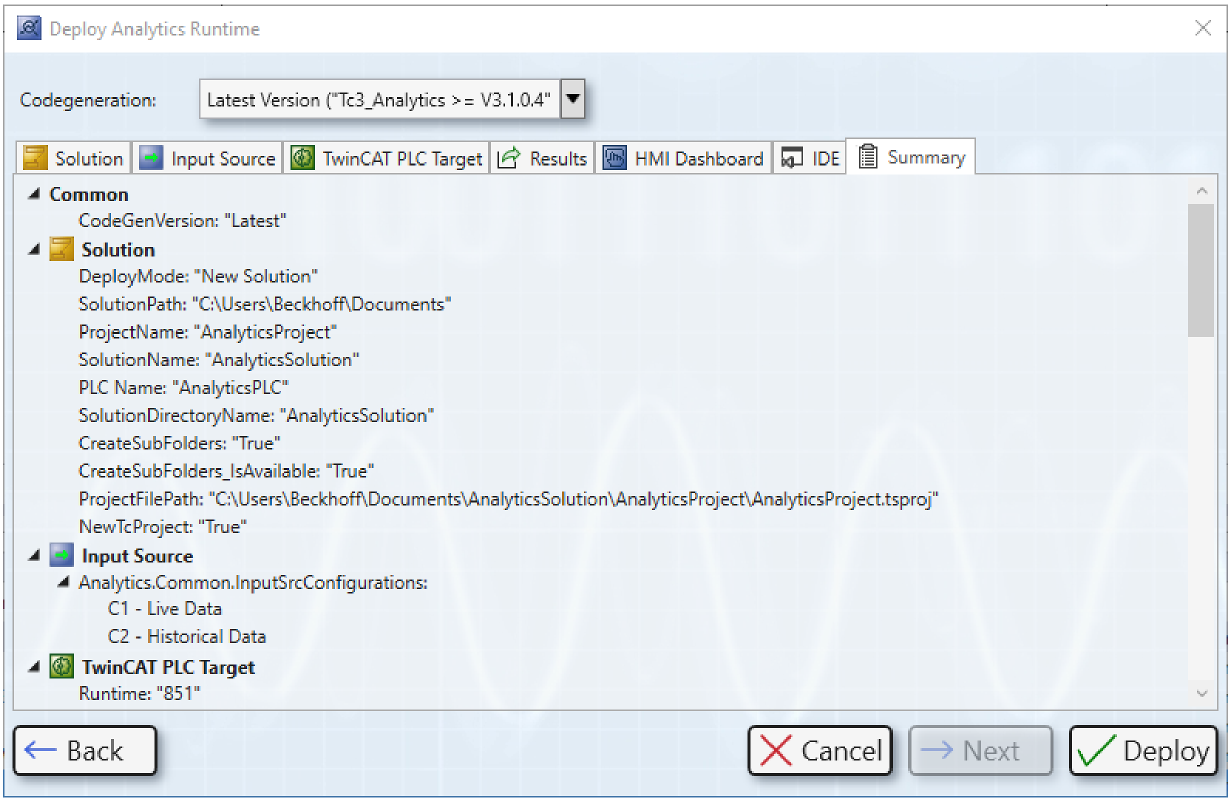
☑ Create Start-Page ☑ Show current time

Dashboard Styles

Dashboard Layout: Allign Center

Dashboard Sorting: Control Type

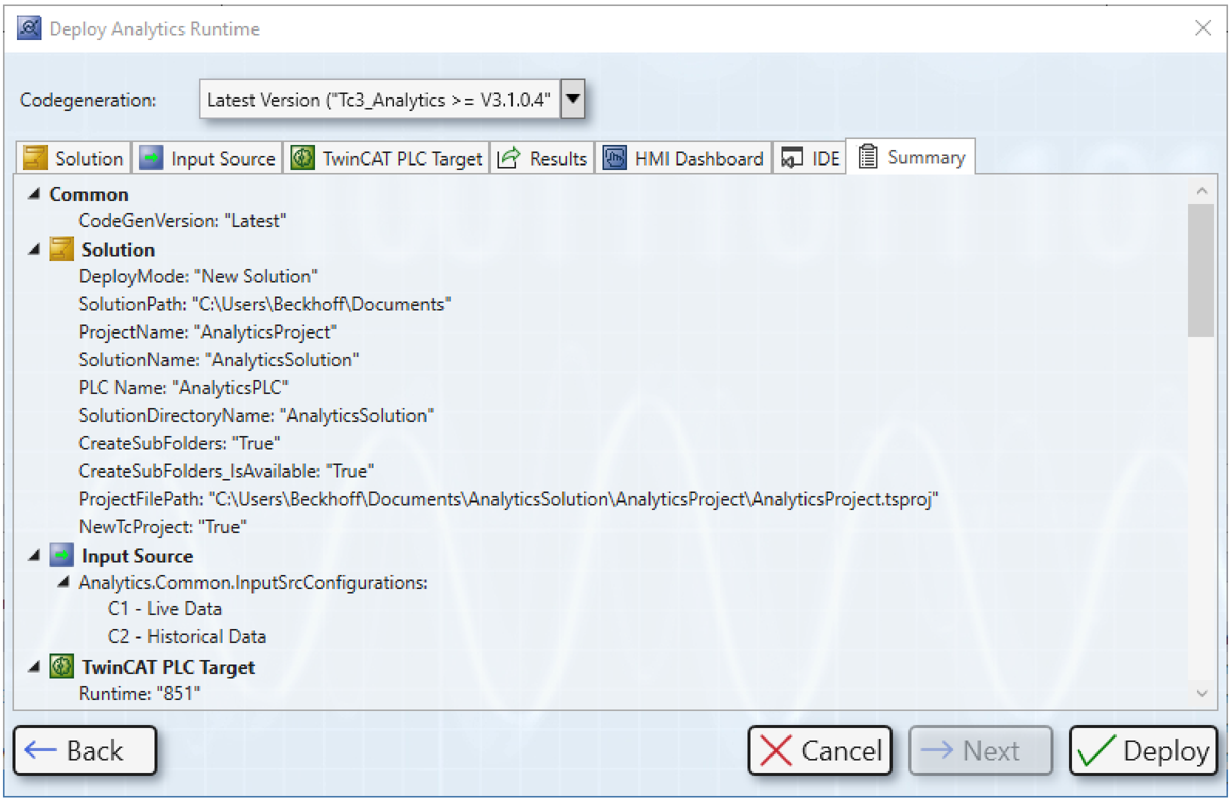
7. On the **IDE** tab, select which Visual Studio® version or TwinCAT XAE Shell is to be used for the generation if several are installed.



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4")

```
◢ Common
    CodeGenVersion: "Latest"
◢ Solution
    DeployMode: "New Solution"
    SolutionPath: "C:\Users\Beckhoff\Documents"
    ProjectName: "AnalyticsProject"
    SolutionName: "AnalyticsSolution"
    PLC Name: "AnalyticsPLC"
    SolutionDirectoryName: "AnalyticsSolution"
    CreateSubFolders: "True"
    CreateSubFolders_IsAvailable: "True"
    ProjectFilePath: "C:\Users\Beckhoff\Documents\AnalyticsSolution\AnalyticsProject\AnalyticsProject.tsproj"
    NewTcProject: "True"
◢ Input Source
  ◢ Analytics.Common.InputSrcConfigurations:
        C1 - Live Data
        C2 - Historical Data
◢ TwinCAT PLC Target
    Runtime: "851"
```

8. The **Summary** tab shows you all the settings you have made for the generation. Click on **Deploy** to start the generation process.



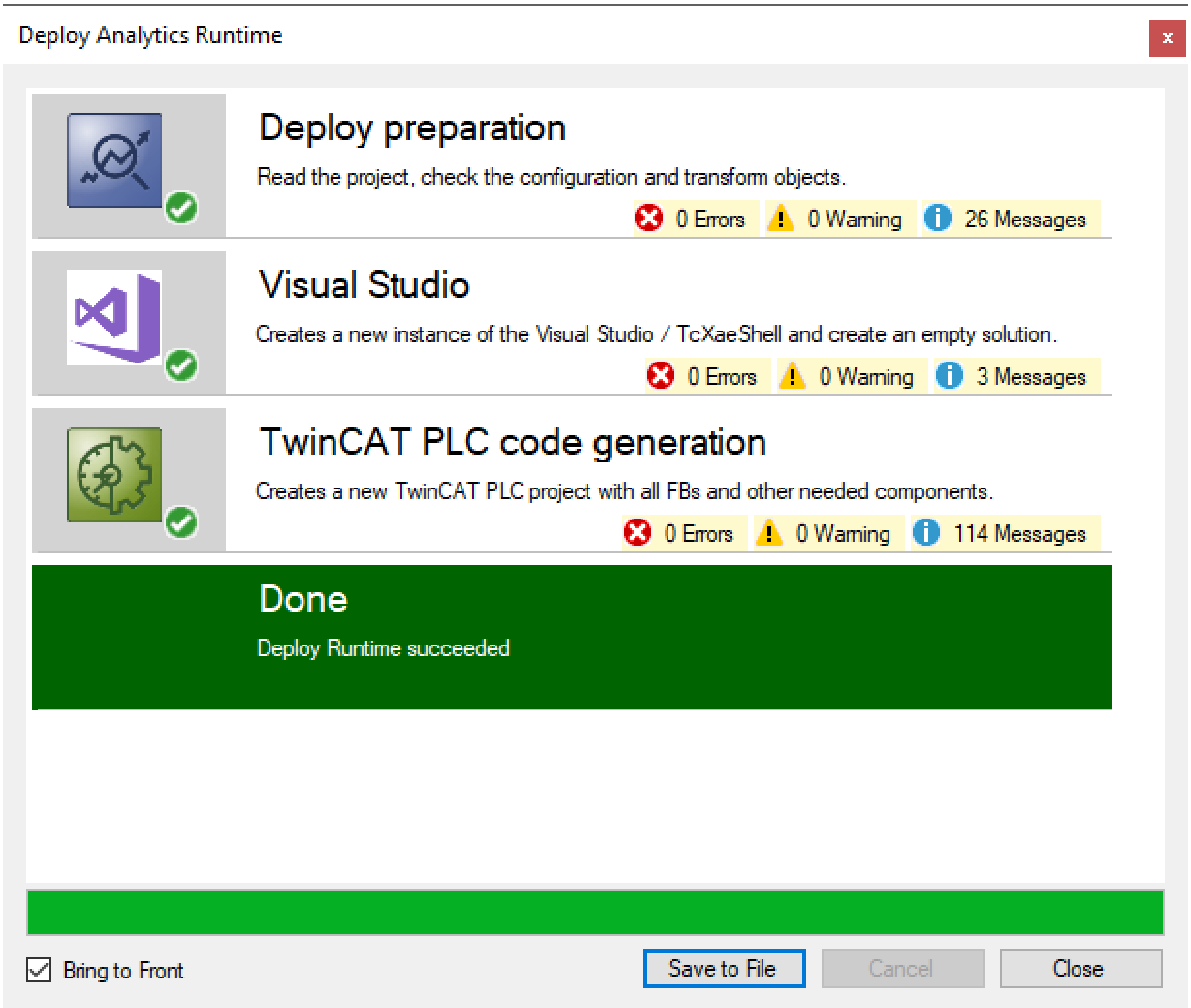
Common
CodeGenVersion: "Latest"

Solution
DeployMode: "New Solution"
SolutionPath: "C:\Users\Beckhoff\Documents"
ProjectName: "AnalyticsProject"
SolutionName: "AnalyticsSolution"
PLC Name: "AnalyticsPLC"
SolutionDirectoryName: "AnalyticsSolution"
CreateSubFolders: "True"
CreateSubFolders_IsAvailable: "True"
ProjectFilePath: "C:\Users\Beckhoff\Documents\AnalyticsSolution\AnalyticsProject\AnalyticsProject.tsproj"
NewTcProject: "True"

Input Source
Analytics.Common.InputSrcConfigurations:
C1 - Live Data
C2 - Historical Data

TwinCAT PLC Target
Runtime: "851"

⇨ The overview window shows each step during the generation process, clearly arranged and divided into categories.



Deploy Analytics Runtime

Deploy preparation

Read the project, check the configuration and transform objects.

0 Errors 0 Warning 26 Messages

Visual Studio

Creates a new instance of the Visual Studio / TcXaeShell and create an empty solution.

0 Errors 0 Warning 3 Messages

TwinCAT PLC code generation

Creates a new TwinCAT PLC project with all FBs and other needed components.

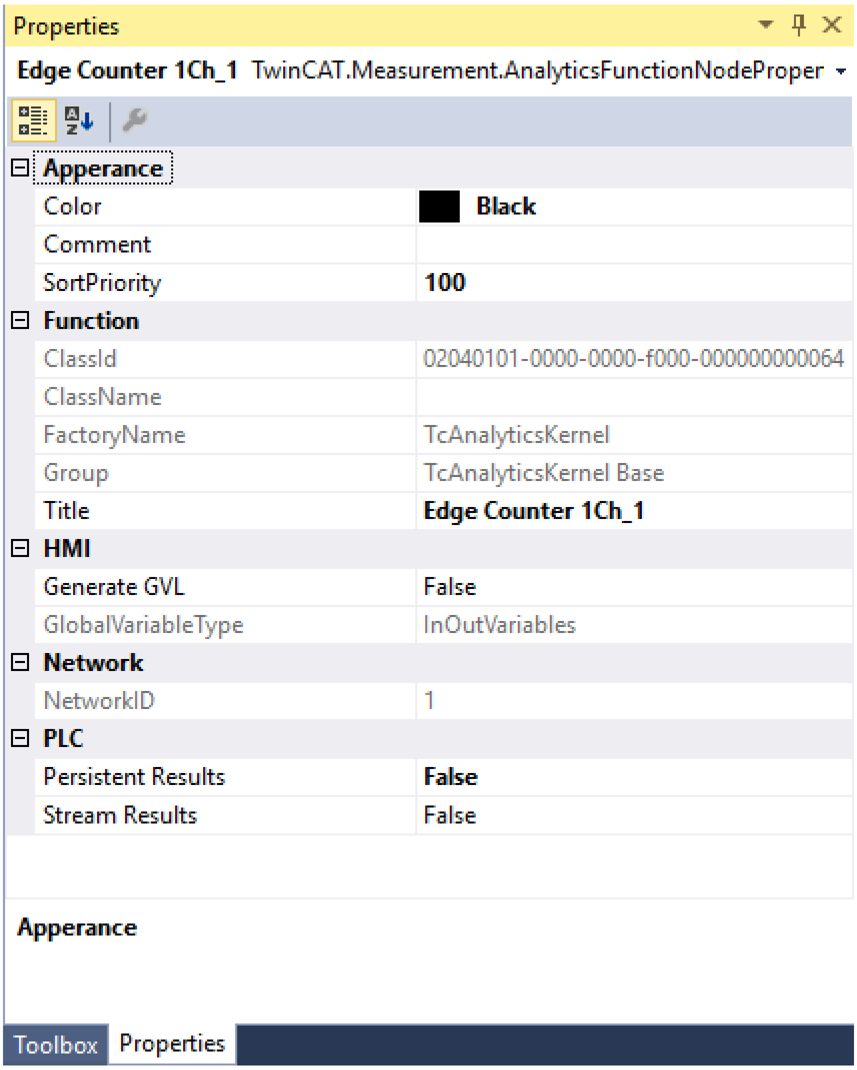
0 Errors 0 Warning 114 Messages

Done

Deploy Runtime succeeded

6.7.1 Algorithm properties

Each algorithm of the Analytics Configurator is providing some properties. The sections of HMI and PLC are necessary for the automatic code generation.



Properties

Edge Counter 1Ch_1 TwinCAT.Measurement.AnalyticsFunctionNodeProper

Apperance	
Color	■ **Black**
Comment	
SortPriority	**100**
Function	
ClassId	02040101-0000-0000-f000-000000000064
ClassName	
FactoryName	TcAnalyticsKernel
Group	TcAnalyticsKernel Base
Title	**Edge Counter 1Ch_1**
HMI	
Generate GVL	False
GlobalVariableType	InOutVariables
Network	
NetworkID	1
PLC	
Persistent Results	**False**
Stream Results	False

Apperance

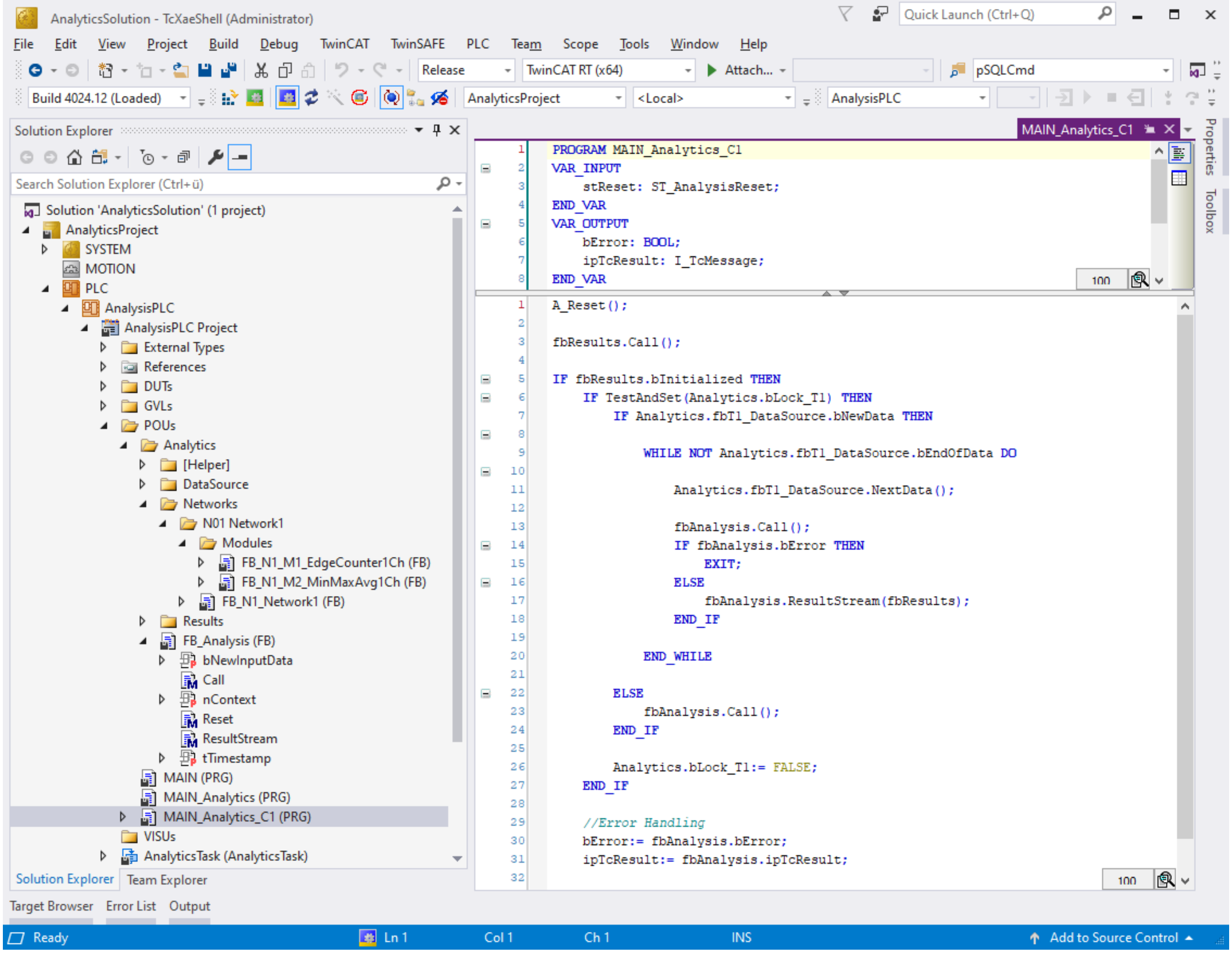
HMI

- **Generate GVL:** Enable the generation of an Global Variable List with a collection of variables and corresponding data type mapping for TwinCAT HMI
- **GlobalVariableType:** Choose the type with InOutVariables just for inputs and outputs of the algorithm or KeyValuePairs for general mapping to STRING for tables
-

PLC

- **Persistent Results:** Enable this flag to store results of algorithm persistent to target system of the Analytics Runtime
- **Stream Results:** Enable this flag to add the In- and Outputs of the algorithm to a result stream which will be generated by the code generation

6.7.2 PLC Code



```
PROGRAM MAIN_Analytics_C1
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

```
A_Reset();

fbResults.Call();

IF fbResults.bInitialized THEN
    IF TestAndSet(Analytics.bLock_T1) THEN
        IF Analytics.fbT1_DataSource.bNewData THEN

            WHILE NOT Analytics.fbT1_DataSource.bEndOfData DO

                Analytics.fbT1_DataSource.NextData();

                fbAnalysis.Call();
                IF fbAnalysis.bError THEN
                    EXIT;
                ELSE
                    fbAnalysis.ResultStream(fbResults);
                END_IF

            END_WHILE

        ELSE
            fbAnalysis.Call();
        END_IF

        Analytics.bLock_T1:= FALSE;
    END_IF

    //Error Handling
    bError:= fbAnalysis.bError;
    ipTcResult:= fbAnalysis.ipTcResult;
```

NOTICE

Compatibility of automatically generated PLC code

The automatically generated PLC code is based on the TwinCAT Analytics Library. Interfaces of the base models of the library are compatible with earlier versions after release of the library. The automatically generated code itself is only a sample code! This code generation may change from version to version.

Code generation version compatibility

	Version 2.1 (obsolete)	Version 3.0 (obsolete)	Version 4.0	Version 4.1	Version 5.0
Analytics algorithms	(X)	(X)	X	X	X
Filter algorithms	(X)	(X)	X	X	X
HMI support	(X)	(X)	X	X	X
HMI with advanced input stream handling	-	(X)	X	X	X
Support of array inputs	(X)	(X)	X	X	X
Support of oversampling inputs	-	-	X	X	X
Network templates	(X)	(X)	X	X	X
"Closed" network templates	-	-	X	X	X
Network with inputs, outputs and parameters	-	-	X	X	X
"Condition Monitoring" network templates	-	-	X	X	X
Lambda algorithms	-	-	X	X	X
Writing parameters via HMI	-	-	-	X	X
"Power Monitoring" algorithms	-	-	X	X	X
Scope Support	-	-	-	-	X

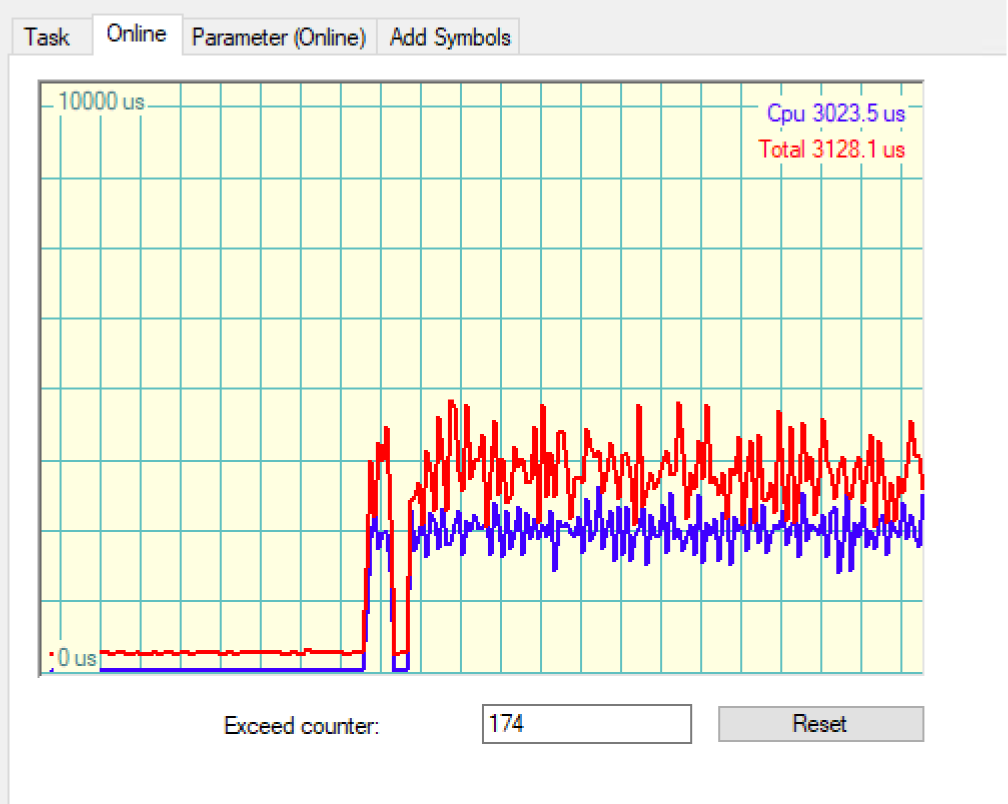
NOTICE

Possible manual PLC code optimizations

Every automatically generated PLC project has specific properties and requirements for smooth and error-free execution. Some of these properties cannot be predicted at the time of generation. To optimize the execution, various places are listed under Manual optimizations [▶ 363] where the generated code can be manually adapted.

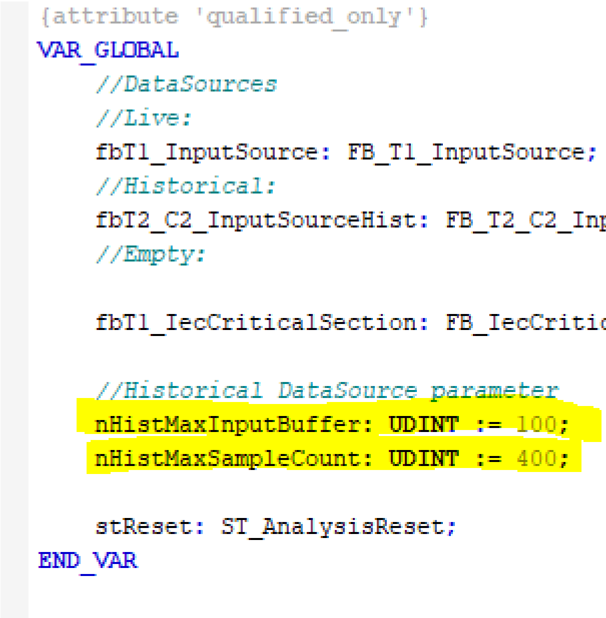
6.7.2.1 Manual optimizations

Cycle overruns in the processing of historical data



Exceed counter: 174

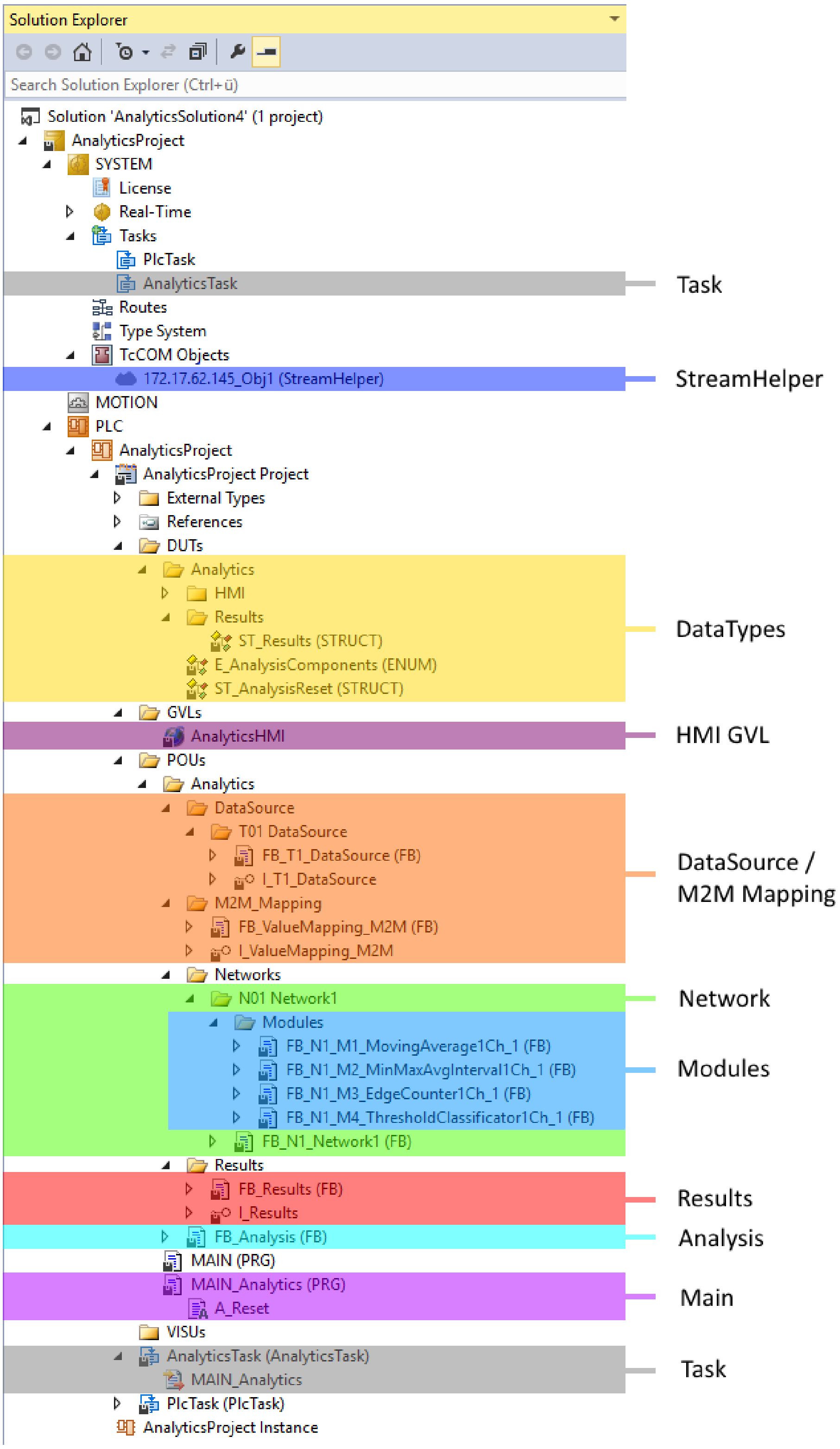
This can be optimized using the following parameters in Analytics.GVL



```
{attribute 'qualified_only'}
VAR_GLOBAL
    //DataSources
    //Live:
    fbT1_InputSource: FB_T1_InputSource;
    //Historical:
    fbT2_C2_InputSourceHist: FB_T2_C2_InputSourceHistorical;
    //Empty:

    fbT1_IecCriticalSection: FB_IecCriticalSection;

    //Historical DataSource parameter
    nHistMaxInputBuffer: UDINT := 100;
    nHistMaxSampleCount: UDINT := 400;

    stReset: ST_AnalysisReset;
END_VAR
```

- The parameter nHistMaxInputBuffer specifies the maximum number of samples to be processed per cycle. The higher this value, the greater the risk of cycle overruns. If this value is too small, the StreamHelper buffer may overflow, resulting in data loss.
- The parameter nHistMaxSampleCount specifies how many samples a message from the ASP may contain. The higher the value, the faster the data of a historical recording is transmitted. However, this can also cause the StreamHelper buffer to overflow more quickly

6.7.2.2 Code version 2.1



Task:

A separate task is created for the analytics analysis.

StreamHelper:

If one or more data sources are of type MQTT binary stream, the code generation creates an instance of a StreamHelper object to process the incoming binary stream patterns.

DataTypes:

The data types are created for the analysis. They contain STRUCTs for the reset function or result processing and ENUMs to select the various components.

HMI GVL:

To conveniently map module inputs and outputs with the HMI dashboard, selected variables are generated as global variables.

DataSource/M2M Mapping:

The FB DataSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs. The FB ValueMapping_M2M manages the value mapping between the modules (M2M - Module to Module) from the module INPUTs to the module OUTPUTs.

Network:

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

Modules:

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

Results:

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker or saves the data to the Analytics binary file.

Analysis:

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

MAIN:

The FB Analysis is called in the program MAIN_Analytics. The program is assigned to the separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics Version >= 3.1.0.0

6.7.2.2.1 FB_DataSource

The DataSource FB manage the receiving of the input values of the different sources. In the OUTPUT declaration you can find all configured inputs.

Syntax

Definition:

```
FUNCTION_BLOCK FB_T[n]_DataSource IMPLEMENTS I_T[n]_DataSource
VAR
END_VAR
```

Methods

Name	Definition location	Description
Call [▸ 367]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
GetData [▸ 367]	Local	Method to get the data of the specified element

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.2.1.1 GetData

Syntax

```
METHOD GetData : BOOL
VAR_INPUT
    nElement : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nElement	UDINT	Element ID to get the specific sample

Return value

Name	Type	Description
GetData	BOOL	Is TRUE if a new element is selected

6.7.2.2.1.2 Call

Syntax

```
METHOD Call : BOOL
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.2.2 FB_Network

All modules are sorted in a specific network to get a better overview and structure of the configured analysis.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_INPUT
    [module FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Inputs

Name	Type	Description
Module FBs		FBs of the configured modules

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE as soon as an error situation occurs.
ipTcResult	I_TcMessage	Message interface from the TwinCAT 3 EventLogger, which provides details on the return value.

Methods

Name	Definition location	Description
Call [▸ 372]	Local	Method for background communication. The method must be called cyclically.
Reset [▸ 372]	Local	Reset the Network with all sub modules
ValueMapping [▸ 368]	Local	Map the input values to the different module inputs
SetHMIValues [▸ 369]	Local	Optional: Map in- outputs of the modules to the global HMI variable

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.2.2.1 ValueMapping

Syntax

```
METHOD ValueMapping : BOOL
VAR_INPUT
    ipDataSource : I_T[n]_DataSource;
END_VAR
```

Inputs

Name	Type	Description
ipDataSource	I_T[n]_DataSource	Data for the analysis from the specific data source

Return value

Name	Type	Description
ValueMapping	BOOL	

6.7.2.2.2.2 SetHMIValues

Syntax

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to global HMI struct

Return value

Name	Type	Description
SetHMIValues	BOOL	Is TRUE if done

6.7.2.2.2.3 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE if done

6.7.2.2.2.4 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipDataSource: I_T[n]_DataSource;
    [ipValueMapping_M2M: I_ValueMapping_M2M;]
END_VAR
```

Inputs

Name	Type	Description
ipDataSource	I_T[n]_DataSource	Data for the analysis.
ipValueMapping_M2M	I_ValueMapping_M2M	Optional: Needed for mapping values between modules

Return value

Name	Type	Description
Call	BOOL	

6.7.2.2.3 FB_Module

The module FBs contains all inputs and outputs of the configured modules from the Workbench Configurator. It is also possible to reconfigure the modules at runtime. You only have to change the parameter and then start the reconfigure process with a rising edge at the bReconfigure INPUT.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_M[n]_[Module]
VAR_INPUT
    [module inputs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    [module outputs]
END_VAR
```

Inputs

Name	Type	Description
Module inputs		Inputs of the selected module

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE as soon as an error situation occurs.
ipTcResult	I_TcMessage	Message interface from the TwinCAT 3 EventLogger, which provides details on the return value.
Module outputs		Outputs of the selected module

Methods

Name	Definition location	Description
Call [▶ 370]	Local	Method for background communication. The method must be called cyclically.
Reset [▶ 371]	Local	Reset the module
SetHMI [▶ 371]	Local	Optional: Sets the in- outputs to the global HMI structs

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.2.3.1 Call

Syntax

```
METHOD Call : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.2.3.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE if done

6.7.2.2.3.3 SetHMI

Syntax

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to global HMI struct

Return value

Name	Type	Description
SetHMI	BOOL	Is TRUE if done

6.7.2.2.4 FB_Analysis

In the analysis FB the whole analytics routine is defined. All configured networks with their modules and error handling is created.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Analysis
VAR_INPUT
    [network FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Inputs

Name	Type	Description
Network FBs		FBs of the configured networks

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE as soon as an error situation occurs.
ipTcResult	I_TcMessage	Message interface from the TwinCAT 3 EventLogger, which provides details on the return value.

Methods

Name	Definition location	Description
Call [▸ 372]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
Reset [▸ 372]	Local	Reset the whole analysis
ResultStream [▸ 373]	Local	Optional: If a result stream has to be created

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.2.4.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipDataSource: I_T[n]_DataSource;
END_VAR
```

Inputs

Name	Type	Description
ipDataSource	I_T[n]_DataSource	Data for the analysis.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.2.4.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

Inputs

Name	Type	Description
stReset	ST_AnalysisReset	Struct to define witch module or network should be reset.

Return value

Name	Type	Description
Reset	BOOL	Is TRUE if done

6.7.2.2.4.3 ResultStream

Syntax

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

Inputs

Name	Type	Description
ipResults	I_Results	Interface pointer to the Result FB

Return value

Name	Type	Description
Call	BOOL	

6.7.2.2.5 FB_Results

If results of the analysis has to be stored or streamed, the result FB managed this and streamed the selected variables to the message broker or store the data into the analytics binary file.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    stResults: ST_Results;
END_VAR
```

Outputs

Name	Type	Description
stResults	ST_Results	Result struct which contains all items of the result stream

Methods

Name	Definition location	Description
Call [▶ 374]	Local	Method for background communication. The method must be called cyclically.
AddResult [▶ 374]	Local	Add a sample to the result stream
SendResults [▶ 374]	Local	Sends all buffered samples of the result stream
Release [▶ 375]	Local	Close stream or file of the result stream

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.2.5.1 Call

Syntax

```
METHOD Call : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.2.5.2 AddResult

Syntax

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

Inputs

Name	Type	Description
tTimestamp	ULINT	Timestamp of the sample
stSample	ST_Results	Sample struct

Return value

Name	Type	Description
AddResult	BOOL	

6.7.2.2.5.3 SendResults

Syntax

```
METHOD SendResults : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
SendResults	BOOL	

6.7.2.2.5.4 Release

Syntax

```
METHOD Release : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Release	BOOL	

6.7.2.2.6 MAIN_Analytics

In the MAIN_Analytics program the analysis FB is called. The program is assign to the separated Task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First you have to choose the component you would like to reset. Then a rising edge at the bReset INPUT starts the reset process.

Inside of the A_Reset Action all reset calls are defined.

Syntax

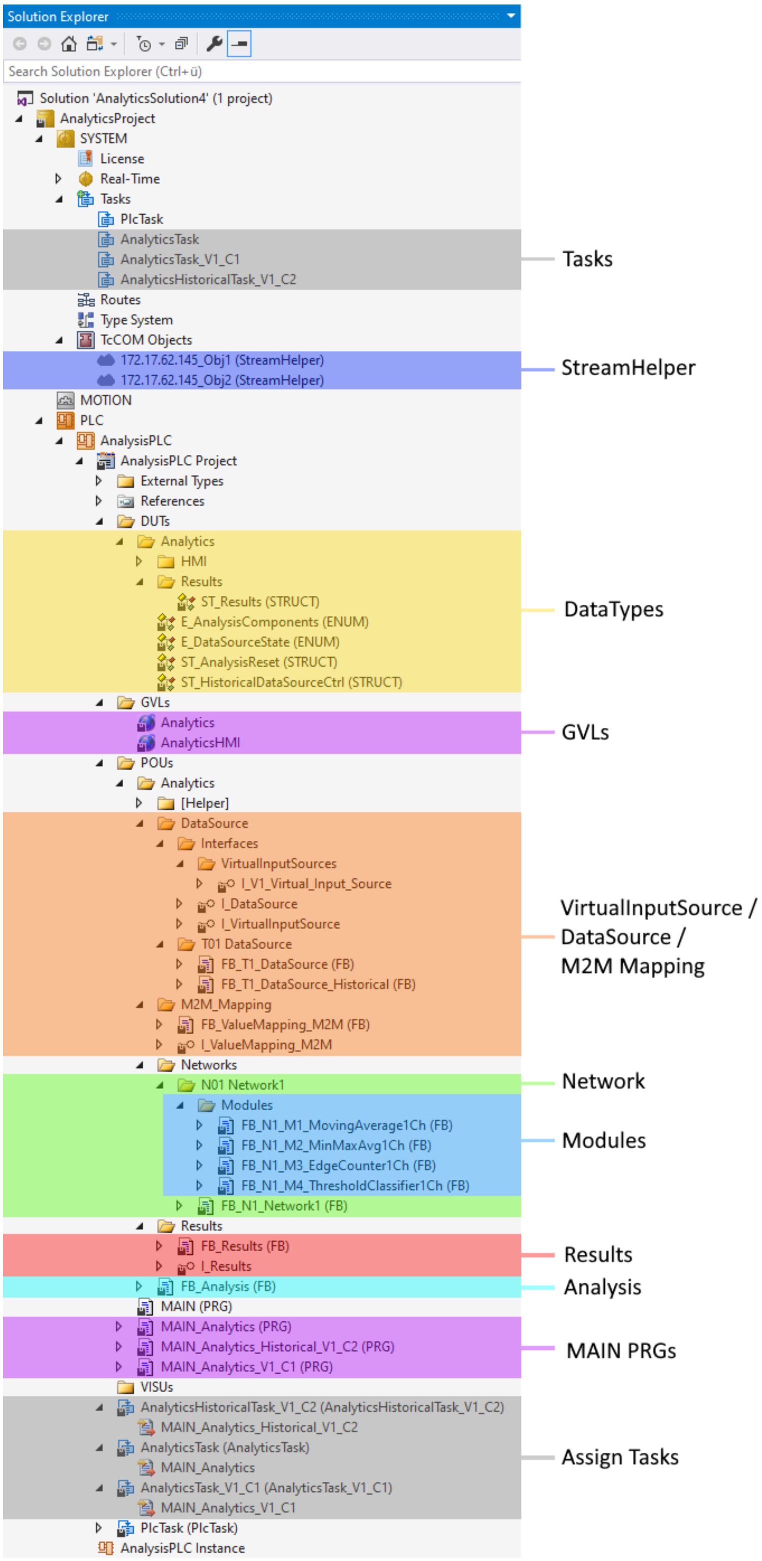
Definition:

```
PROGRAM MAIN_Analytics
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3 Code version 3.0



```
Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'AnalyticsSolution4' (1 project)
  AnalyticsProject
    SYSTEM
      License
      Real-Time
      Tasks
        PlcTask
        AnalyticsTask
        AnalyticsTask_V1_C1
        AnalyticsHistoricalTask_V1_C2
      Routes
      Type System
      TcCOM Objects
        172.17.62.145_Obj1 (StreamHelper)
        172.17.62.145_Obj2 (StreamHelper)
    MOTION
    PLC
      AnalysisPLC
        AnalysisPLC Project
          External Types
          References
          DUTs
            Analytics
              HMI
              Results
                ST_Results (STRUCT)
              E_AnalysisComponents (ENUM)
              E_DataSourceState (ENUM)
              ST_AnalysisReset (STRUCT)
              ST_HistoricalDataSourceCtrl (STRUCT)
          GVLs
            Analytics
            AnalyticsHMI
          POUs
            Analytics
              [Helper]
              DataSource
                Interfaces
                  VirtualInputSources
                    I_V1_Virtual_Input_Source
                  I_DataSource
                  I_VirtualInputSource
                T01 DataSource
                  FB_T1_DataSource (FB)
                  FB_T1_DataSource_Historical (FB)
              M2M_Mapping
                FB_ValueMapping_M2M (FB)
                I_ValueMapping_M2M
              Networks
                N01 Network1
                  Modules
                    FB_N1_M1_MovingAverage1Ch (FB)
                    FB_N1_M2_MinMaxAvg1Ch (FB)
                    FB_N1_M3_EdgeCounter1Ch (FB)
                    FB_N1_M4_ThresholdClassifier1Ch (FB)
                  FB_N1_Network1 (FB)
              Results
                FB_Results (FB)
                I_Results
              FB_Analysis (FB)
            MAIN (PRG)
            MAIN_Analytics (PRG)
            MAIN_Analytics_Historical_V1_C2 (PRG)
            MAIN_Analytics_V1_C1 (PRG)
          VISUs
          AnalyticsHistoricalTask_V1_C2 (AnalyticsHistoricalTask_V1_C2)
            MAIN_Analytics_Historical_V1_C2
          AnalyticsTask (AnalyticsTask)
            MAIN_Analytics
          AnalyticsTask_V1_C1 (AnalyticsTask_V1_C1)
            MAIN_Analytics_V1_C1
          PlcTask (PlcTask)
        AnalysisPLC Instance
```

Tasks:

A separate task is created for the analytics analysis and for each configuration of a Virtual Input Source.

StreamHelper:

For each data source of type MQTT binary stream, the code generation creates an instance of a stream helper object to process the incoming binary stream patterns.

DataTypes:

The data types are created for the analysis. They contain STRUCTs for the reset function or result processing and ENUMs to select the various components.

GVLs:

To conveniently map module inputs and outputs with the HMI dashboard, selected variables are generated as global variables. In addition, the Data Source function block instances and various parameters are generated as global variables.

VirtualInputSource / DataSource / M2M Mapping:

The Virtual Input Source interfaces abstract the Data Source symbols from the analysis. The FB DataSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs. The FB ValueMapping_M2M manages the value mapping between the modules (M2M - Module to Module) from the module INPUTs to the module OUTPUTs.

Network:

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

Modules:

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

Results:

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker in binary or Json format, or saves the data locally to an Analytics binary file.

Analysis:

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

MAIN PRGs:

In the MAIN_Analytics program, the DataSource FBs are called, the reset function is managed and, if appropriate, the values are mapped with the HMI dashboard. The program is assigned to a separate task.

The FB Analysis is called in the programs MAIN_Analytics_Vx_Cx. The programs are each assigned to a separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics Version >= 3.1.0.0

6.7.2.3.1 FB_DataSource

The FB DataSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs.

Syntax

Definition:

```
FUNCTION_BLOCK FB_T[n]_DataSource IMPLEMENTS I_DataSource, I_V[n]_Virtual_Input_Source
VAR_OUTPUT
eDataState: E_DataSourceState;
END_VAR
```

Methods

Name	Definition location	Description
Call [▸ 378]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
GetData [▸ 378]	Local	Method to retrieve the data of the specified element.
GetDataOversampling [▸ 379]	Local	Method to retrieve the oversampling data of the specified element.
NewDataAvailable [▸ 379]	Local	Method to check if new data is available.
HistoricalCtrl [▸ 379]	Local	Method for retrieving historical data.
UpdateRecordList [▸ 380]	Local	Method for updating the historical record list.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3.1.1 GetData

Syntax

```
METHOD GetData : BOOL
VAR_INPUT
    nElement : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nElement	UDINT	Element ID to obtain the specific example.

Return value

Name	Type	Description
GetData	BOOL	Is TRUE if a new element is selected.

6.7.2.3.1.2 Call

Syntax

```
METHOD Call : BOOL
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.3.1.3 GetDataOversampling

Syntax

```
METHOD GetDataOversampling : BOOL
VAR_INPUT
nElement : UDINT;
nSample: : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nElement	UDINT	Element ID to obtain the specific sample element.
nSample	UDINT	Sample ID to obtain the specific sample.

Return value

Name	Type	Description
GetDataOversampling	BOOL	Is TRUE if a new element is selected.

6.7.2.3.1.4 NewDataAvailable

Syntax

```
METHOD NewDataAvailable : BOOL
VAR_INPUT
    nLastDataHandle : ULINT;
END_VAR
```

Inputs

Name	Type	Description
nLastDataHandle	ULINT	Handle from the last fetched data packet.

Return value

Name	Type	Description
NewDataAvailable	BOOL	Is TRUE if new data is available.

6.7.2.3.1.5 HistoricalCtrl

Syntax

```
METHOD HistoricalCtrl : BOOL
VAR_INPUT
    stCtrl : REFERENCE TO ST_HMI_DataSourceCtrl;
    stHistStreamInfo : REFERENCE TO ST_HMI_DataSourceHist;
    stRecordInfo : REFERENCE TO ST_HMI_DataSourceHistRecordInfo;
END_VAR
```

Inputs

Name	Type	Description
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
stRecordInfo	ST_HMI_DataSourceHistRecordInfo	

Return value

Name	Type	Description
HistoricalCtrl	BOOL	

6.7.2.3.1.6 UpdateRecordList

Syntax

```
METHOD UpdateRecordList : BOOL
VAR_INPUT
    stCtrl : REFERENCE TO ST_HMI_DataSourceCtrl;
    stHistStreamInfo : REFERENCE TO ST_HMI_DataSourceHist;
    sStreamSystemID : GUID;
END_VAR
```

Inputs

Name	Type	Description
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
sStreamSystemID	GUID	

Return value

Name	Type	Description
UpdateRecordList	BOOL	

6.7.2.3.2 FB_Network

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_INPUT
    [module FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Inputs

Name	Type	Description
Module FBs		Function blocks of the configured modules.

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.

Methods

Name	Definition location	Description
Call [▸ 385]	Local	Method for background communication. The method must be called cyclically.
Reset [▸ 385]	Local	Resetting the network with all submodules.
ValueMapping [▸ 381]	Local	Assignment of the input values to the various module inputs.
SetHMIValues [▸ 381]	Local	Optional: Mapping of the inputs/outputs of the modules to the global HMI variable.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3.2.1 ValueMapping

Syntax

```
METHOD ValueMapping : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for analysis from the specific data source.

Return value

Name	Type	Description
ValueMapping	BOOL	

6.7.2.3.2.2 SetHMIValues

Syntax

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to the global HMI structure.

Return value

Name	Type	Description
SetHMIValues	BOOL	Is TRUE when completed.

6.7.2.3.2.3 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.3.2.4 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M: I_ValueMapping_M2M;]
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_T[n]_Virtual_Input_Source	Data for the analysis.
ipValueMapping_M2M	I_ValueMapping_M2M	Optional: Necessary for mapping values between modules.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.3.3 FB_Module

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_M[n]_[Module]
VAR_INPUT
    [module inputs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    [module outputs]
END_VAR
```

Inputs

Name	Type	Description
Module inputs		Inputs of the selected module.

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.
Module outputs		Outputs of the selected module.

Methods

Name	Definition location	Description
Call [▶ 383]	Local	Method for background communication. The method must be called cyclically.
Reset [▶ 383]	Local	Resetting the module.
SetHMI [▶ 384]	Local	Optional: Sets the inputs/outputs to the global HMI structures.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3.3.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M : I_ValueMapping_M2M;]
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for the analysis.
ipValueMapping_M2M	I_ValueMapping_M2M	Optional: Necessary for mapping values between modules.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.3.3.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.3.3.3 SetHMI

Syntax

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to the global HMI structure.

Return value

Name	Type	Description
SetHMI	BOOL	Is TRUE when completed.

6.7.2.3.4 FB_Analysis

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Analysis
VAR_INPUT
    [network FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Inputs

Name	Type	Description
Network FBs		Function blocks of the configured networks.

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.

Methods

Name	Definition location	Description
Call [▶ 385]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
Reset [▶ 385]	Local	Resets the entire analysis.
ResultStream [▶ 386]	Local	Optional: If a result stream needs to be created.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3.4.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for the analysis.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.3.4.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

Inputs

Name	Type	Description
stReset	ST_AnalysisReset	Structure to define which module or network is to be reset.

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.3.4.3 ResultStream

Syntax

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

Inputs

Name	Type	Description
ipResults	I_Results	Interface pointer to the FB Results.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.3.5 FB_Results

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker or saves the data to the Analytics binary file.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    stResults: ST_Results;
END_VAR
```

Outputs

Name	Type	Description
stResults	ST_Results	Result structure that contains all elements of the result stream.

Methods

Name	Definition location	Description
Call [▸ 387]	Local	Method for background communication. The method must be called cyclically.
AddResult [▸ 387]	Local	Add a sample to the result stream
SendResults [▸ 387]	Local	Sends all buffered samples of the result stream
Release [▸ 387]	Local	Close stream or file of the result stream

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3.5.1 Call

Syntax

```
METHOD Call : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.3.5.2 AddResult

Syntax

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

Inputs

Name	Type	Description
tTimestamp	ULINT	Timestamp of the sample
stSample	ST_Results	Sample structure

Return value

Name	Type	Description
AddResult	BOOL	

6.7.2.3.5.3 SendResults

Syntax

```
METHOD SendResults : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
SendResults	BOOL	

6.7.2.3.5.4 Release

Syntax

```
METHOD Release : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Release	BOOL	

6.7.2.3.6 MAIN_Analytics

In the MAIN_Analytics program, the DataSource FBs are called, the reset function is managed and, if appropriate, the values are mapped with the HMI Dashboard. The program is assigned to a separate task.

Syntax

Definition:

```
PROGRAM MAIN_Analytics
VAR_INPUT
    sCurrentStreamSystemID: GUID;
    stHistStreamInfo: REFERENCE TO ST_HMI_DataSourceHist;
    stHistRecordInfo: REFERENCE TO ST_HMI_DataSourceHistRecordInfo;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.3.7 MAIN_Analytics_V[n]_C[n]

The FB Analysis is called in the program MAIN_Analytics_V[n]_C[n]. The program is assigned to the separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

Mapping of the HMI values takes place in the action A_MapToHMI.

Syntax

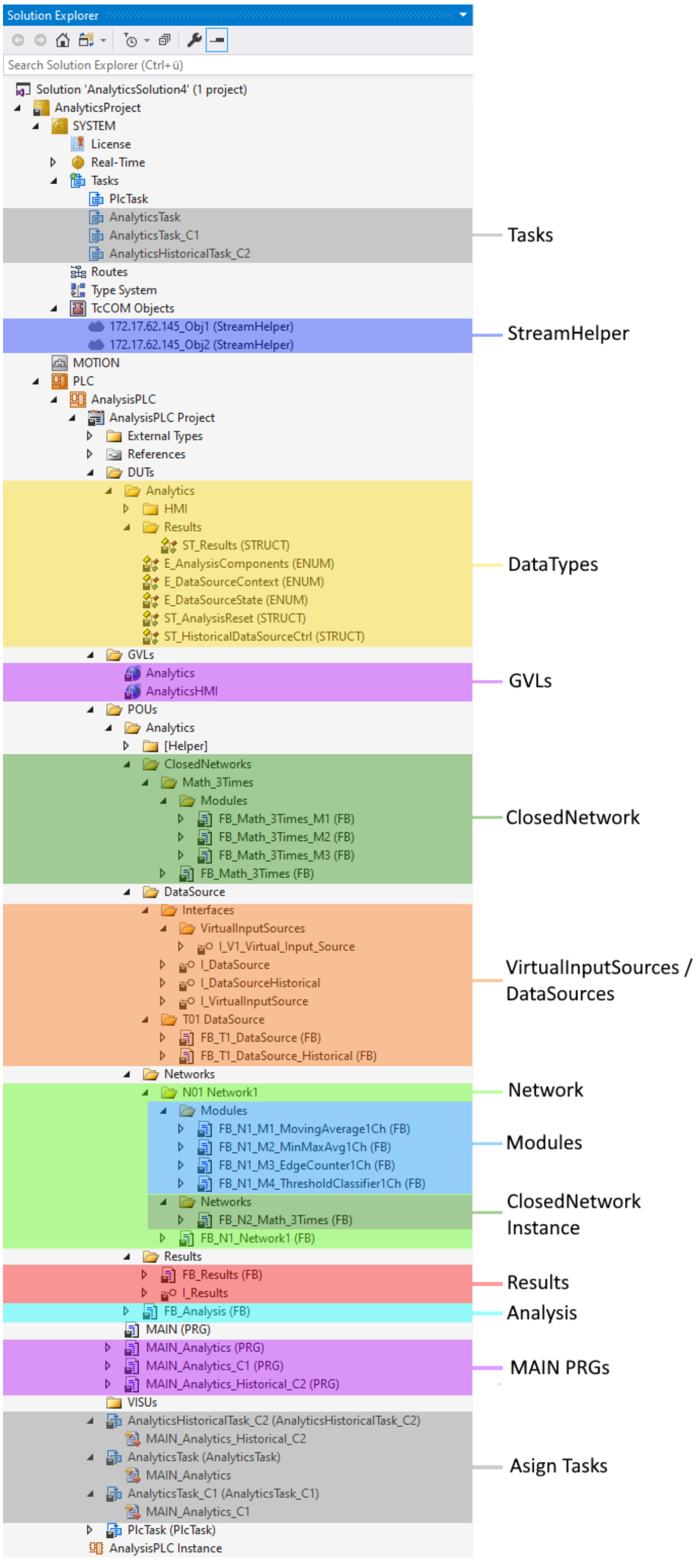
Definition:

```
PROGRAM MAIN_Analytics_V[n]_C[n]
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    nLastDataHandle: ULINT;
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4 Code version 4.x



Tasks:

A separate task is created for the analytics analysis and for each configuration of a Virtual Input Source.

StreamHelper:

For each data source of type MQTT binary stream, the code generation creates an instance of a stream helper object to process the incoming binary stream patterns.

DataTypes:

The data types are created for the analysis. They contain STRUCTs for the reset function or result processing and ENUMs to select the various components.

GVLs:

To conveniently map module inputs and outputs with the HMI dashboard, selected variables are generated as global variables. In addition, the Data Source function block instances and various parameters are generated as global variables.

ClosedNetwork:

The ClosedNetwork FBs are generated once with all subnetworks and modules. They can be instantiated multiple times in the analysis. In this way, the generated code can be reduced and simplified.

VirtualInputSource / DataSource:

The VirtualInputSource interfaces abstract the DataSource symbols from the analysis. The FB DataSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs.

Network:

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

Modules:

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

ClosedNetwork Instance:

In this FB the corresponding ClosedNetwork is instantiated for the analysis. Internally used modules are no longer generated as module FBs in this case.

Results:

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker in binary or Json format, or saves the data locally to an Analytics binary file.

Analysis:

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

MAIN PRGs:

In the MAIN_Analytics program, the DataSource FBs are called, the reset function is managed and, if appropriate, the values are mapped with the HMI dashboard. The program is assigned to a separate task.

The FB Analysis is called in the programs MAIN_Analytics_Vx_Cx. The programs are each assigned to a separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics Version >= 3.1.0.0

6.7.2.4.1 FB_DataSource

The FB DataSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs.

Syntax

Definition:

```
FUNCTION_BLOCK FB_T[n]_DataSource IMPLEMENTS I_DataSource, I_V[n]_Virtual_Input_Source
VAR_OUTPUT
eDataState: E_DataSourceState;
END_VAR
```

Methods

Name	Definition location	Description
Call [▶ 392]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
GetData [▶ 391]	Local	Method to retrieve the data of the specified element.
GetDataOversampling [▶ 392]	Local	Method to retrieve the oversampling data of the specified element
NewDataAvailable [▶ 392]	Local	Method to check if new data is available.
HistoricalCtrl [▶ 392]	Local	Method for retrieving historical data
UpdateRecordList [▶ 393]	Local	Method for updating the historical record list.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4.1.1 GetData

Syntax

```
METHOD GetData : BOOL
VAR_INPUT
    nElement : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nElement	UDINT	Element ID to obtain the specific sample

Return value

Name	Type	Description
GetData	BOOL	Is TRUE if a new element is selected

6.7.2.4.1.2 Call

Syntax

```
METHOD Call : BOOL
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.4.1.3 GetDataOversampling

Syntax

```
METHOD GetDataOversampling : BOOL
VAR_INPUT
nElement : UDINT;
nSample: : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nElement	UDINT	Element ID to obtain the specific sample element
nSample	UDINT	Sample ID to obtain the specific sample

Return value

Name	Type	Description
GetDataOversampling	BOOL	Is TRUE if a new element is selected

6.7.2.4.1.4 NewDataAvailable

Syntax

```
METHOD NewDataAvailable : BOOL
VAR_INPUT
    nLastDataHandle : ULINT;
END_VAR
```

Inputs

Name	Type	Description
nLastDataHandle	ULINT	Handle of the last fetched data packet

Return value

Name	Type	Description
NewDataAvailable	BOOL	Is TRUE if new data is available

6.7.2.4.1.5 HistoricalCtrl

Syntax

```
METHOD HistoricalCtrl : BOOL
VAR_INPUT
    stCtrl : REFERENCE TO ST_HMI_DataSourceCtrl;
    stHistStreamInfo : REFERENCE TO ST_HMI_DataSourceHist;
    stRecordInfo : REFERENCE TO ST_HMI_DataSourceHistRecordInfo;
END_VAR
```

Inputs

Name	Type	Description
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
stRecordInfo	ST_HMI_DataSourceHistRecordInfo	

Return value

Name	Type	Description
HistoricalCtrl	BOOL	

6.7.2.4.1.6 UpdateRecordList

Syntax

```
METHOD UpdateRecordList : BOOL
VAR_INPUT
    stCtrl : REFERENCE TO ST_HMI_DataSourceCtrl;
    stHistStreamInfo : REFERENCE TO ST_HMI_DataSourceHist;
    sStreamSystemID : GUID;
END_VAR
```

Inputs

Name	Type	Description
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
sStreamSystemID	GUID	

Return value

Name	Type	Description
UpdateRecordList	BOOL	

6.7.2.4.1.7 NextData

Syntax

```
METHOD NextData : BOOL
VAR_INPUT
END_VAR
```

6.7.2.4.1.8 NextDataOversample

Syntax

```
METHOD GetDataOversampling : BOOL
VAR_INPUT
nMaxOversampling : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nMaxOversampling	UDINT	Specifies the maximum oversampling factor.

Return value

Name	Type	Description
NextDataOversample	BOOL	Is TRUE if a new element is selected

6.7.2.4.2 FB_Network

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
VAR
    [module FBs]
END_VAR
```

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.

Methods

Name	Definition location	Description
Call [▸ 385]	Local	Method for background communication. The method must be called cyclically.
Reset [▸ 385]	Local	Resetting the network with all submodules.
ValueMapping [▸ 395]	Local	Assignment of the input values to the various module inputs.
SetHMIValues [▸ 395]	Local	Optional: Mapping of the inputs/outputs of the modules to the global HMI variable.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4.2.1 ValueMapping

Syntax

```
METHOD ValueMapping : BOOL
VAR_INPUT
    pAnalysis : POINTER TO FB_Analysis;
END_VAR
```

Inputs

Name	Type	Description
pAnalysis	FB_Analysis	Instance of the analysis FB

Return value

Name	Type	Description
ValueMapping	BOOL	

6.7.2.4.2.2 SetHMIValues

Syntax

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to the global HMI structure

Return value

Name	Type	Description
SetHMIValues	BOOL	Is TRUE when completed

6.7.2.4.2.3 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pAnalysis: POINTER TO FB_Analysis;
END_VAR
```

Inputs

Name	Type	Description
pAnalysis	FB_Analysis	Instance of the analysis FB.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.4.2.4 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.4.3 FB_Module

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_M[n]_[Module]
VAR_INPUT
    [module inputs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    [module outputs]
END_VAR
```

Inputs

Name	Type	Description
Module inputs		Inputs of the selected module.

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.
Module outputs		Outputs of the selected module.

Methods

Name	Definition location	Description
Call [▸ 397]	Local	Method for background communication. The method must be called cyclically.
Reset [▸ 397]	Local	Resetting the module.
SetHMI [▸ 397]	Local	Sets the inputs/outputs to the global HMI structures.
GetHMI [▸ 398]	Local	Optional: Sets the inputs of the global HMI structures to the inputs of the module

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4.3.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M : I_ValueMapping_M2M;]
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for the analysis
ipValueMapping_M2M	I_ValueMapping_M2M	Optional: Necessary for mapping values between modules

Return value

Name	Type	Description
Call	BOOL	

6.7.2.4.3.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.4.3.3 SetHMI

Syntax

```
METHOD GetHMI : BOOL
VAR_INPUT
    nContent : DINT
    pContent : PVOID
    bHMIReinit : BOOL
END_VAR
```

```
VAR
    pHMI_C[n]_[Content] : POINTER TO ST_HMI_C[n]_[Content];
END_VAR
```

Inputs

Name	Type	Description
nContent	DINT	HMI Content Index
pContent	PVOID	Pointer to the HMI content structure
bHMIReinit	BOOL	Initialize the HMI content structure

Return value

Name	Type	Description
SetHMI	BOOL	Is TRUE when completed

6.7.2.4.3.4 GetHMI (4.1)

Syntax

```
METHOD GetHMI : BOOL
VAR_INPUT
    nContent :  DINT
    pContent :  PVOID
END_VAR
```

```
VAR
    pHMI_C[n]_[Content] : POINTER TO ST_HMI_C[n]_[Content];
END_VAR
```

Inputs

Name	Type	Description
nContent	DINT	HMI Content Index
pContent	PVOID	Pointer to the HMI content structure

Return value

Name	Type	Description
GetHMI	BOOL	Is TRUE when completed

6.7.2.4.4 FB_Analysis

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Analysis
VAR_INPUT
    ipV[n]_VirtualInputs: I_V[n]_Virtual_Input_Source;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
VAR
    [network FBs]
END_VAR
```

Inputs

Name	Type	Description
ipV[n]_VirtualInputs	I_V[n]_Virtual_Input_Source	Data for analysis from the specific data source

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.

Methods

Name	Definition location	Description
Call [▶ 399]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
Reset [▶ 399]	Local	Reset the whole analysis
ResultStream [▶ 400]	Local	Optional: If a result stream has to be created

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4.4.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for the analysis.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.4.4.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

Inputs

Name	Type	Description
stReset	ST_AnalysisReset	Structure to define which module or network is to be reset.

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.4.4.3 ResultStream

Syntax

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

Inputs

Name	Type	Description
ipResults	I_Results	Interface pointer to the FB Results

Return value

Name	Type	Description
Call	BOOL	

6.7.2.4.5 FB_Results

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker or saves the data to the Analytics binary file.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    stResults: ST_Results;
END_VAR
```

Outputs

Name	Type	Description
stResults	ST_Results	Result structure that contains all elements of the result stream.

Methods

Name	Definition location	Description
Call [▶ 401]	Local	Method for background communication. The method must be called cyclically.
AddResult [▶ 401]	Local	Add a sample to the result stream
SendResults [▶ 401]	Local	Sends all buffered samples of the result stream
Release [▶ 402]	Local	Close stream or file of the result stream

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4.5.1 Call

Syntax

```
METHOD Call : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.4.5.2 AddResult

Syntax

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

Inputs

Name	Type	Description
tTimestamp	ULINT	Timestamp of the sample
stSample	ST_Results	Sample structure

Return value

Name	Type	Description
AddResult	BOOL	

6.7.2.4.5.3 SendResults

Syntax

```
METHOD SendResults : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
SendResults	BOOL	

6.7.2.4.5.4 Release

Syntax

```
METHOD Release : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Release	BOOL	

6.7.2.4.6 MAIN_Analytics

In the MAIN_Analytics program, the DataSource FBs are called, the reset function is managed and, if appropriate, the values are mapped with the HMI Dashboard. The program is assigned to a separate task.

Syntax

Definition:

```
PROGRAM MAIN_Analytics
VAR_INPUT
    sCurrentStreamSystemID: GUID;
    stHistStreamInfo: REFERENCE TO ST_HMI_DataSourceHist;
    stHistRecordInfo: REFERENCE TO ST_HMI_DataSourceHistRecordInfo;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.4.7 MAIN_Analytics_C[n]

The FB Analysis is called in the program MAIN_Analytics_C[n]. The program is assigned to the separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

Mapping of the HMI values takes place in the action A_MapToHMI.

Syntax

Definition:

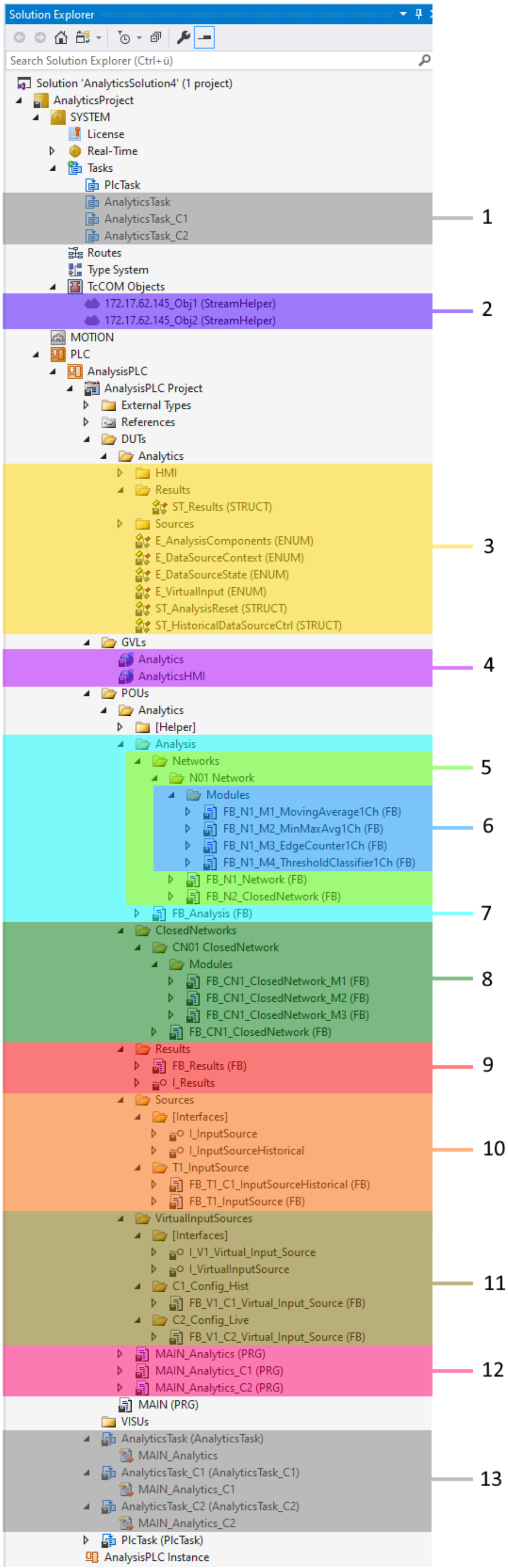
```
PROGRAM MAIN_Analytics_C[n]
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
```

```
    bError: BOOL;
    ipTcResult: I_TcMessage;
    nLastDataHandle: ULINT;
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5 Code version 5.0



Solution Explorer

Search Solution Explorer (Ctrl+ü)

```
Solution 'AnalyticsSolution4' (1 project)
  AnalyticsProject
    SYSTEM
      License
      Real-Time
      Tasks
        PlcTask
        AnalyticsTask
        AnalyticsTask_C1
        AnalyticsTask_C2
      Routes
      Type System
      TcCOM Objects
        172.17.62.145_Obj1 (StreamHelper)
        172.17.62.145_Obj2 (StreamHelper)
    MOTION
    PLC
      AnalysisPLC
        AnalysisPLC Project
          External Types
          References
          DUTs
            Analytics
              HMI
              Results
                ST_Results (STRUCT)
              Sources
              E_AnalysisComponents (ENUM)
              E_DataSourceContext (ENUM)
              E_DataSourceState (ENUM)
              E_VirtualInput (ENUM)
              ST_AnalysisReset (STRUCT)
              ST_HistoricalDataSourceCtrl (STRUCT)
          GVLs
            Analytics
            AnalyticsHMI
          POUs
            Analytics
              [Helper]
              Analysis
                Networks
                  N01 Network
                    Modules
                      FB_N1_M1_MovingAverage1Ch (FB)
                      FB_N1_M2_MinMaxAvg1Ch (FB)
                      FB_N1_M3_EdgeCounter1Ch (FB)
                      FB_N1_M4_ThresholdClassifier1Ch (FB)
                    FB_N1_Network (FB)
                    FB_N2_ClosedNetwork (FB)
                FB_Analysis (FB)
              ClosedNetworks
                CN01 ClosedNetwork
                  Modules
                    FB_CN1_ClosedNetwork_M1 (FB)
                    FB_CN1_ClosedNetwork_M2 (FB)
                    FB_CN1_ClosedNetwork_M3 (FB)
                  FB_CN1_ClosedNetwork (FB)
              Results
                FB_Results (FB)
                I_Results
              Sources
                [Interfaces]
                  I_InputSource
                  I_InputSourceHistorical
                T1_InputSource
                  FB_T1_C1_InputSourceHistorical (FB)
                  FB_T1_InputSource (FB)
              VirtualInputSources
                [Interfaces]
                  I_V1_Virtual_Input_Source
                  I_VirtualInputSource
                C1_Config_Hist
                  FB_V1_C1_Virtual_Input_Source (FB)
                C2_Config_Live
                  FB_V1_C2_Virtual_Input_Source (FB)
              MAIN_Analytics (PRG)
              MAIN_Analytics_C1 (PRG)
              MAIN_Analytics_C2 (PRG)
            MAIN (PRG)
          VISUs
          AnalyticsTask (AnalyticsTask)
            MAIN_Analytics
          AnalyticsTask_C1 (AnalyticsTask_C1)
            MAIN_Analytics_C1
          AnalyticsTask_C2 (AnalyticsTask_C2)
            MAIN_Analytics_C2
          PlcTask (PlcTask)
        AnalysisPLC Instance
```

1: Tasks

A separate task is created for the analytics analysis and for each configuration of a Virtual Input Source.

2: StreamHelper

For each data source of type MQTT binary stream, the code generation creates an instance of a stream helper object to process the incoming binary stream patterns.

3: DataTypes

The data types are created for the analysis. They contain STRUCTs for the reset function or result processing and ENUMs to select the various components.

4: GVLs

To conveniently map module inputs and outputs with the HMI dashboard, selected variables are generated as global variables. In addition, the Data Source function block instances and various parameters are generated as global variables.

5: Network

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

6: Modules

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

7: Analysis

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

8: ClosedNetwork

The ClosedNetwork FBs are generated once with all subnetworks and modules. They can be instantiated multiple times in the analysis. In this way, the generated code can be reduced and simplified.

9: Results

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker in binary or Json format, or saves the data locally to an Analytics binary file.

10: InputSource

The FB InputSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs.

11: VirtualInputSource

The VirtualInputSource interfaces abstract the InputSource symbols from the analysis.

12: MAIN PRGs

In the MAIN_Analytics program, the DataSource FBs are called, the reset function is managed and, if appropriate, the values are mapped with the HMI dashboard. The program is assigned to a separate task.

In the programs MAIN_Analytics_Vx_Cx the FB-Analysis is called. The programs are each assigned to a separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics Version >= 3.1.0.0

6.7.2.5.1 FB_Analysis

The entire analysis routine is defined in the FB Analysis. All configured networks with their modules and error handling are created.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Analysis
VAR_INPUT
    nConfigurationID: INT;
    ipV[n]_VirtualInputs: I_V[n]_Virtual_Input_Source;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
VAR
    [network FBs]
END_VAR
```

Inputs

Name	Type	Description
nConfigurationID	INT	Configuration index
ipV[n]_VirtualInputs	I_V[n]_Virtual_Input_Source	Data for analysis from the specific data source

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.

Properties

Name	Type	Access	Description
nContext	DWORD	Get	Context index

Methods

Name	Definition location	Description
Call [▶ 407]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
Reset [▶ 407]	Local	Resets the entire analysis.
ResultStream [▶ 408]	Local	Optional: If a result stream needs to be created.
SetHMIValues [▶ 408]	Local	Method for filling the HMI structures
GetHMIValues [▶ 408]	Local	Method for setting the parameters from the HMI into the analysis

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.1.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for the analysis.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.5.1.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

Inputs

Name	Type	Description
stReset	ST_AnalysisReset	Structure to define which module or network is to be reset.

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.5.1.3 ResultStream

Syntax

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

Inputs

Name	Type	Description
ipResults	I_Results	Interface pointer to the FB results

Return value

Name	Type	Description
Call	BOOL	

6.7.2.5.1.4 SetHMIValues

Syntax

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to the global HMI structure

Return value

Name	Type	Description
SetHMIValues	BOOL	Is TRUE when completed

6.7.2.5.1.5 GetHMIValues

Syntax

```
METHOD GetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

Inputs

Name	Type	Description
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	Pointer to the global HMI structure

Return value

Name	Type	Description
SetHMIValues	BOOL	Is TRUE when completed

6.7.2.5.2 FB_Network

All modules are sorted in a specific network to achieve a better overview and structure of the configured analysis.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
VAR
    [module FBs]
END_VAR
```

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.

Methods

Name	Definition location	Description
Call [▶ 385]	Local	Method for background communication. The method must be called cyclically.
Reset [▶ 385]	Local	Resetting the network with all submodules.
ValueMapping [▶ 410]	Local	Assignment of the input values to the various module inputs.

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.2.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pAnalysis: POINTER TO FB_Analysis;
END_VAR
```

Inputs

Name	Type	Description
pAnalysis	FB_Analysis	Instance of the analysis FB.

Return value

Name	Type	Description
Call	BOOL	

6.7.2.5.2.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.5.2.3 ValueMapping

Syntax

```
METHOD ValueMapping : BOOL
VAR_INPUT
    pAnalysis : POINTER TO FB_Analysis;
END_VAR
```

Inputs

Name	Type	Description
pAnalysis	FB_Analysis	Instance of the analysis FB

Return value

Name	Type	Description
ValueMapping	BOOL	

6.7.2.5.3 FB_Module

The module FBs contain all inputs and outputs of the configured modules from the workbench configurator. It is also possible to reconfigure the modules during runtime. To do this, simply change the parameter and then start the reconfiguration process with a rising edge at INPUT bReconfigure.

Syntax

Definition:

```
FUNCTION_BLOCK FB_N[n]_M[n]_[Module]
VAR_INPUT
    [module inputs]
END_VAR
VAR_INPUT PERSISTENT
    [module persistent parameter inputs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    [module outputs]
END_VAR
```

Inputs

Name	Type	Description
Module inputs		Inputs of the selected module.

Outputs

Name	Type	Description
bError	BOOL	Becomes TRUE when an error situation occurs.
ipTcResult	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.
Module outputs		Outputs of the selected module.

Properties

Name	Type	Access	Description
nContext	DWORD	Get	Context index

Methods

Name	Definition location	Description
Call [▶ 411]	Local	Method for background communication. The method must be called cyclically.
Reset [▶ 412]	Local	Resetting the module.
SetHMI [▶ 412]	Local	Sets the inputs/outputs to the global HMI structures.
GetHMI [▶ 412]	Local	Optional: Sets the inputs of the global HMI structures to the inputs of the module

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.3.1 Call

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M : I_ValueMapping_M2M;]
END_VAR
```

Inputs

Name	Type	Description
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	Data for the analysis
ipValueMapping_M2M	I_ValueMapping_M2M	Optional: Necessary for mapping values between modules

Return value

Name	Type	Description
Call	BOOL	

6.7.2.5.3.2 Reset

Syntax

```
METHOD Reset : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Reset	BOOL	Is TRUE when completed.

6.7.2.5.3.3 SetHMI

Syntax

```
METHOD SetHMI : BOOL
VAR_INPUT
    nContent : DINT
    pContent : PVOID
    bHMIReinit : BOOL
END_VAR
```

```
VAR
    pHMI_C[n]_[Content] : POINTER TO ST_HMI_C[n]_[Content];
END_VAR
```

Inputs

Name	Type	Description
nContent	DINT	HMI Content Index
pContent	PVOID	Pointer to the HMI content structure
bHMIReinit	BOOL	Initialize the HMI content structure

Return value

Name	Type	Description
SetHMI	BOOL	Is TRUE when completed

6.7.2.5.3.4 GetHMI

Syntax

```
METHOD GetHMI : BOOL
VAR_INPUT
    nContent :  DINT
    pContent :  PVOID
END_VAR
```

```
VAR
    pHMI_C[n]_[Content] : POINTER TO ST_HMI_C[n]_[Content];
END_VAR
```

Inputs

Name	Type	Description
nContent	DINT	HMI Content Index
pContent	PVOID	Pointer to the HMI content structure

Return value

Name	Type	Description
GetHMI	BOOL	Is TRUE when completed

6.7.2.5.4 FB_InputSource

The FB DataSource manages the receipt of input values from the various sources. In the OUTPUT declaration you will find all configured inputs.

Syntax

Definition:

```
FUNCTION_BLOCK FB_T[n]_InputSource IMPLEMENTS I_InputSource
VAR
END_VAR
```

Interfaces

Type	Description
I_InputSource	Interface for communication with a data source

Methods

Name	Definition location	Description
Call [▶ 413]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
GetData [▶ 414]	Local	Method to retrieve the data of the specified element.
NewDataAvailable [▶ 414]	Local	Method to check if new data is available.
AddClient [▶ 415]	Local	Method for adding data Clients
ClientDone [▶ 415]	Local	Method to signal that the client has received all data.

Properties

Name	Type	Access	Description
bReadNewData	BOOL	Get	.
eDataState	E_DataSourceState	Get	
nDataHandle	ULINT	Get	
nElements	UDINT	Get	
nMaxOversamplingFactor	UDINT	Get	

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.4.1 Call

Syntax

```
METHOD Call : BOOL
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.5.4.2 GetData

Syntax

```
METHOD GetData : BOOL
VAR_INPUT
    nElement    : UDINT;
    pInputs     : PVOID;
    nInputsSize : UDINT
END_VAR
VAR_OUTPUT
    nTimestamp  : ULINT;
    nContext    : DWORD;
END_VAR
```

Inputs

Name	Type	Description
nElement	UDINT	Element ID to obtain the specific sample
pInputs	PVOID	Pointer to the data structure
nInputsSize	UDINT	Size of the data structure

Outputs

Name	Type	Description
nTimestamp	ULINT	Timestamp data
nContext	DWORD	Data context

Return value

Name	Type	Description
GetData	BOOL	Is TRUE if a new element is selected

6.7.2.5.4.3 NewDataAvailable

Syntax

```
METHOD NewDataAvailable : BOOL
VAR_INPUT
    nLastDataHandle : ULINT;
END_VAR
```

Inputs

Name	Type	Description
nLastDataHandle	ULINT	Handle of the last fetched data packet

Return value

Name	Type	Description
NewDataAvailable	BOOL	Is TRUE if new data is available

6.7.2.5.4.4 AddClient

Syntax

```
METHOD AddClient : BOOL
VAR_OUTPUT
    nClientID     : DWORD;
END_VAR
```

Outputs

Name	Type	Description
nClientID	LWORD	Client-ID

6.7.2.5.4.5 ClientDone

Syntax

```
METHOD ClientDone : BOOL
VAR_OUTPUT
    nClientID     : DWORD;
END_VAR
```

Outputs

Name	Type	Description
nClientID	LWORD	Client-ID

6.7.2.5.5 FB_VirtualInputSource

The FB VirtualInputSource abstracts the InputSources for the different analysis configurations. The virtual inputs configured in the Analytics Workbench are set here.

Syntax

Definition:

```
FUNCTION_BLOCK FB_V[n]_C[n]_VirtualInputSource IMPLEMENTS I_VirtualInputSource,
I_V[n]_VirtualInputSource
VAR
END_VAR
```

Interfaces

Type	Description
I_VirtualInputSource	Interface for communication with a data source
I_V[n]_VirtualInputSource	Interface that provides all defined virtual inputs

Methods

Name	Definition location	Description
SourceSync [▸ 416]	Local	Method for synchronizing multiple FB instances
NextData [▸ 416]	Local	Method to accept the next data set
Done [▸ 416]	Local	Method of signaling that the entire data packet has been processed.

Properties

Name	Type	Access	Description
bEndOfData	BOOL	Get	Signals the end of the data packet
dtTimestamp	DCTIMESTRUCT	Get	Timestamp of the currently accepted data set
nDataHandle	ULINT	Get	Data handle
nContext	DWORD	Get	Data context
tTimestamp	ULINT	Get	Timestamp of the currently received data set

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.5.1 SourceSync

Syntax

```
METHOD SourceSync : BOOL
```

6.7.2.5.5.2 NextData

Syntax

```
METHOD NextData : BOOL
VAR_INPUT
    nMaxOversampling    : UDINT;
END_VAR
```

Inputs

Name	Type	Description
nMaxOversampling	UDINT	Specifies the maximum oversampling value

Return value

Name	Type	Description
NextData	BOOL	Is TRUE if a new element is selected

6.7.2.5.5.3 Done

Syntax

```
METHOD Done : BOOL
VAR_INPUT
END_VAR
```

6.7.2.5.6 FB_Results

If analysis results need to be saved or streamed, the FB Results manages this and streams the selected variables to the message broker or saves the data to the Analytics binary file.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    nTimestamp: ULINT;
    stResults: ST_Results;
END_VAR
```

Outputs

Name	Type	Description
nTimestamp	ULINT	Associated timestamp of the result structure data
stResults	ST_Results	Result structure that contains all elements of the result stream.

Properties

Name	Type	Access	Description
bInitialized	BOOL	Get	Indicates whether the function block has been properly initialized
nMaxSamples	INT	Get	Maximum number of buffered results
nResultCount	INT	Get	Current number of buffered results

Methods

Name	Definition location	Description
Call [▶ 417]	Local	Method for background communication. The method must be called cyclically.
AddResult [▶ 417]	Local	Add a sample to the result stream
SendResults [▶ 418]	Local	Sends all buffered samples of the result stream
Release [▶ 418]	Local	Close stream or file of the result stream

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.6.1 Call

Syntax

```
METHOD Call : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Call	BOOL	

6.7.2.5.6.2 AddResult

Syntax

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

Inputs

Name	Type	Description
tTimestamp	ULINT	Timestamp of the sample
stSample	ST_Results	Sample structure

Return value

Name	Type	Description
AddResult	BOOL	

6.7.2.5.6.3 SendResults

Syntax

```
METHOD SendResults : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
SendResults	BOOL	

6.7.2.5.6.4 Release

Syntax

```
METHOD Release : BOOL
VAR
END_VAR
```

Return value

Name	Type	Description
Release	BOOL	

6.7.2.5.7 MAIN_Analytics

In the MAIN_Analytics program, the InputSource FBs are called, the reset function is managed and, if necessary, the values are mapped with the HMI dashboard. The program is assigned to a separate task.

Syntax

Definition:

```
PROGRAM MAIN_Analytics
VAR
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.7.2.5.8 MAIN_Analytics_C[n]

The FB Analysis is called in the program MAIN_Analytics_C[n]. The program is assigned to the separate task.

It is also possible to reset single modules, whole networks or all defined networks with only one rising flag. First, select the component to be reset. Then a rising edge at INPUT bReset starts the reset process.

All reset calls are defined in the action A_Reset.

The handling of the "InputSources" is done in the action A_InputSources.

The mapping of the HMI values is done in the action A_MapToHMI.

Syntax

Definition:

```
PROGRAM MAIN_Analytics_C[n]
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    nAnalysisResultsTimestamp: ULINT;
    stAnalysisResults: ST_Results;
END_VAR
```

Requirements

Development environment	Target platform	Plc libraries to include
TwinCAT v3.1.4024.0	PC or CX (x64, x86)	Tc3_Analytics

6.8 HMI One-Click Dashboard

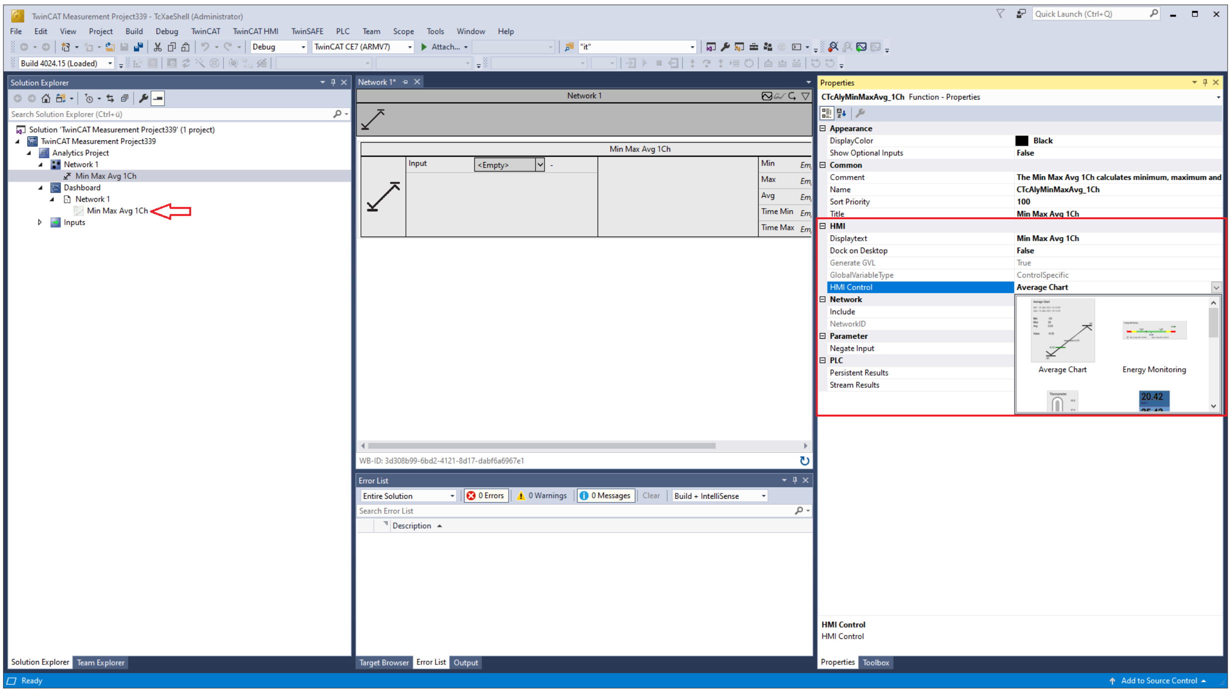
It is possible to automatically generate an HMI dashboard with HMI Controls for all modules and parameters configured in the TwinCAT Analytics Workbench Configurator. The HMI Dashboard is based on the TwinCAT HMI and visualizes the PLC data from the runtime deployment [▶ 353].

The automatically generated One-Click Dashboard is only available with the new HMI version 1.12. An Analytics Runtime license is required in order to use the Analytics HMI Controls.

✓ After configuring your Analytics Workbench project, an HMI Control can be selected for each algorithm.

1. Open the **Properties** window of the module and select an **HMI Control** from the dropdown list. You can change the display text for the title in the HMI dashboard (display text). You can also choose whether the control should be docked to the start page (Dock on Desktop). In the **Solution Explorer** all the controls

to be generated are stored under the Manage dashboard structure and content in Analytics project [▶ 422]


TwinCAT Measurement Project339 - TcXaeShell (Administrator)
Solution Explorer
Solution 'TwinCAT Measurement Project339' (1 project)
TwinCAT Measurement Project339
Analytics Project
Network 1
Min Max Avg 1Ch
Dashboard
Network 1
Min Max Avg 1Ch
Inputs
Network 1*
Min Max Avg 1Ch
Error List
Properties

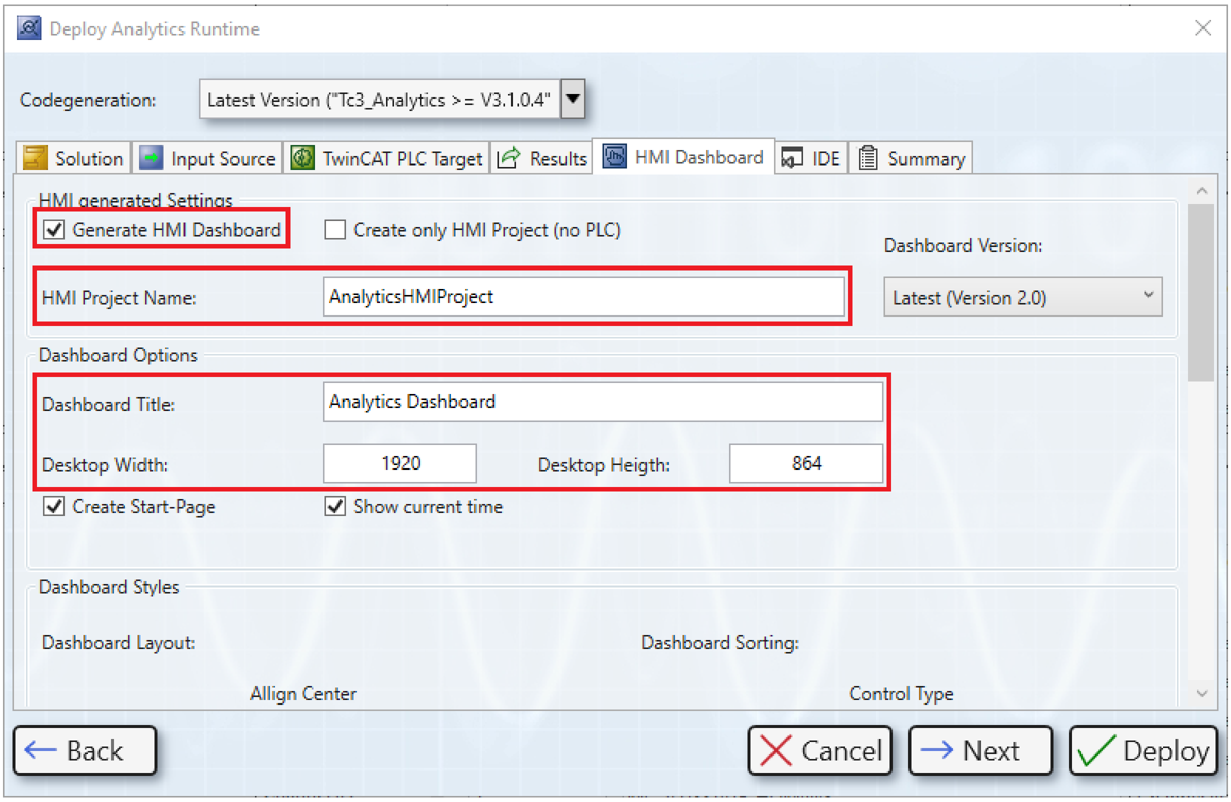
CTcAlyMinMaxAvg_1Ch Function - Properties

Appearance	
DisplayColor	■ Black
Show Optional Inputs	False
Common	
Comment	The Min Max Avg 1Ch calculates minimum, maximum and
Name	CTcAlyMinMaxAvg_1Ch
Sort Priority	100
Title	Min Max Avg 1Ch
HMI	
Displaytext	Min Max Avg 1Ch
Dock on Desktop	False
Generate GVL	True
GlobalVariableType	ControlSpecific
HMI Control	Average Chart
Network	
Include	
NetworkID	
Parameter	
Negate Input	
PLC	
Persistent Results	
Stream Results	

2. After completion of the configuration, click the **Deploy Analytics Runtime** command in the context menu. The Deploy Wizard starts and it is possible to set up the entire required configuration step by step for use.

   ⇨ You can configure your HMI Dashboard on the **HMI Dashboard** tab.

3. Activate the **Generate HMI Dashboard** checkbox. It is also possible to create only one HMI project without a PLC. Furthermore, you can also assign an **HMI Project Name** to the dashboard and set a Dashboard Title as well as the Desktop Height and Desktop Width in order to generate a tailor-made dashboard that suits your needs. The remaining configurations are explained in Dashboard Configuration [▶ 454].



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4")

HMI generated Settings

☑ Generate HMI Dashboard ☐ Create only HMI Project (no PLC)

Dashboard Version: Latest (Version 2.0)

HMI Project Name: AnalyticsHMIProject

Dashboard Options

Dashboard Title: Analytics Dashboard

Desktop Width: 1920 Desktop Heigth: 864

☑ Create Start-Page ☑ Show current time

Dashboard Styles

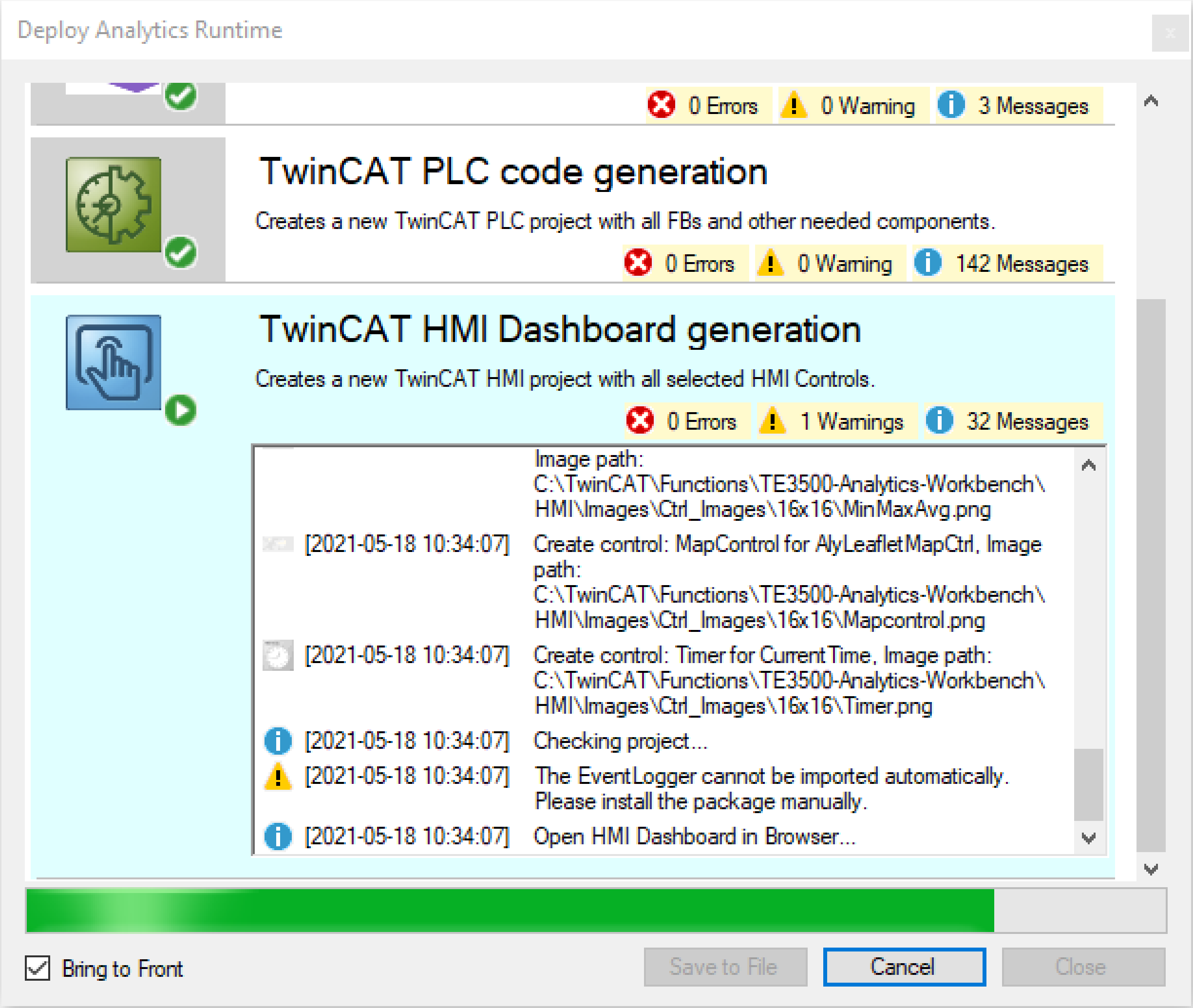
Dashboard Layout: Allign Center

Dashboard Sorting: Control Type

⇨ As usual, the last tab shows you all the settings you have made for the generation.

4. Now you can start the generation process by clicking **Deploy**.

⇨ The HMI generation begins immediately after runtime deployment (if selected). Each step for generating the HMI dashboard is also displayed in the overview window during the generation process.



TwinCAT PLC code generation

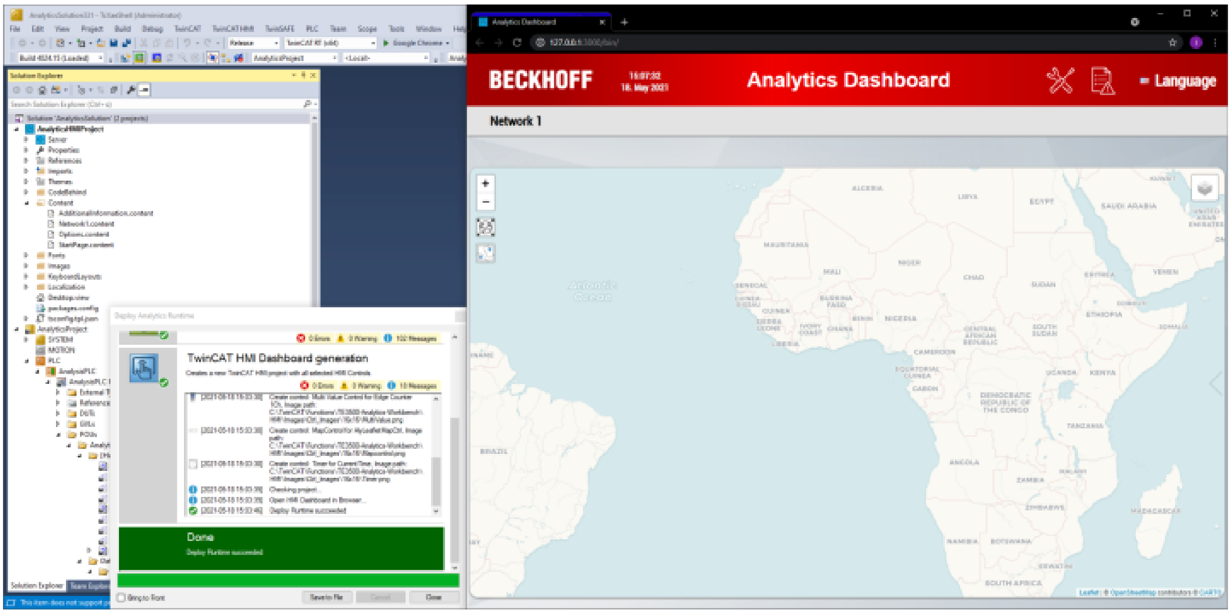
Creates a new TwinCAT PLC project with all FBs and other needed components.

TwinCAT HMI Dashboard generation

Creates a new TwinCAT HMI project with all selected HMI Controls.

```
                       Image path:
                       C:\TwinCAT\Functions\TE3500-Analytics-Workbench\
                       HMI\Images\Ctrl_Images\16x16\MinMaxAvg.png
[2021-05-18 10:34:07]  Create control: MapControl for AlyLeafletMapCtrl, Image
                       path:
                       C:\TwinCAT\Functions\TE3500-Analytics-Workbench\
                       HMI\Images\Ctrl_Images\16x16\Mapcontrol.png
[2021-05-18 10:34:07]  Create control: Timer for CurrentTime, Image path:
                       C:\TwinCAT\Functions\TE3500-Analytics-Workbench\
                       HMI\Images\Ctrl_Images\16x16\Timer.png
[2021-05-18 10:34:07]  Checking project...
[2021-05-18 10:34:07]  The EventLogger cannot be imported automatically.
                       Please install the package manually.
[2021-05-18 10:34:07]  Open HMI Dashboard in Browser...
```

⇨ The dashboard opens automatically in your default browser.



Deploy Analytics Runtime

0 Errors 0 Warning 102 Messages

TwinCAT HMI Dashboard generation

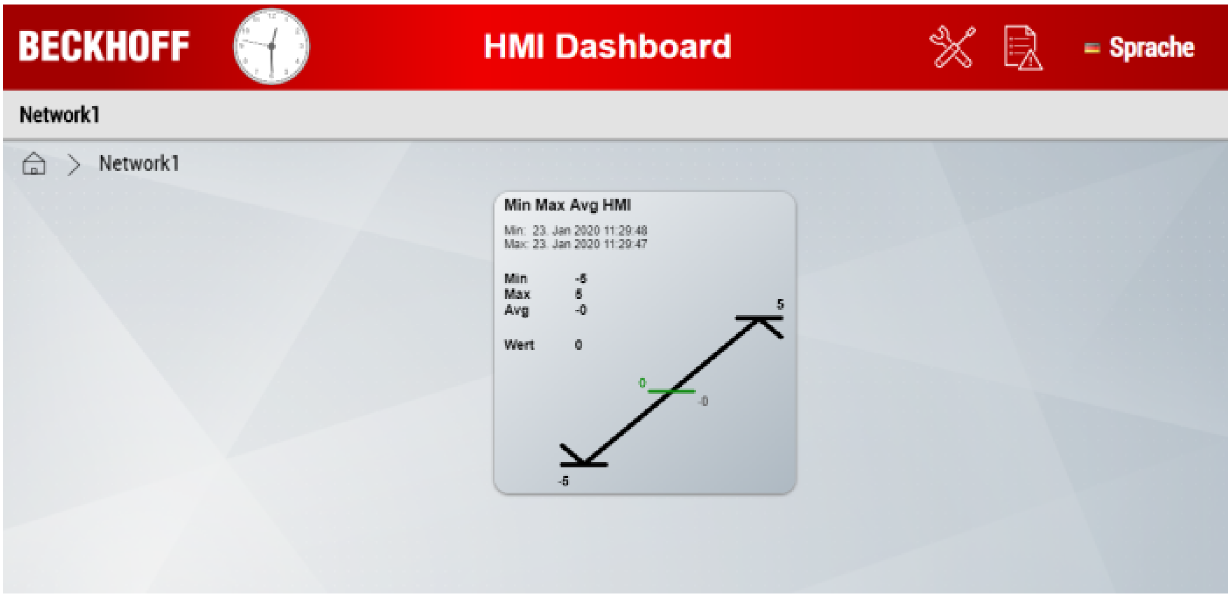
Creates a new TwinCAT HMI project with all selected HMI Controls.

0 Errors 0 Warning 10 Messages

```
[2021-05-18 15:03:38]  Create control: Multi Value Control for Edge Counter 1Ch. Image path: C:\TwinCAT\Functions\TE3500-Analytics-Workbench\HMI Images\Ctrl_Images\18x18\MultiValue.png
[2021-05-18 15:03:38]  Create control: MapControl for MyLeafletMapCtrl. Image path: C:\TwinCAT\Functions\TE3500-Analytics-Workbench\HMI Images\Ctrl_Images\18x18\Mapcontrol.png
[2021-05-18 15:03:38]  Create control: Timer for CurrentTime. Image path: C:\TwinCAT\Functions\TE3500-Analytics-Workbench\HMI Images\Ctrl_Images\18x18\Timer.png
[2021-05-18 15:03:39]  Checking project...
[2021-05-18 15:03:39]  Open HMI Dashboard in Browser...
[2021-05-18 15:03:40]  Deploy Runtime succeeded
```

Done

Deploy Runtime succeeded

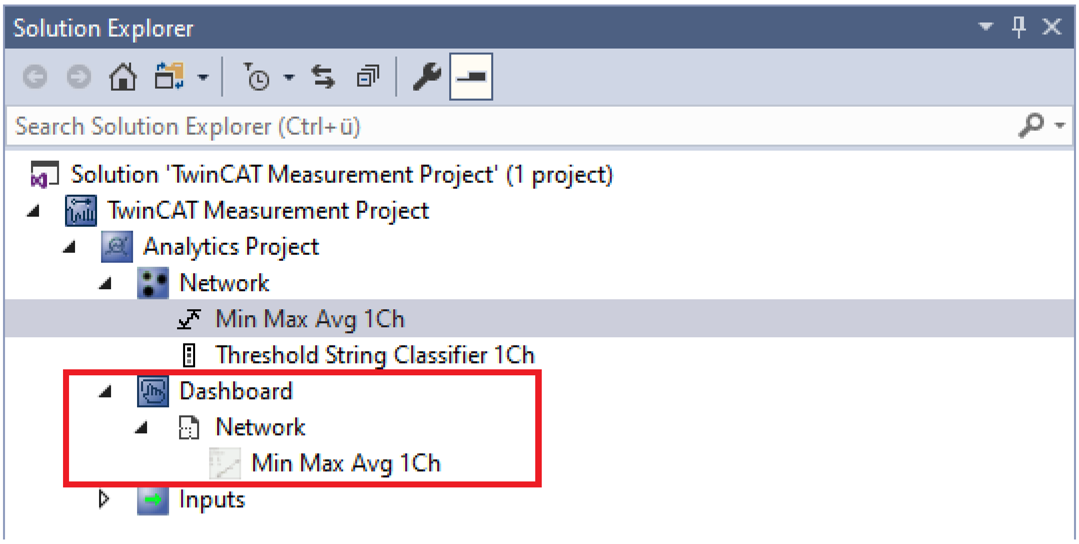


Min: 23. Jan 2020 11:29:48
Max: 23. Jan 2020 11:29:47

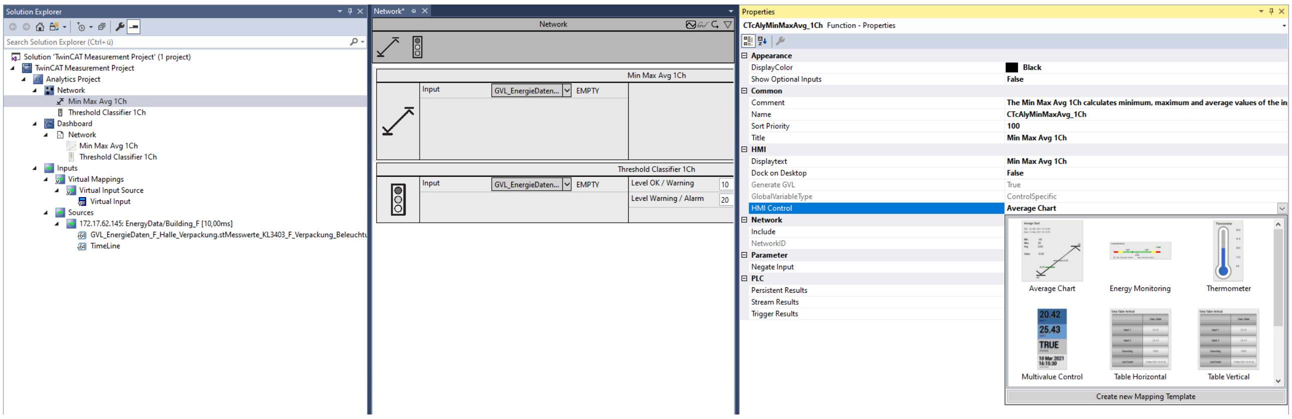
Min	-5
Max	5
Avg	-0
Wert	0

6.8.1 Manage dashboard structure and content in Analytics project

An Analytics project has a configuration of HMI contents (pages) and HMI controls (display elements) that are created during dashboard generation. This configuration can be viewed and changed via the **Dashboard node** in the Solution Explorer. You can rename the contents and controls at any time, move them to other contents via drag & drop, copy them (**Ctrl-C, Ctrl-V**) or delete them. A control can also be edited by double-clicking on it.


Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'TwinCAT Measurement Project' (1 project)
TwinCAT Measurement Project
Analytics Project
Network
Min Max Avg 1Ch
Threshold String Classifier 1Ch
Dashboard
Network
Min Max Avg 1Ch
Inputs

Each Analytics module has existing control mappings, which can be selected via the window **Properties** (you can also create/edit these yourself via "Create new Mapping Template" or via the Use customized and own controls [▶ 432] (point 2)). Once a control is selected for a module, it is listed under the **Dashboard** node.



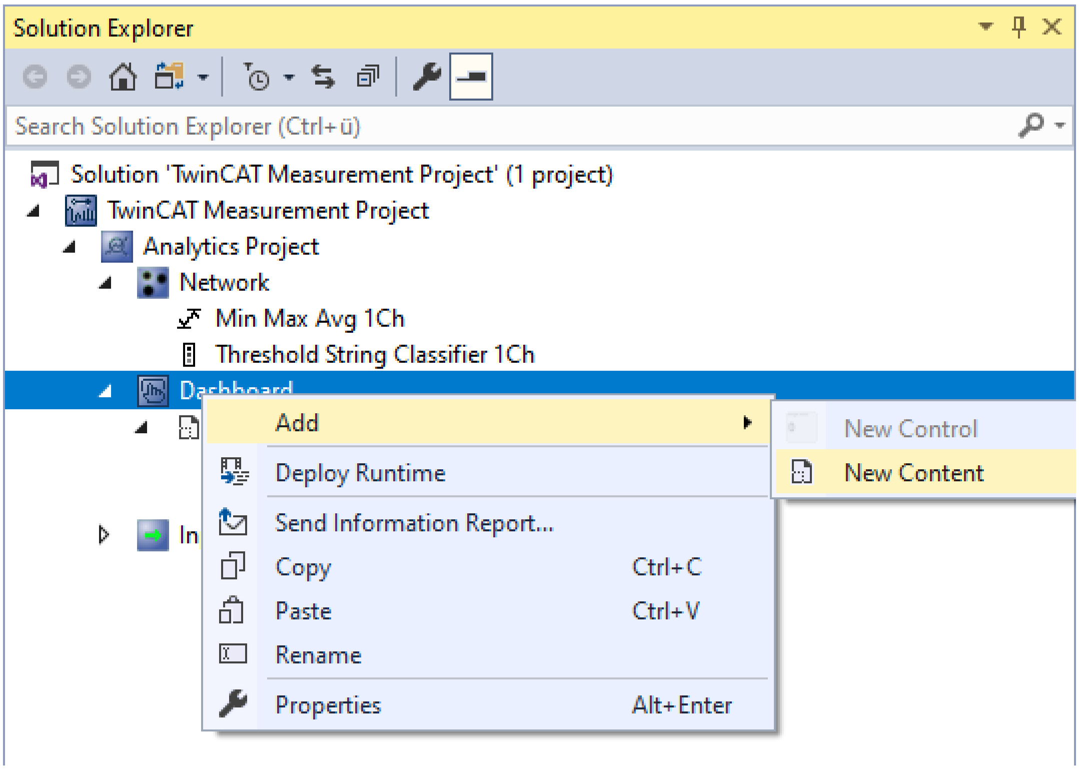
Min Max Avg 1Ch: Input GVL_EnergieDaten... EMPTY

Threshold Classifier 1Ch: Input GVL_EnergieDaten... EMPTY; Level OK / Warning 10; Level Warning / Alarm 20

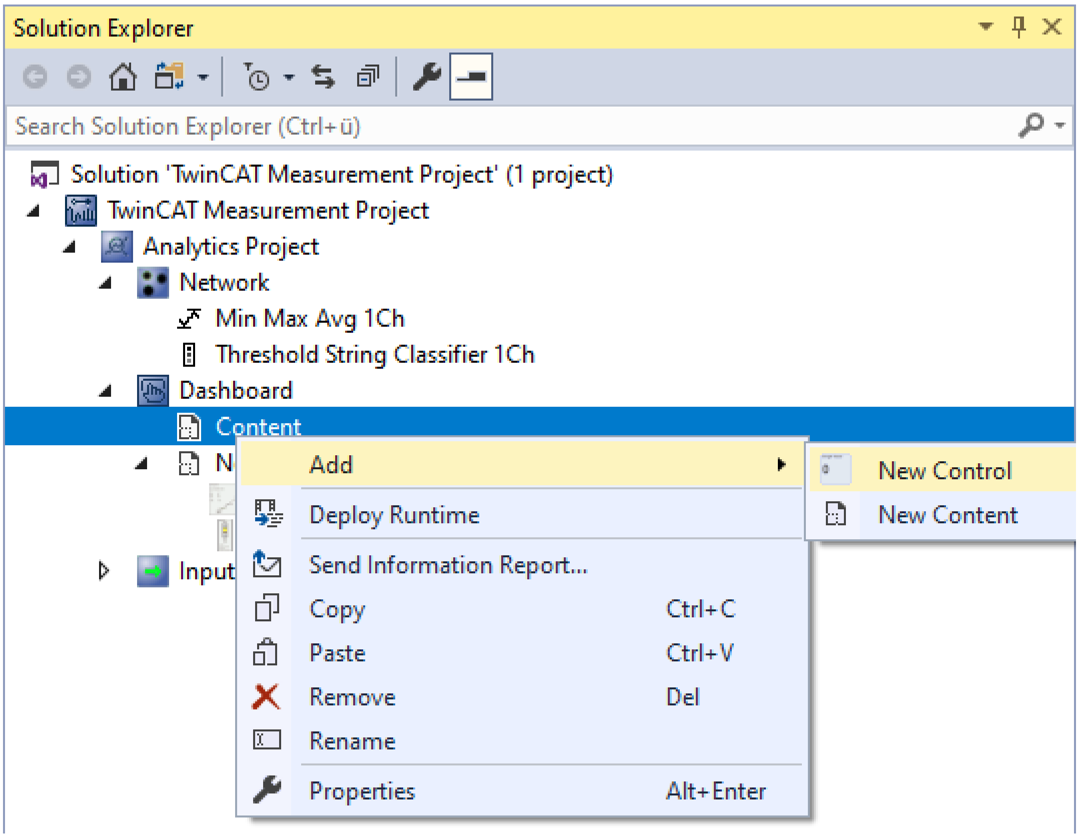
Properties
CTcAlyMinMaxAvg_1Ch Function - Properties

Appearance	
DisplayColor	Black
Show Optional Inputs	False
Common	
Comment	The Min Max Avg 1Ch calculates minimum, maximum and average values of the in
Name	CTcAlyMinMaxAvg_1Ch
Sort Priority	100
Title	Min Max Avg 1Ch
HMI	
Displaytext	Min Max Avg 1Ch
Dock on Desktop	False
Generate GVL	True
GlobalVariableType	ControlSpecific
HMI Control	Average Chart
Network	
Include	
NetworkID	
Parameter	
Negate Input	
PLC	
Persistent Results	
Stream Results	
Trigger Results	

You can create new content for the **Dashboard** and **Content** nodes by right-clicking.


Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'TwinCAT Measurement Project' (1 project)
TwinCAT Measurement Project
Analytics Project
Network
Min Max Avg 1Ch
Threshold String Classifier 1Ch
Dashboard
In
Add
Deploy Runtime
Send Information Report...
Copy Ctrl+C
Paste Ctrl+V
Rename
Properties Alt+Enter
New Control
New Content

Likewise, you can add **new controls** by right-clicking on a **Content** node.



This opens the **Analytics Dashboard Wizard**. This wizard guides you step-by-step through the configuration for adding a control.

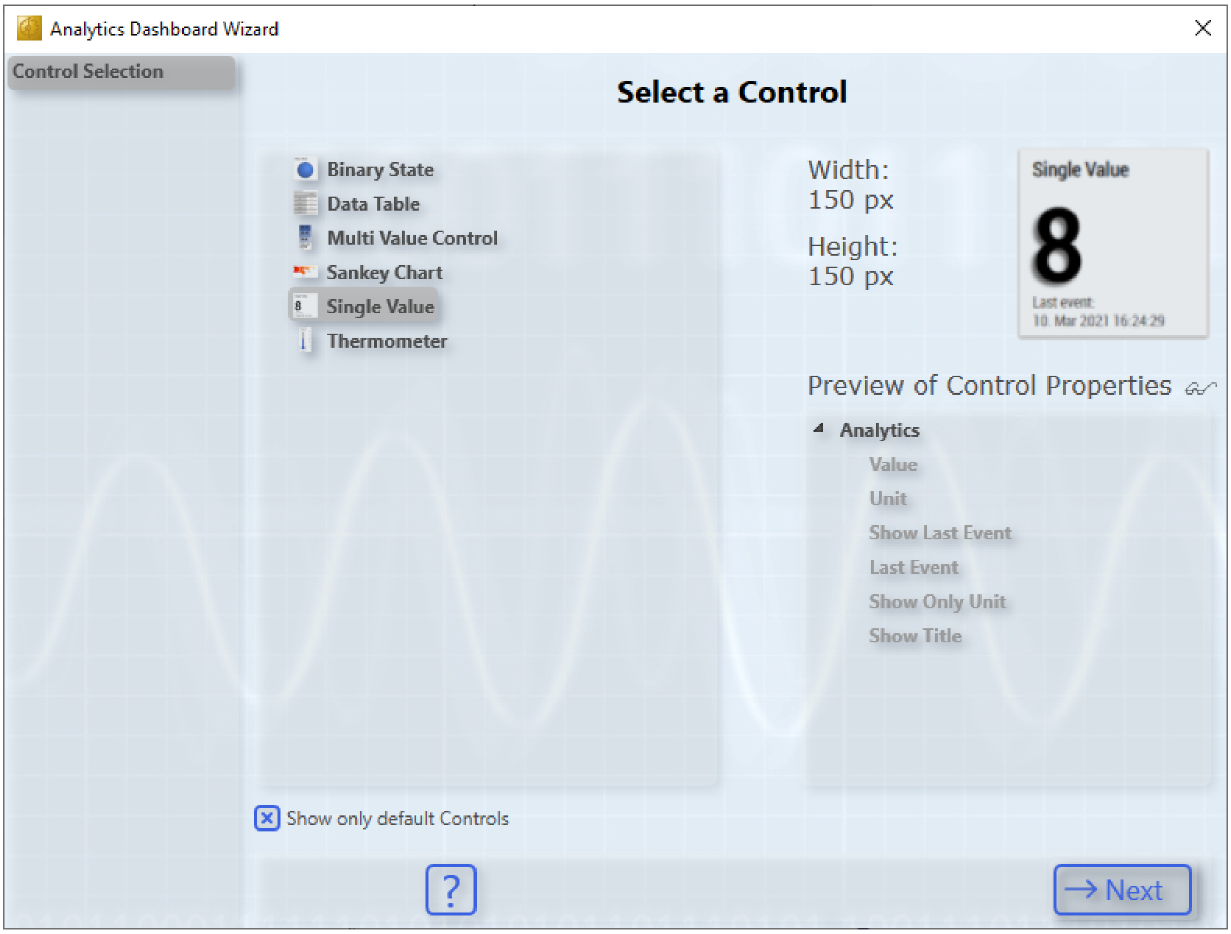
Analytics Dashboard Wizard

Familiarize yourself with the wizard by adding a control (select a control from the Properties window of the module). You can then look up the entire configuration in the wizard by double-clicking on the control. There you can follow the individual configuration steps.

Analytics Dashboard Wizard

Select a control. All available controls are listed on the left. By default, only controls that are not algorithm-specific are listed. All controls can be made available by unchecking the **Show only default Controls** checkbox.

1. Select **Single Value** to display a single value in the dashboard. Click the **Next** button to continue.



Analytics Dashboard Wizard

Control Selection

Select a Control

Binary State
Data Table
Multi Value Control
Sankey Chart
Single Value
Thermometer

Width:
150 px

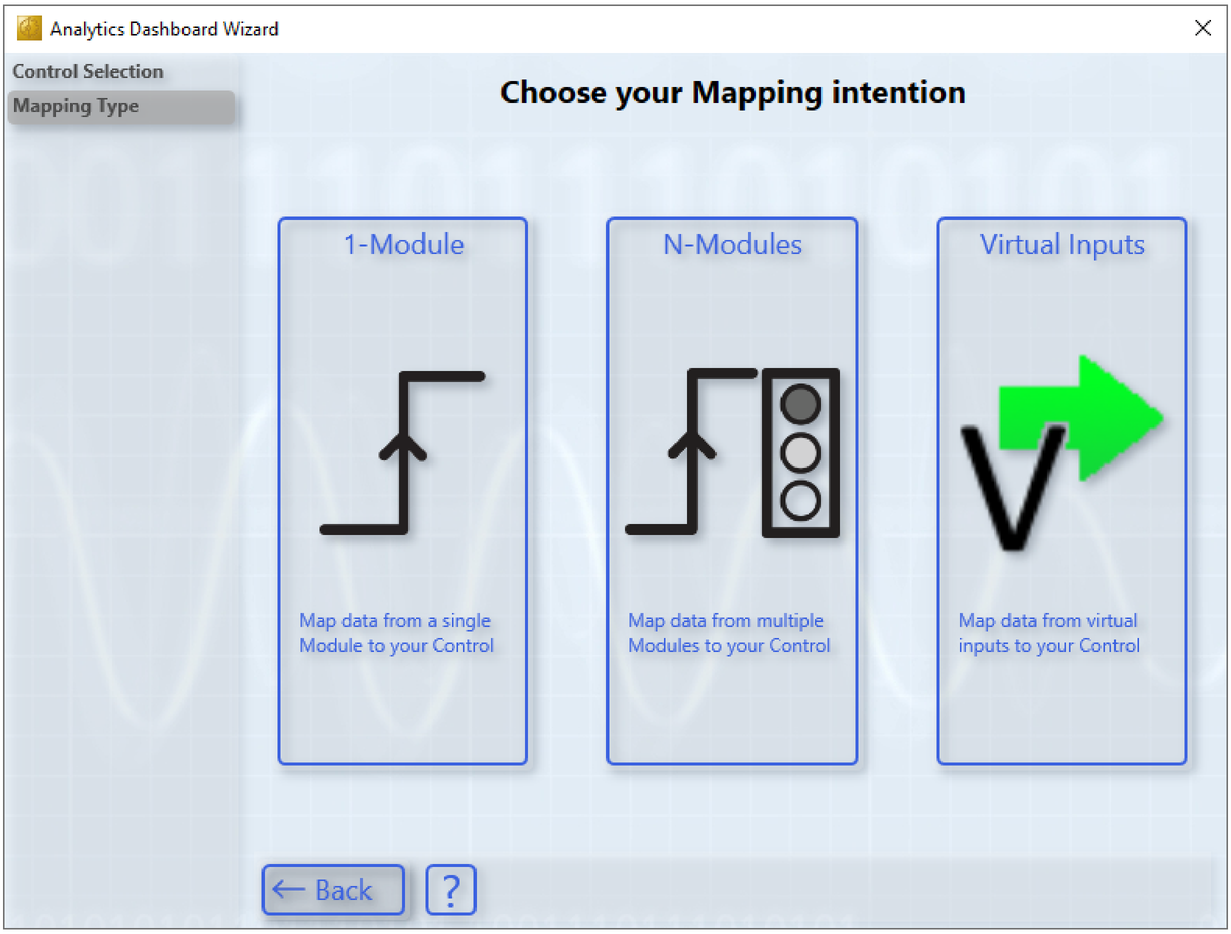
Height:
150 px

Preview of Control Properties

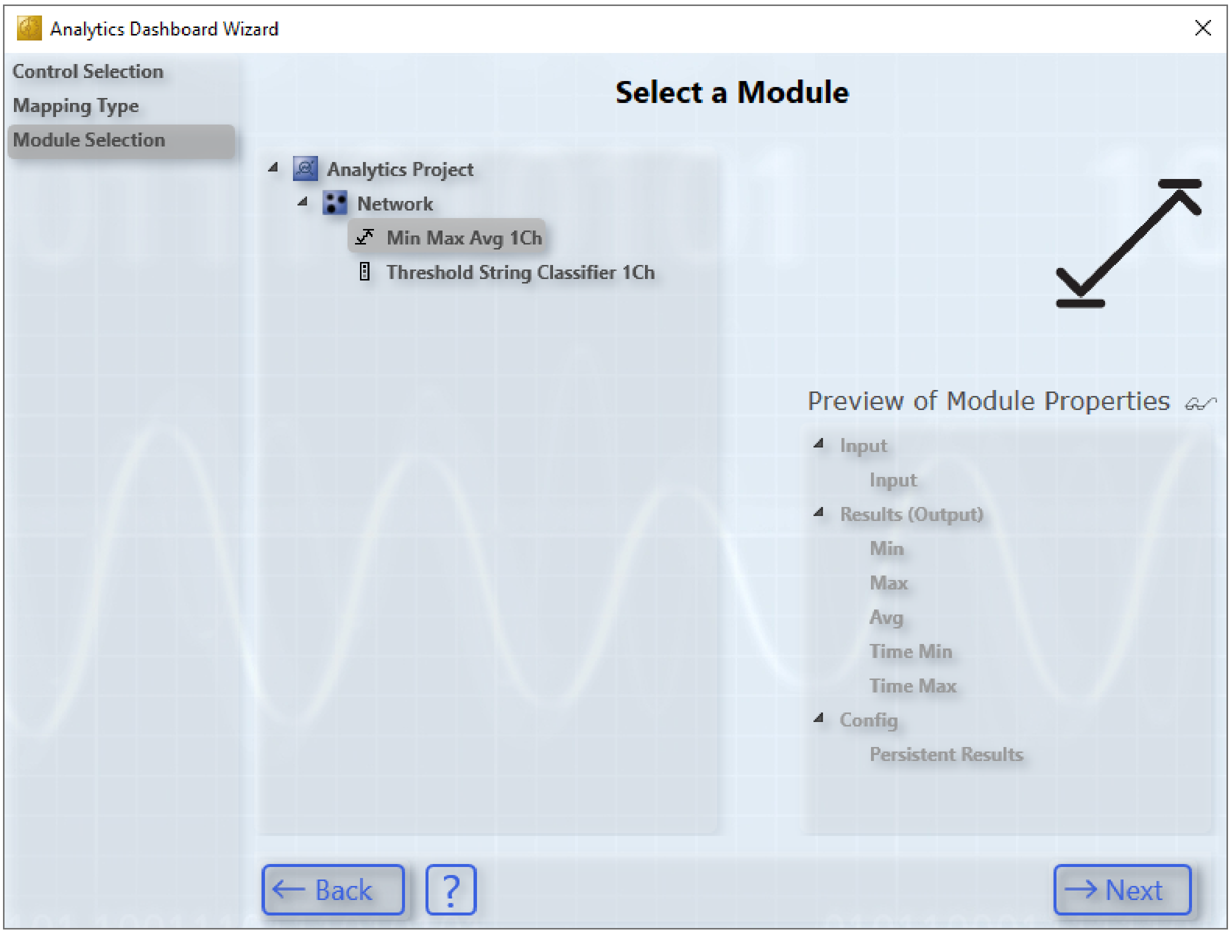
Analytics
Value
Unit
Show Last Event
Last Event
Show Only Unit
Show Title

☒ Show only default Controls

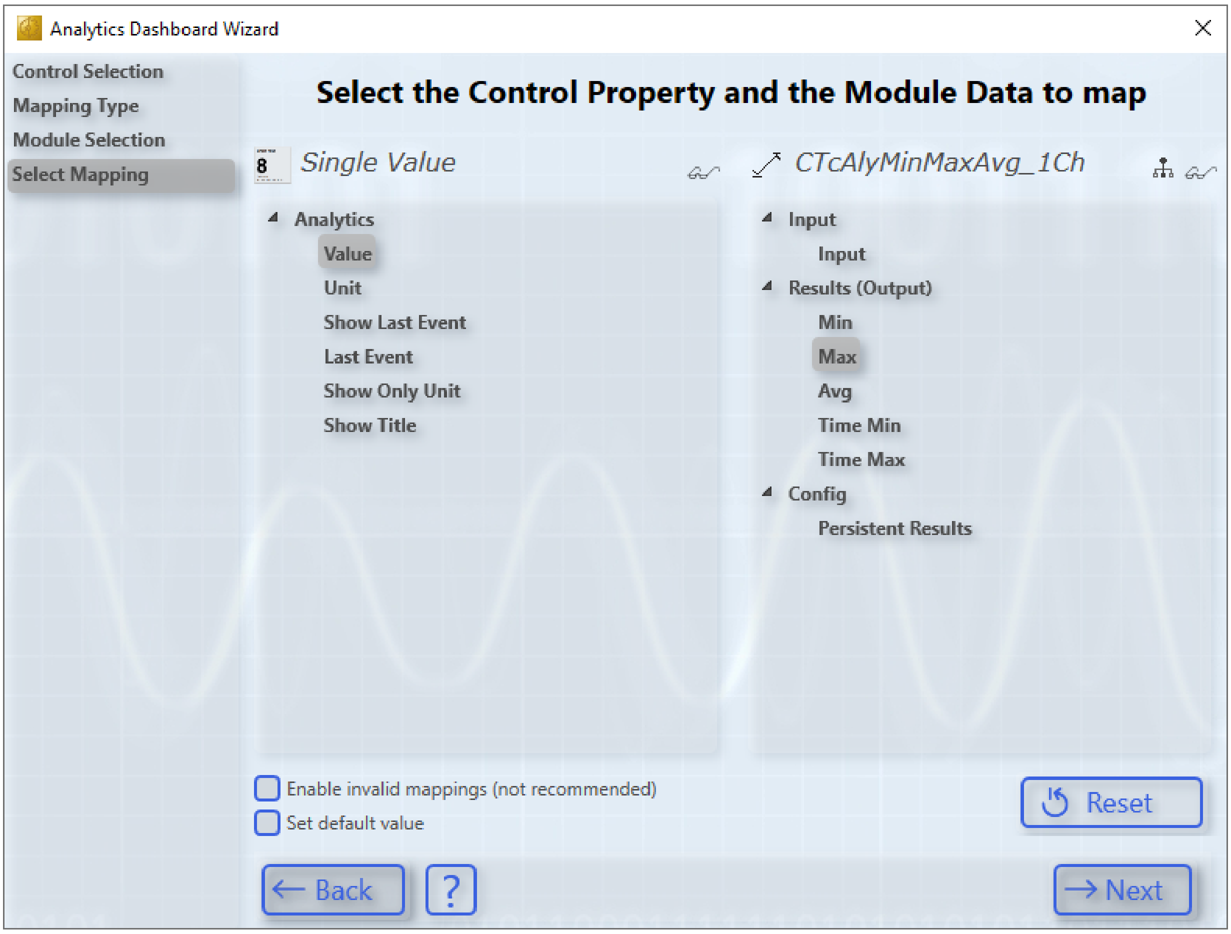
2. You can now link data from one module (**1-Module**), several modules (**N-Modules**) or from virtual inputs (**Virtual Inputs**). For this example, select **1-Module**.


Analytics Dashboard Wizard
Control Selection
Mapping Type
Choose your Mapping intention
1-Module
Map data from a single Module to your Control
N-Modules
Map data from multiple Modules to your Control
Virtual Inputs
Map data from virtual inputs to your Control
← Back
?

3. All modules from the Analytics project are listed. Select the module from which you want to display the data. A preview of the selected module is available on the right.



4. Here the first control property is linked to a module variable. To do this, select the **Value** to be displayed with the variable **Max**.



Analytics Dashboard Wizard

Select the Control Property and the Module Data to map

Single Value

- Analytics
  - Value
  - Unit
  - Show Last Event
  - Last Event
  - Show Only Unit
  - Show Title

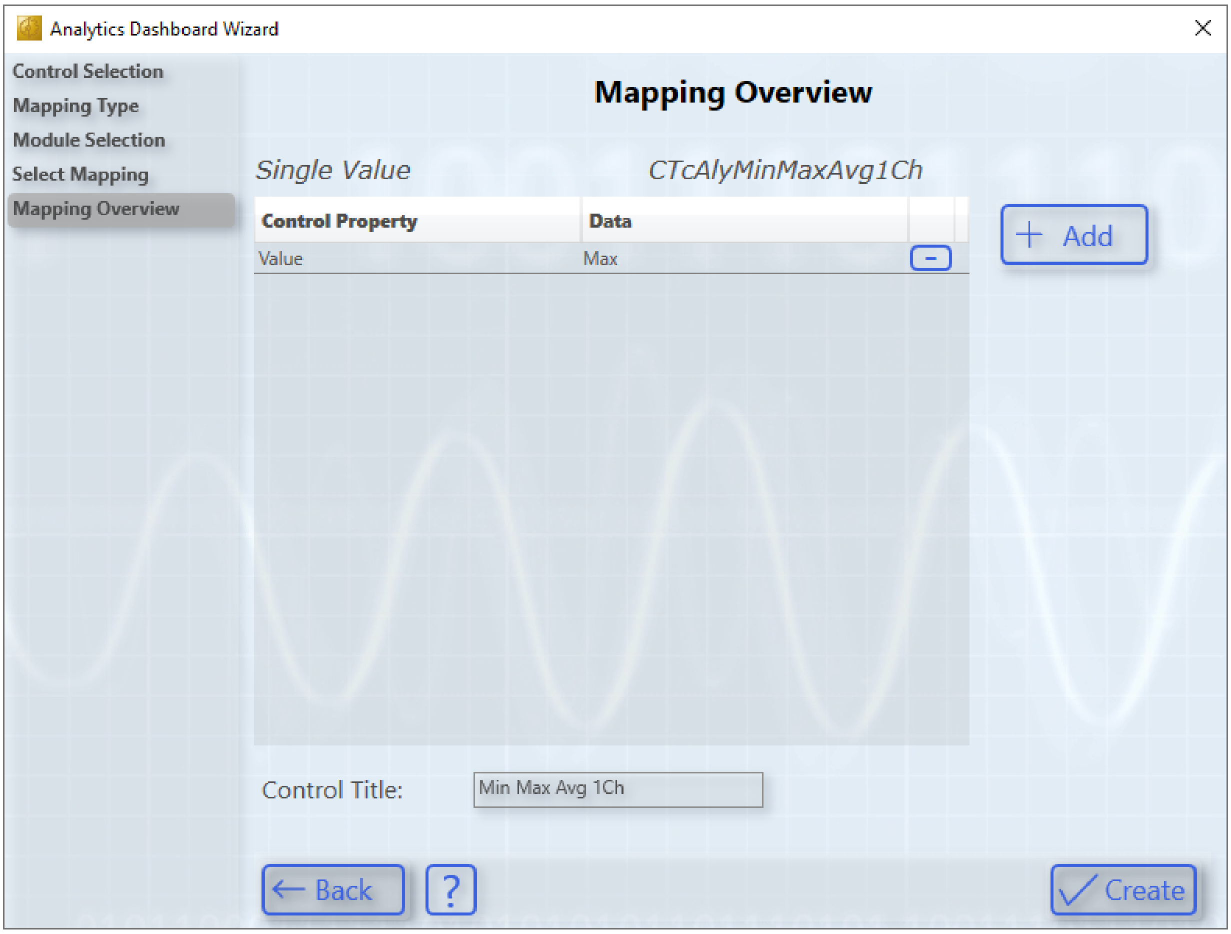
CTcAlyMinMaxAvg_1Ch

- Input
  - Input
- Results (Output)
  - Min
  - Max
  - Avg
  - Time Min
  - Time Max
- Config
  - Persistent Results

□ Enable invalid mappings (not recommended)

□ Set default value

5. This overview page lists all existing links. Add another link with **Add**.

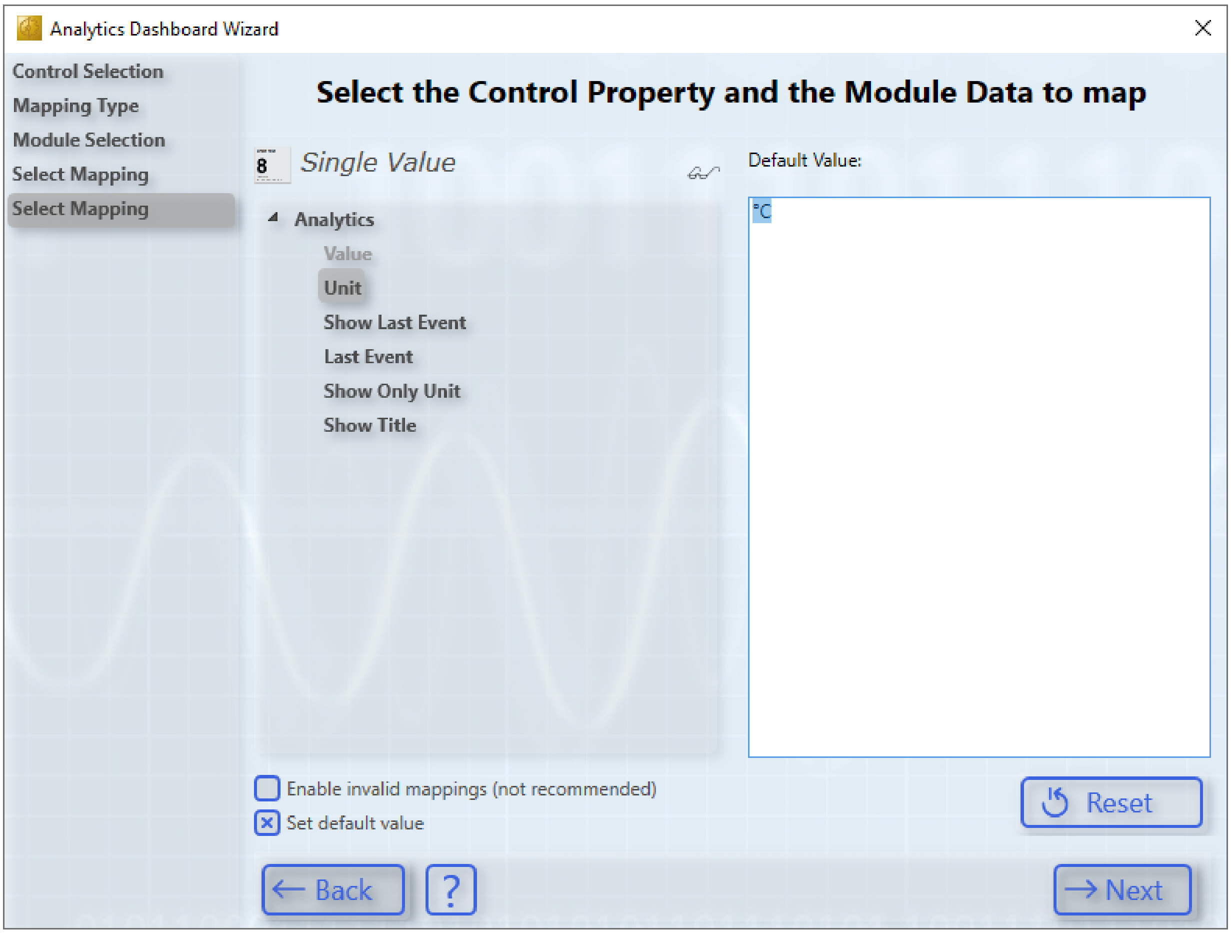


Single Value CTcAlyMinMaxAvg1Ch

Control Property	Data
Value	Max

Control Title: Min Max Avg 1Ch

6. Select **Unit** and check **Set default value** to assign a static value. A text field opens on the right; enter **°C**.

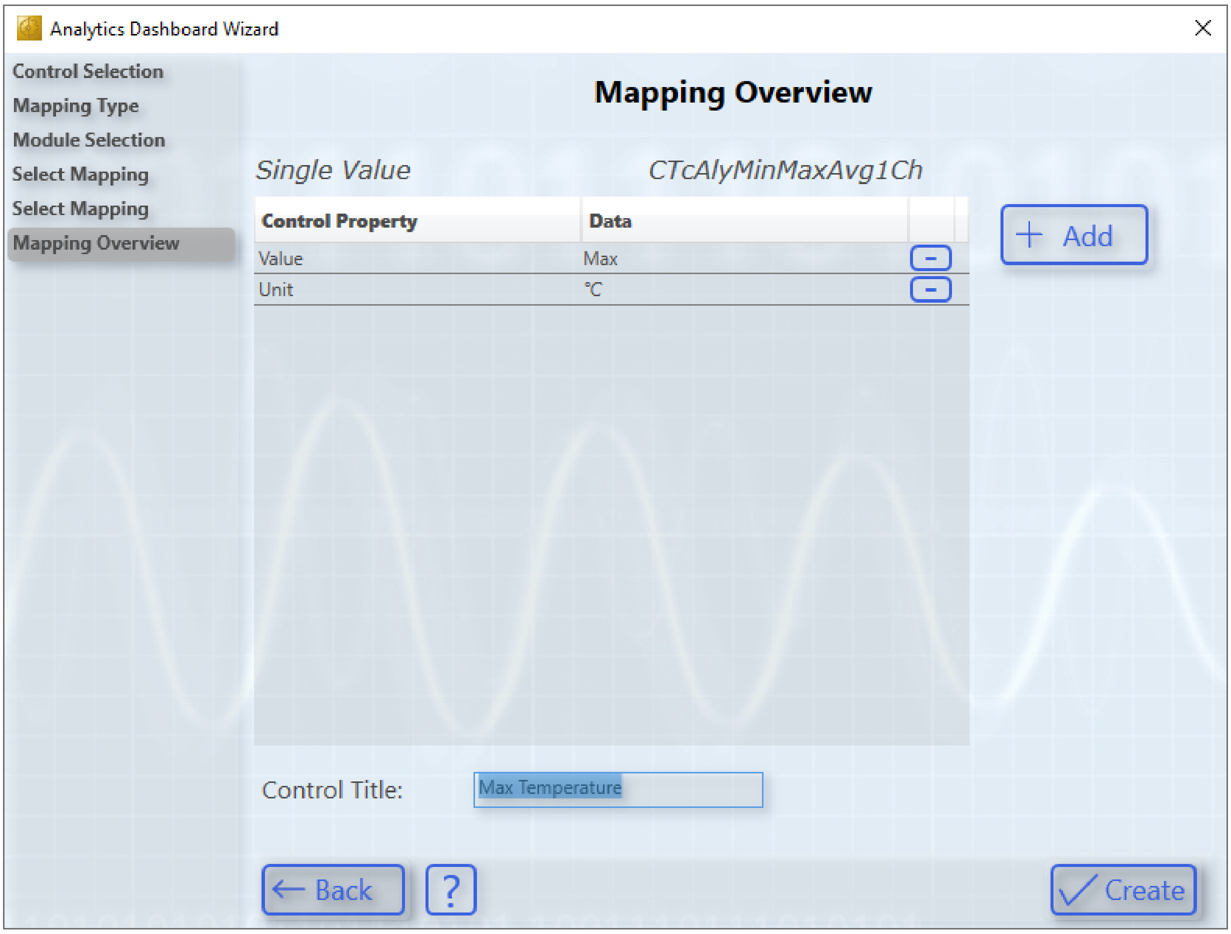


Default Value: °C

□ Enable invalid mappings (not recommended)

☒ Set default value

7. Change the title of the control to **Max Temperature** and add the control mapping via **Create**.



Analytics Dashboard Wizard

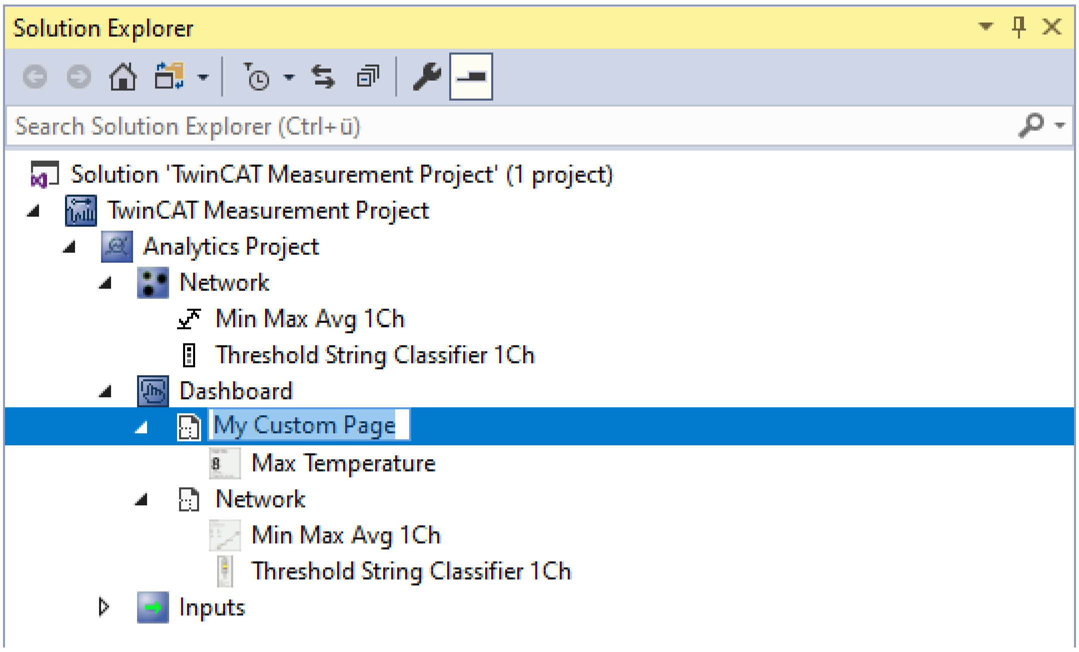
Mapping Overview

Single Value *CTcAlyMinMaxAvg1Ch*

Control Property	Data
Value	Max
Unit	°C

Control Title: Max Temperature

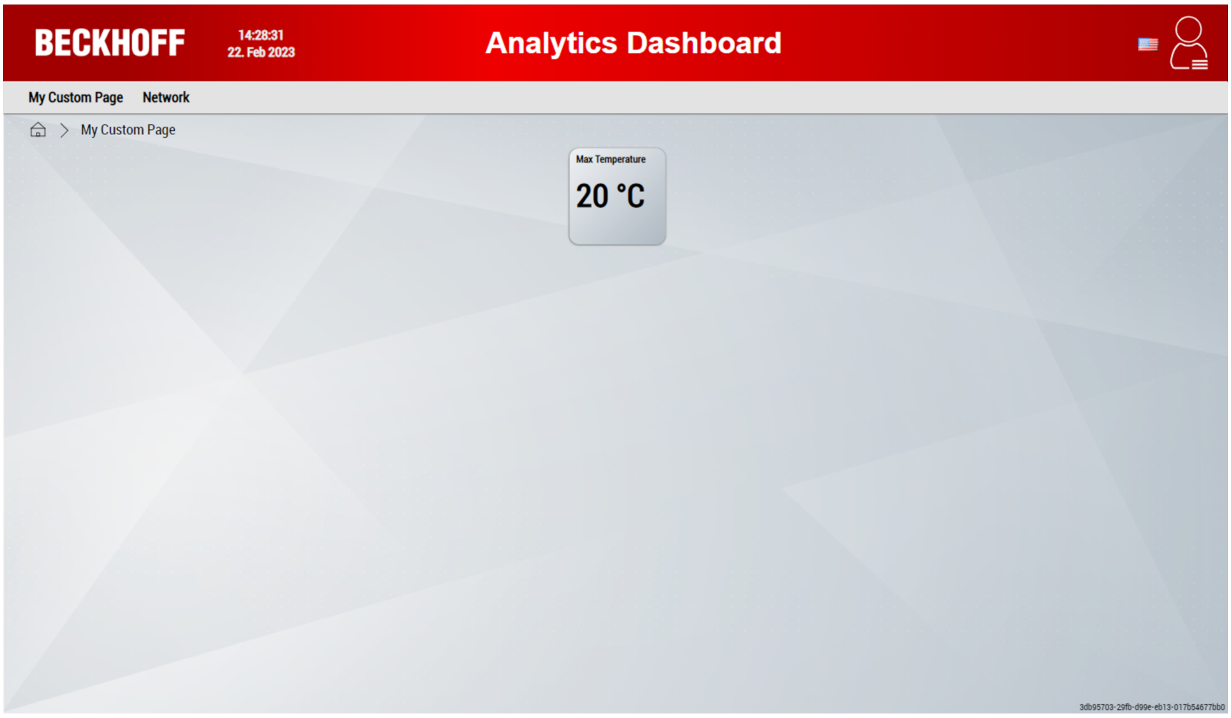
8. The **Max Temperature** control that was created appears in the **Dashboard** node. **Right-click** > **Rename** to rename the content.



Solution Explorer

```
Solution 'TwinCAT Measurement Project' (1 project)
  TwinCAT Measurement Project
    Analytics Project
      Network
        Min Max Avg 1Ch
        Threshold String Classifier 1Ch
      Dashboard
        My Custom Page
          Max Temperature
        Network
          Min Max Avg 1Ch
          Threshold String Classifier 1Ch
      Inputs
```

⇨ After a successful HMI and PLC generation, you can open the previously created **My Custom Page** via the navigation. There you can see the manually created control.



6.8.2 Use customized and own controls

HMI Control Mapping Wizard

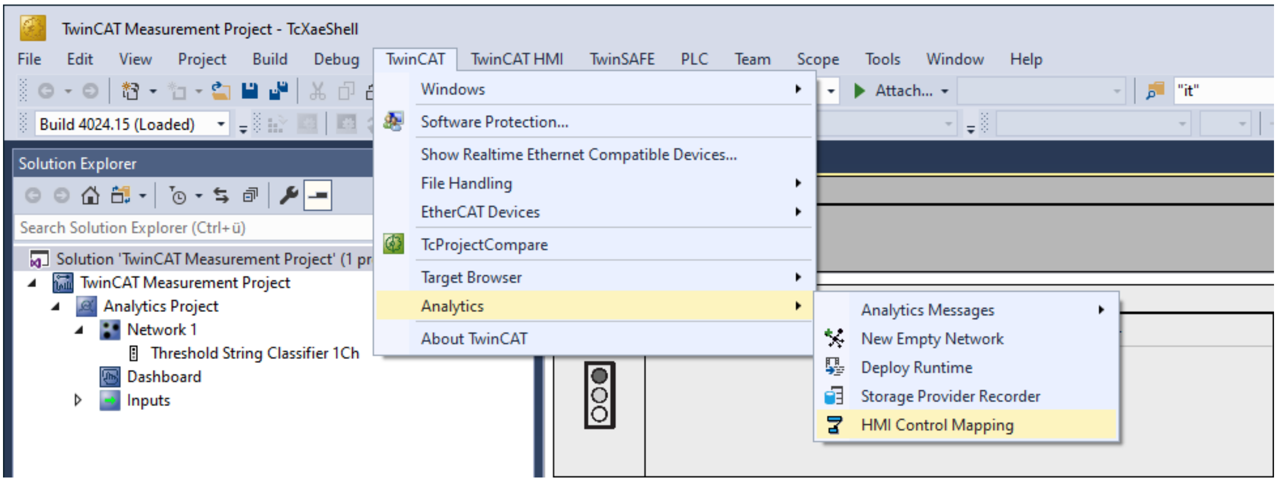
The HMI Control Mapping wizard enables the following:

1. Adding your own controls.

2. Mapping controls to module classes or changing existing mappings.

3. Mapping controls to module instances or virtual inputs (Analytics project must be open)

Open the wizard via the tab **TwinCAT > Analytics > HMI Control Mapping**.

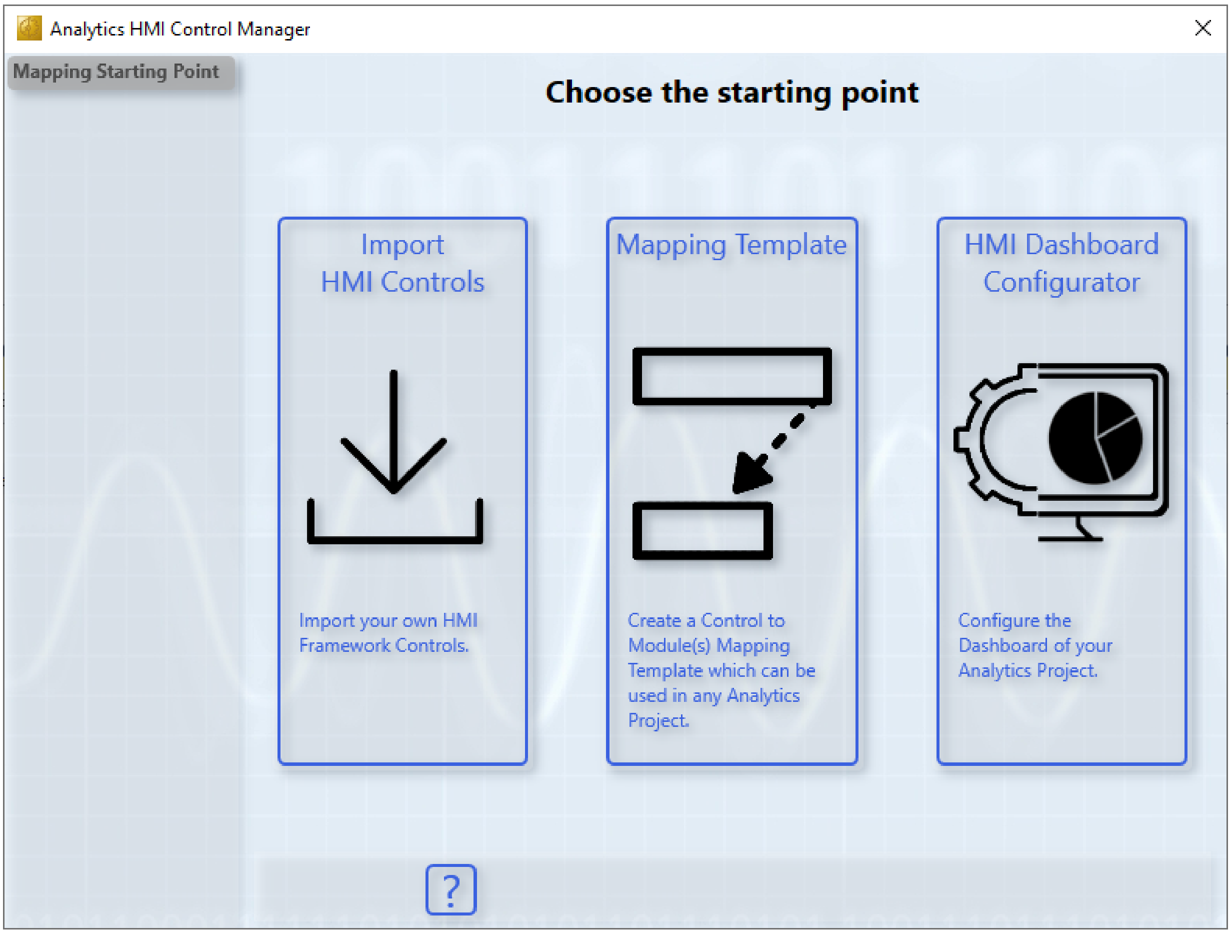


Additional help is offered via the question mark **?** in the wizard.

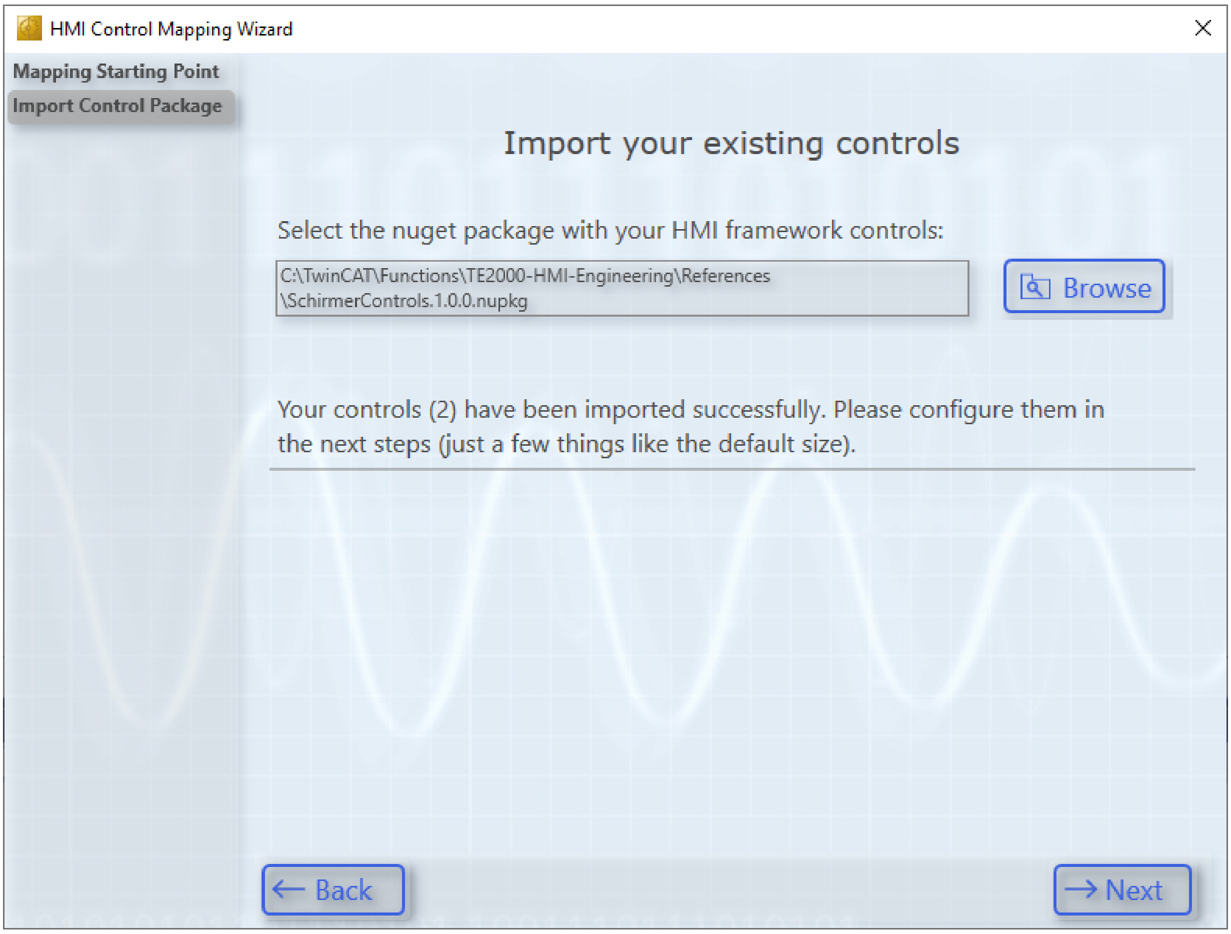
1. Adding your own Controls

The TwinCAT 3 HMI allows you to create your own HMI Framework controls and export them as a NuGet package.

1. To assign your own framework controls to the Analytics modules, click **Import HMI Controls**


Analytics HMI Control Manager
Mapping Starting Point
Choose the starting point
Import HMI Controls
Import your own HMI Framework Controls.
Mapping Template
Create a Control to Module(s) Mapping Template which can be used in any Analytics Project.
HMI Dashboard Configurator
Configure the Dashboard of your Analytics Project.
?

2. Select the NuGet package via **Browse**.



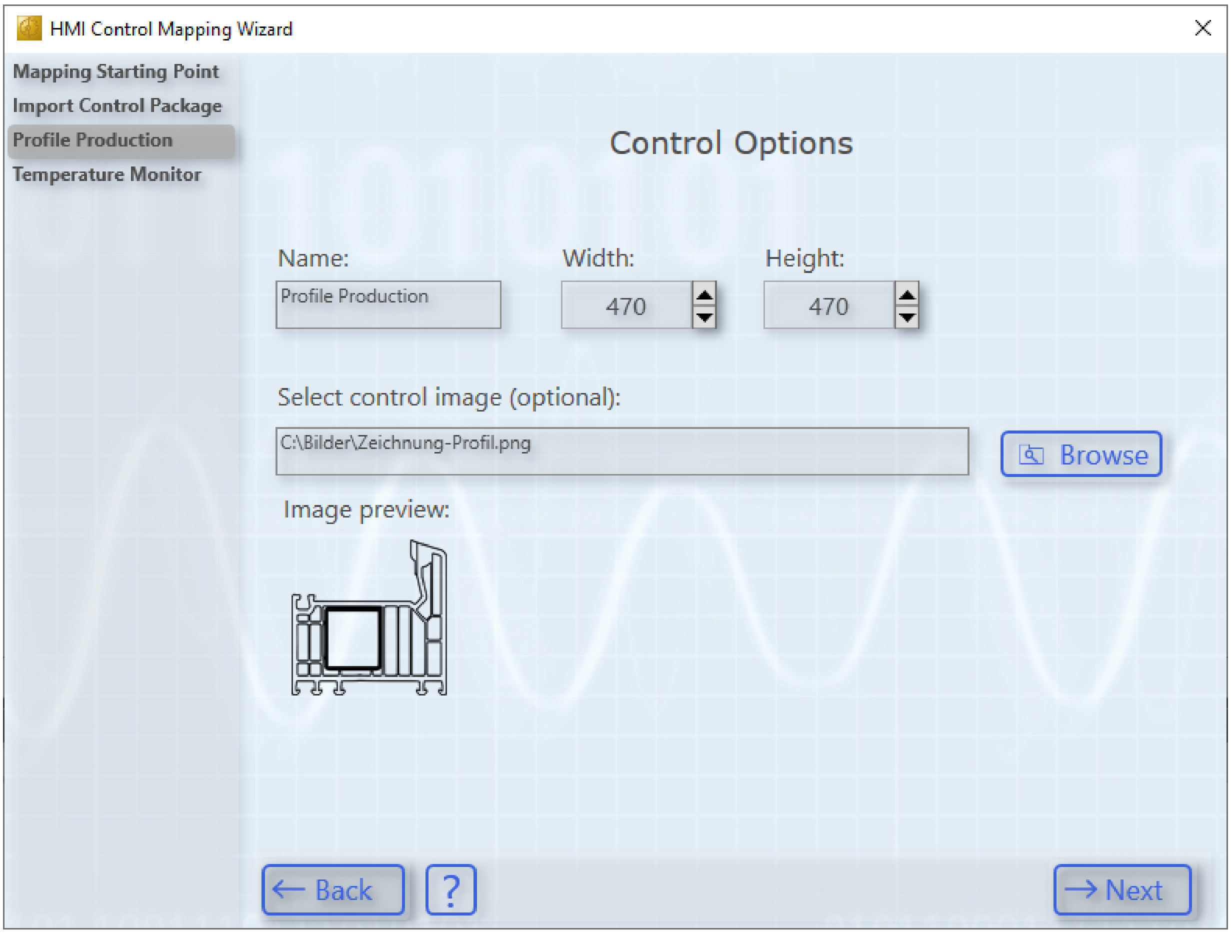
Import your existing controls

Select the nuget package with your HMI framework controls:

C:\TwinCAT\Functions\TE2000-HMI-Engineering\References\SchirmerControls.1.0.0.nupkg

Your controls (2) have been imported successfully. Please configure them in the next steps (just a few things like the default size).

3. Next, choose a name, size and image for your control. Then click **Next** and for the last control click **Create**.



Name: Profile Production

Width: 470

Height: 470

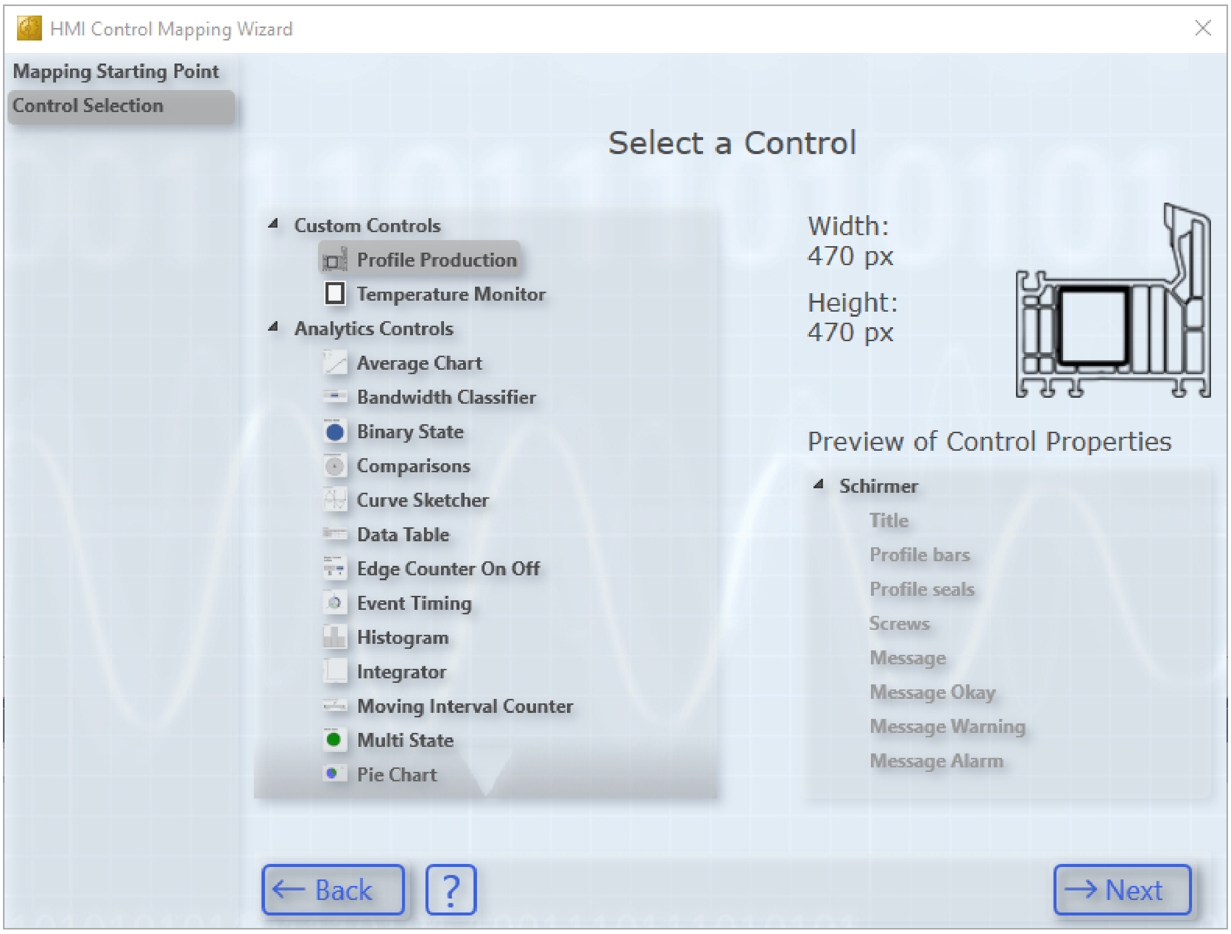
Select control image (optional): C:\Bilder\Zeichnung-Profil.png

⇨ You will automatically be redirected so that you can create a mapping between your controls and Analytics modules.

2. a) Mapping of controls to modules (further to 1.)

Under 2. b) this step is explained in more detail using "Binary State" control.

1. Now select your control.



HMI Control Mapping Wizard

Mapping Starting Point

Control Selection

Select a Control

- Custom Controls
  - Profile Production
  - Temperature Monitor
- Analytics Controls
  - Average Chart
  - Bandwidth Classifier
  - Binary State
  - Comparisons
  - Curve Sketcher
  - Data Table
  - Edge Counter On Off
  - Event Timing
  - Histogram
  - Integrator
  - Moving Interval Counter
  - Multi State
  - Pie Chart

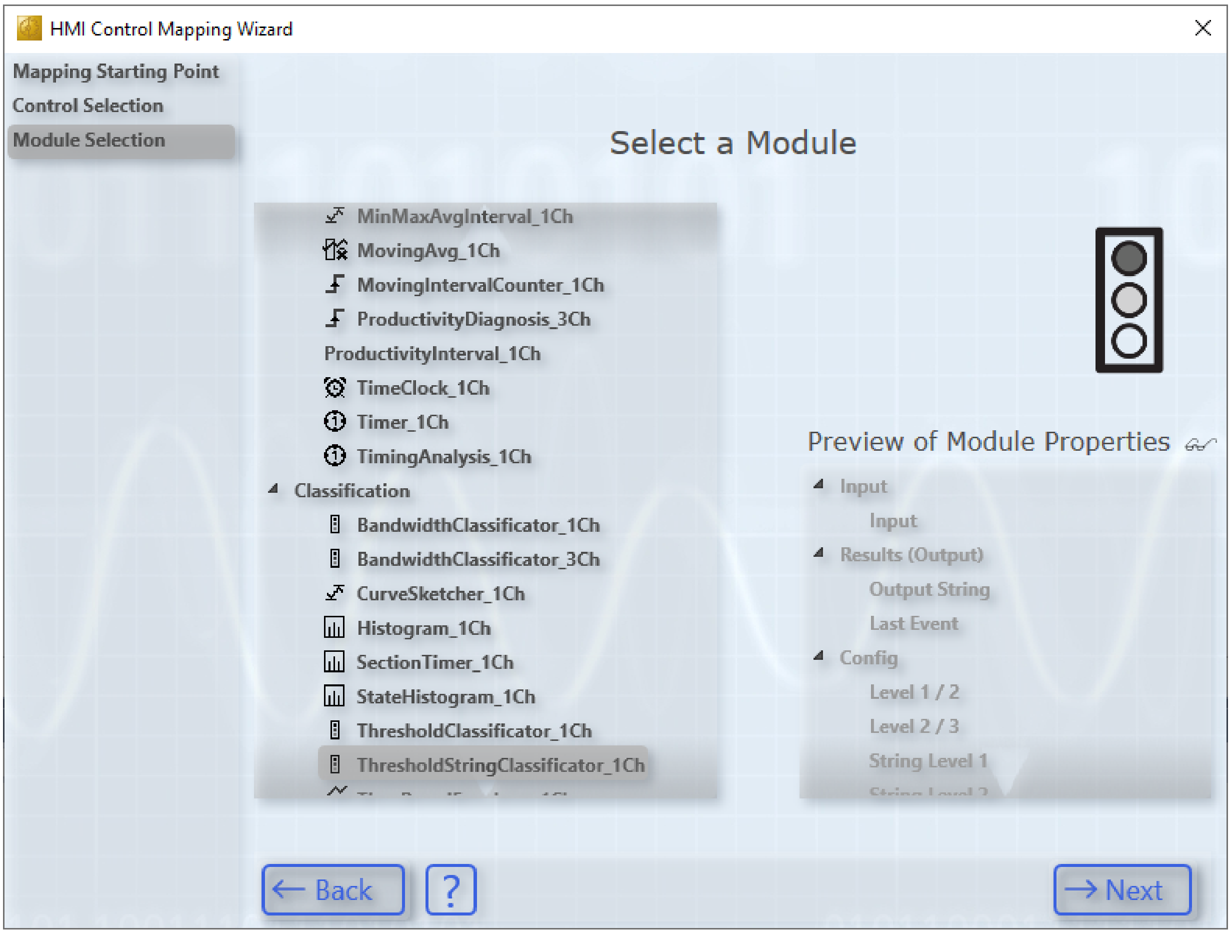
Width:
470 px

Height:
470 px

Preview of Control Properties

- Schirmer
  - Title
  - Profile bars
  - Profile seals
  - Screws
  - Message
  - Message Okay
  - Message Warning
  - Message Alarm

2. Now select the module to which you want to assign the control.



HMI Control Mapping Wizard

Mapping Starting Point

Control Selection

Module Selection

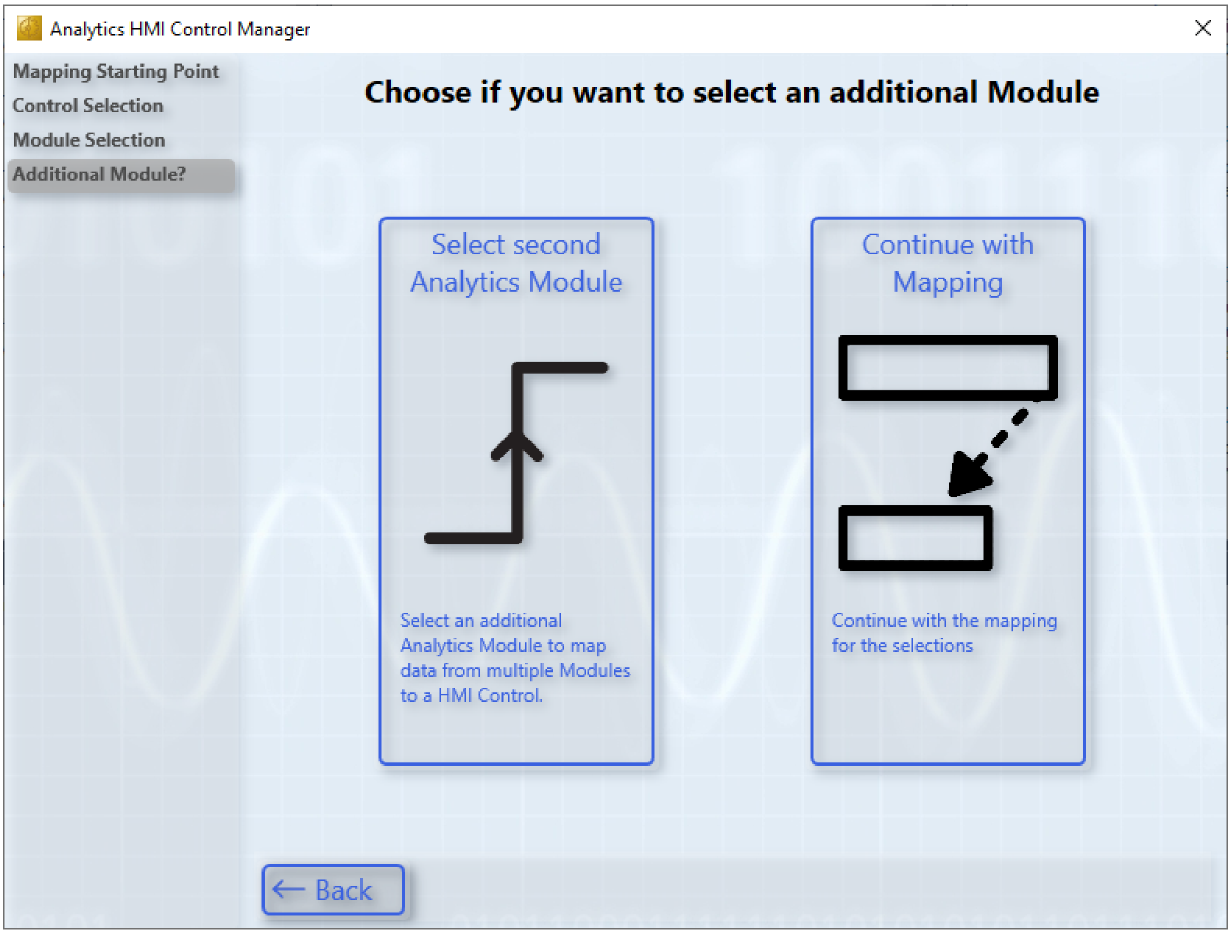
Select a Module

- MinMaxAvgInterval_1Ch
- MovingAvg_1Ch
- MovingIntervalCounter_1Ch
- ProductivityDiagnosis_3Ch
- ProductivityInterval_1Ch
- TimeClock_1Ch
- Timer_1Ch
- TimingAnalysis_1Ch
- Classification
  - BandwidthClassificator_1Ch
  - BandwidthClassificator_3Ch
  - CurveSketcher_1Ch
  - Histogram_1Ch
  - SectionTimer_1Ch
  - StateHistogram_1Ch
  - ThresholdClassificator_1Ch
  - ThresholdStringClassificator_1Ch

Preview of Module Properties

- Input
  - Input
- Results (Output)
  - Output String
  - Last Event
- Config
  - Level 1 / 2
  - Level 2 / 3
  - String Level 1

3. Select **Continue with Mapping**.



Choose if you want to select an additional Module

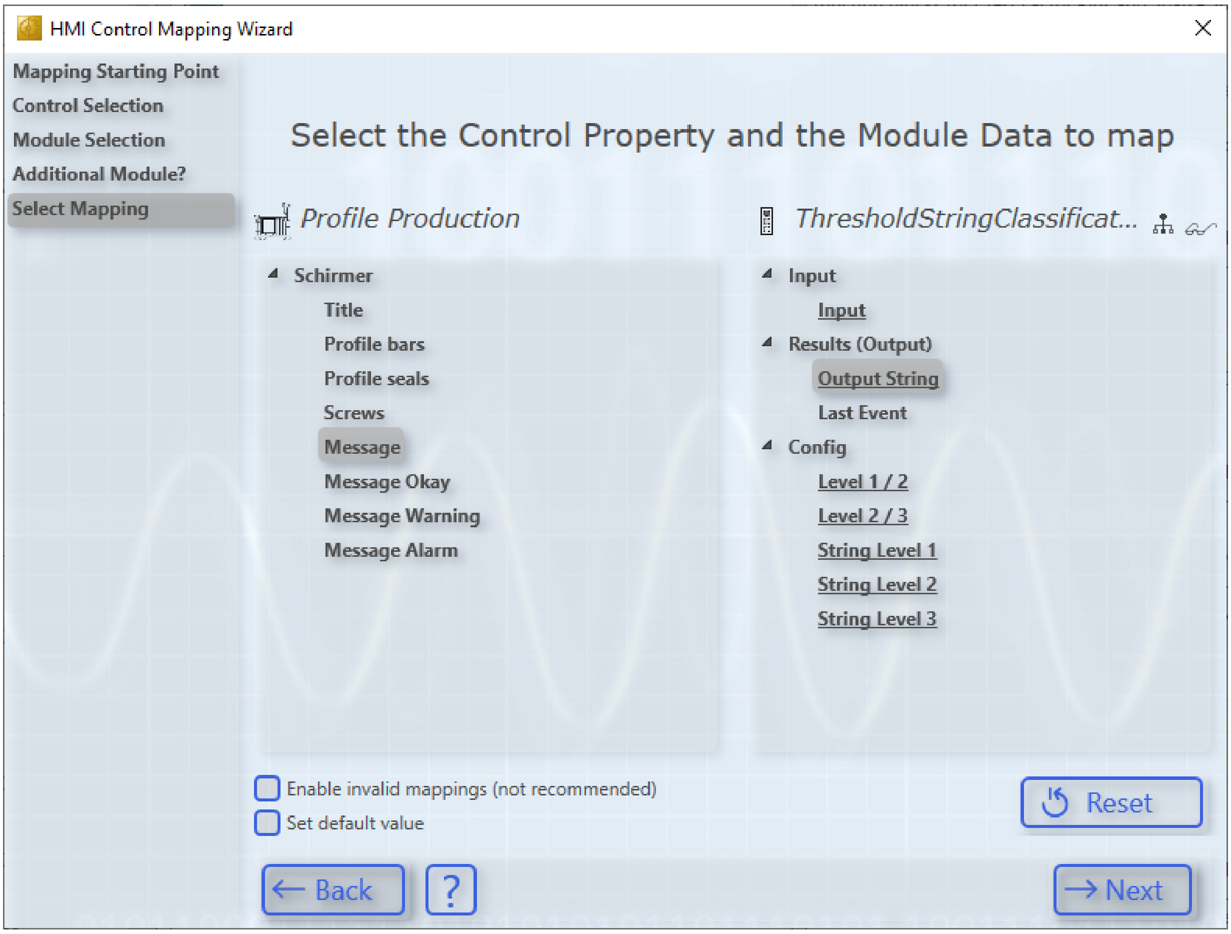
Select second Analytics Module

Select an additional Analytics Module to map data from multiple Modules to a HMI Control.

Continue with Mapping

Continue with the mapping for the selections

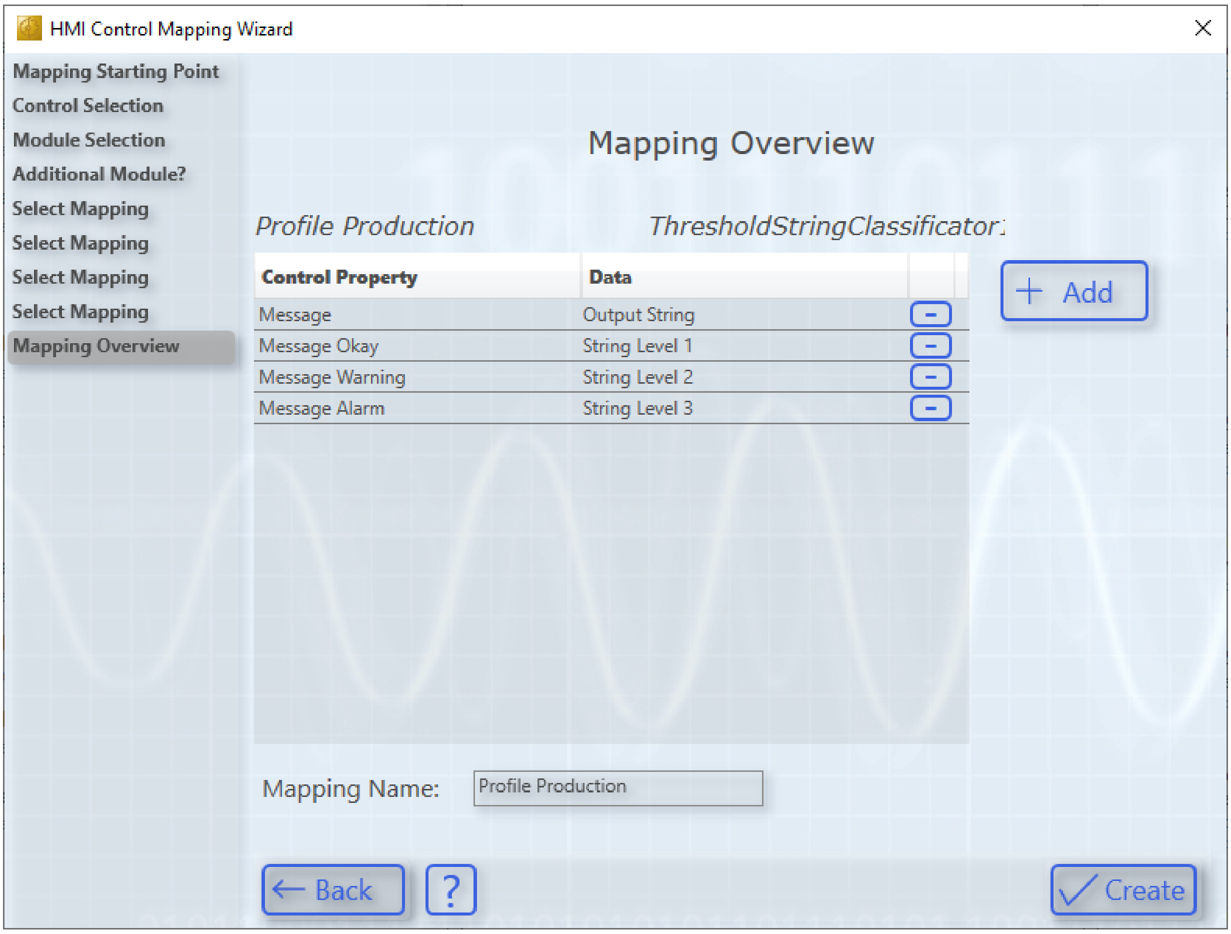
4. In the next steps, connect the control inputs to the module data.



Select the Control Property and the Module Data to map

□ Enable invalid mappings (not recommended)
□ Set default value

5. Last, click **Create** to add the mapping to the Analytics workbench.



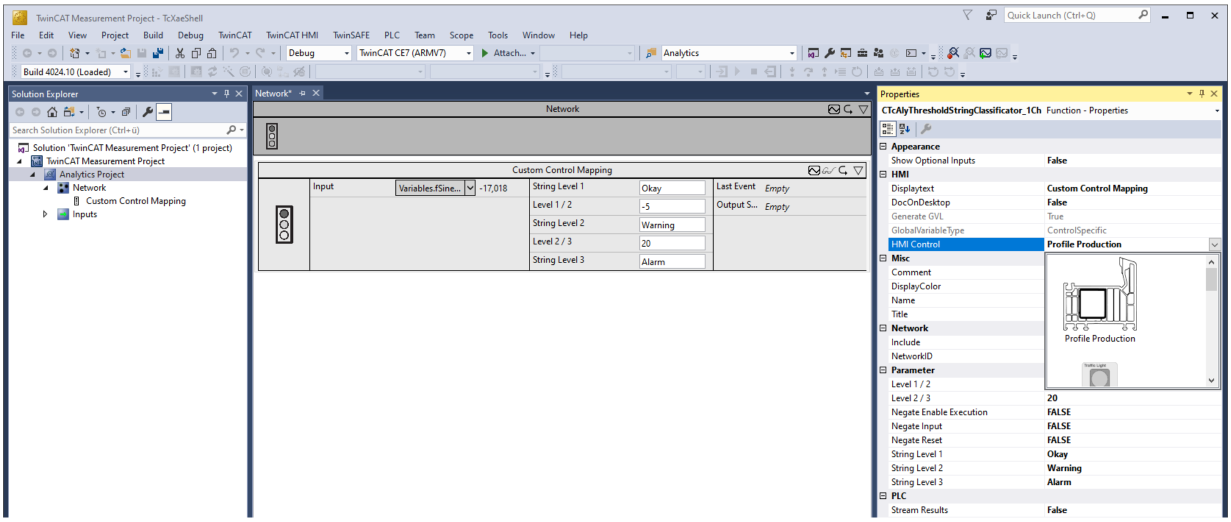
Mapping Overview

Profile Production *ThresholdStringClassificator1*

Control Property	Data
Message	Output String
Message Okay	String Level 1
Message Warning	String Level 2
Message Alarm	String Level 3

Mapping Name: Profile Production

6. You can now close the wizard and the mapping will automatically be available for the module. Select it and generate your dashboard.


TwinCAT Measurement Project - TcXaeShell
File Edit View Project Build Debug TwinCAT TwinCAT HMI TwinSAFE PLC Team Scope Tools Window Help
Debug
TwinCAT CE7 (ARMV7)
Attach...
Analytics
Build 4024.10 (Loaded)
Quick Launch (Ctrl+Q)
Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'TwinCAT Measurement Project' (1 project)
TwinCAT Measurement Project
Analytics Project
Network
Custom Control Mapping
Inputs
Network*
Network
Profile Production
Traffic Light

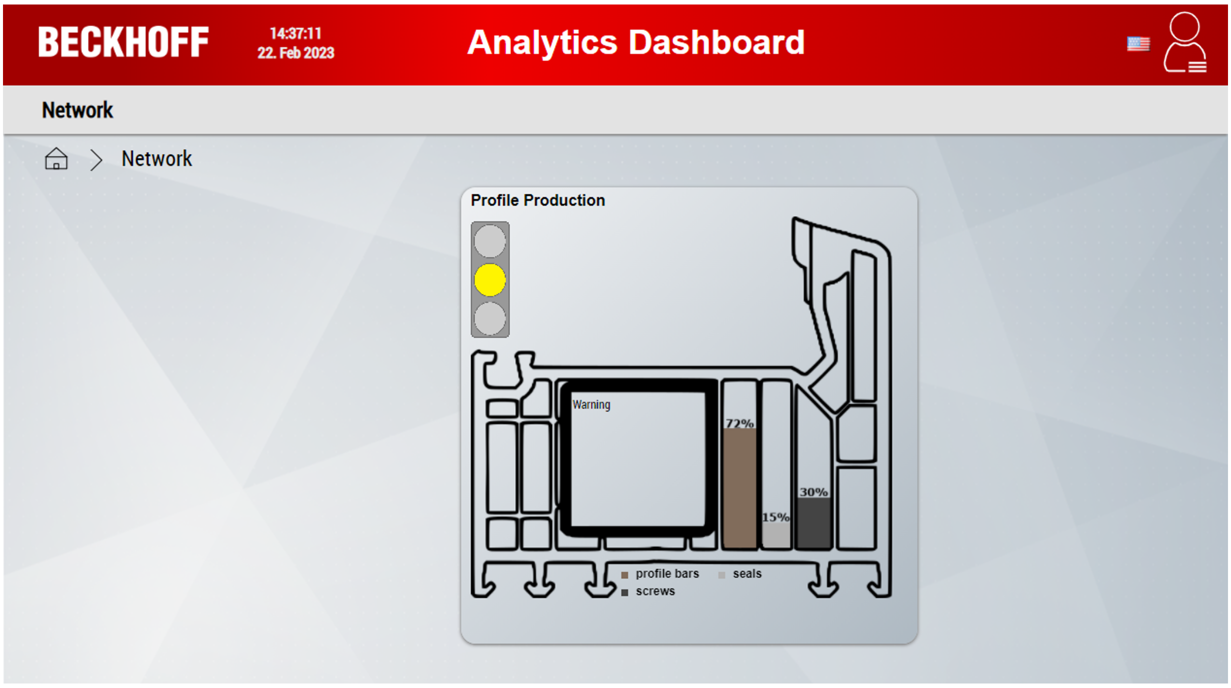
Custom Control Mapping

Input: Variables.fSine... -17,018	String Level 1: Okay	Last Event: *Empty*
	Level 1 / 2: -5	Output S...: *Empty*
	String Level 2: Warning	
	Level 2 / 3: 20	
	String Level 3: Alarm	

Properties

CTcAlyThresholdStringClassificator_1Ch Function - Properties

Appearance	
Show Optional Inputs	False
HMI	
Displaytext	Custom Control Mapping
DocOnDesktop	False
Generate GVL	True
GlobalVariableType	ControlSpecific
HMI Control	Profile Production
Misc	
Comment	
DisplayColor	
Name	
Title	
Network	
Include	
NetworkID	
Parameter	
Level 1 / 2	
Level 2 / 3	20
Negate Enable Execution	FALSE
Negate Input	FALSE
Negate Reset	FALSE
String Level 1	Okay
String Level 2	Warning
String Level 3	Alarm
PLC	
Stream Results	False

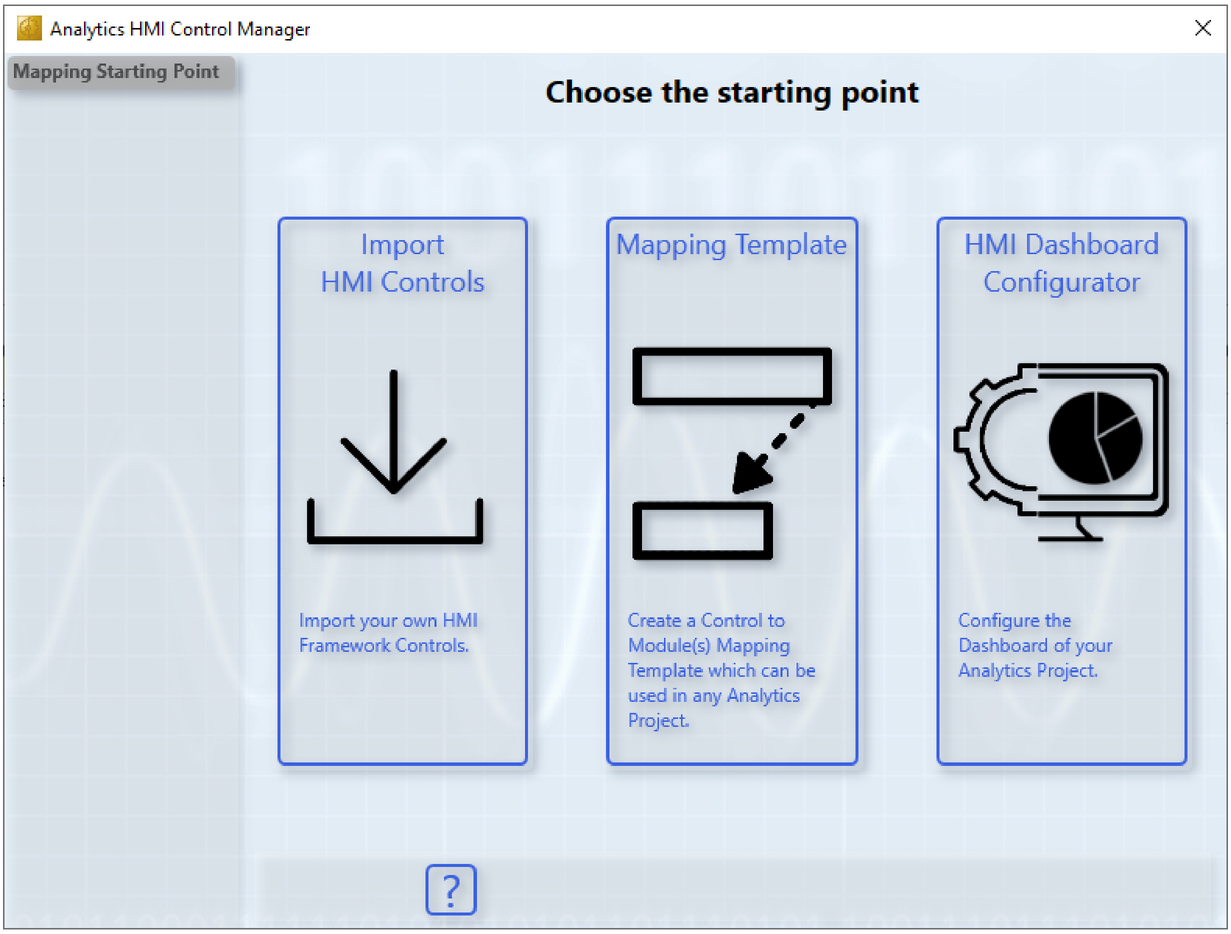


⇨ The dashboard is ready.

2.b) Mapping controls to modules (without own controls)

Analogous to 2.a), this section describes how to perform a mapping between a control and a module. An existing Analytics Control is used as an example. You can try this example directly.

1. To do this, select the item **Mapping Template**.



Analytics HMI Control Manager

Choose the starting point

Import HMI Controls

Import your own HMI Framework Controls.

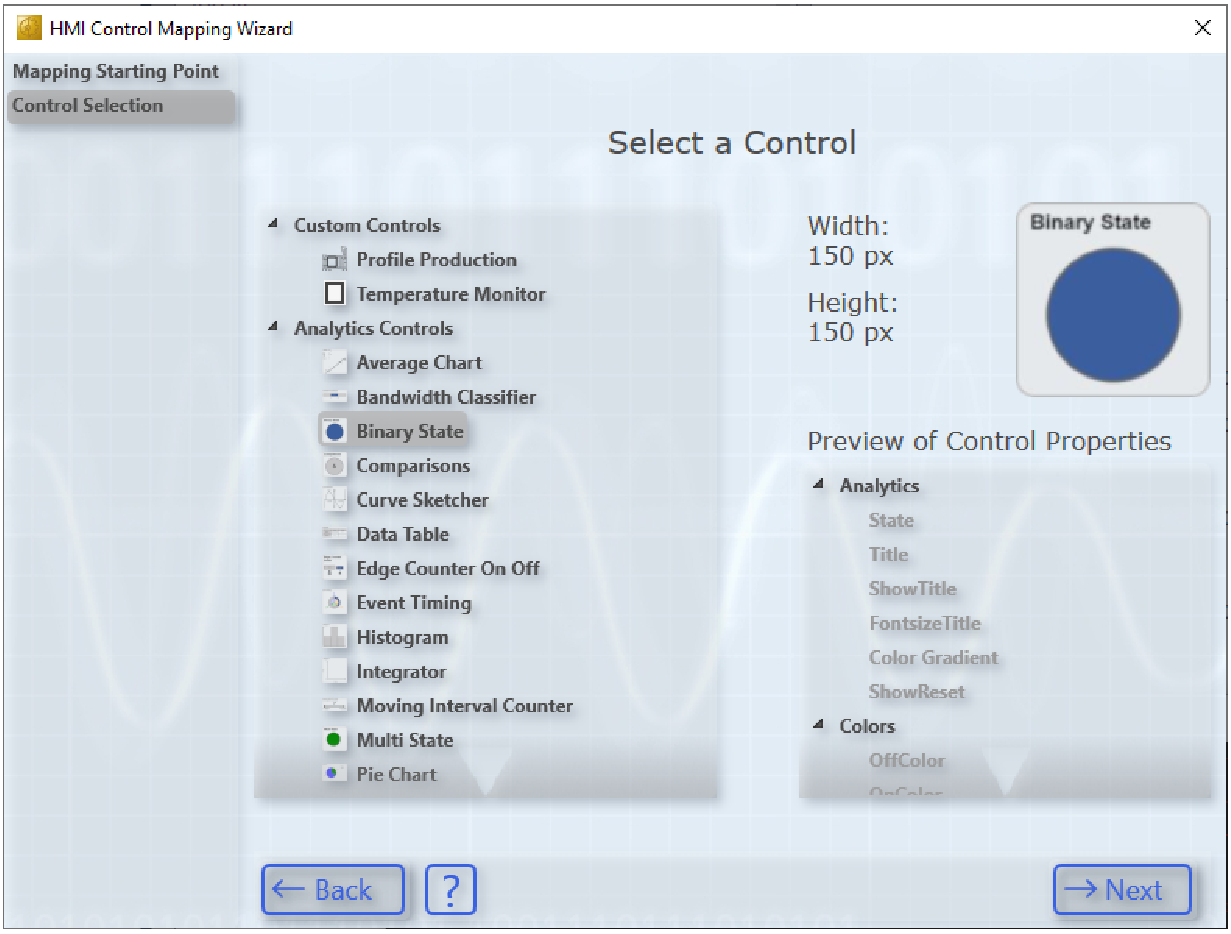
Mapping Template

Create a Control to Module(s) Mapping Template which can be used in any Analytics Project.

HMI Dashboard Configurator

Configure the Dashboard of your Analytics Project.

2. Select a control. The properties of the control are displayed on the right. It also shows how big the control will be on the dashboard.



HMI Control Mapping Wizard

Mapping Starting Point
Control Selection

Select a Control

- Custom Controls
  - Profile Production
  - Temperature Monitor
- Analytics Controls
  - Average Chart
  - Bandwidth Classifier
  - Binary State
  - Comparisons
  - Curve Sketcher
  - Data Table
  - Edge Counter On Off
  - Event Timing
  - Histogram
  - Integrator
  - Moving Interval Counter
  - Multi State
  - Pie Chart

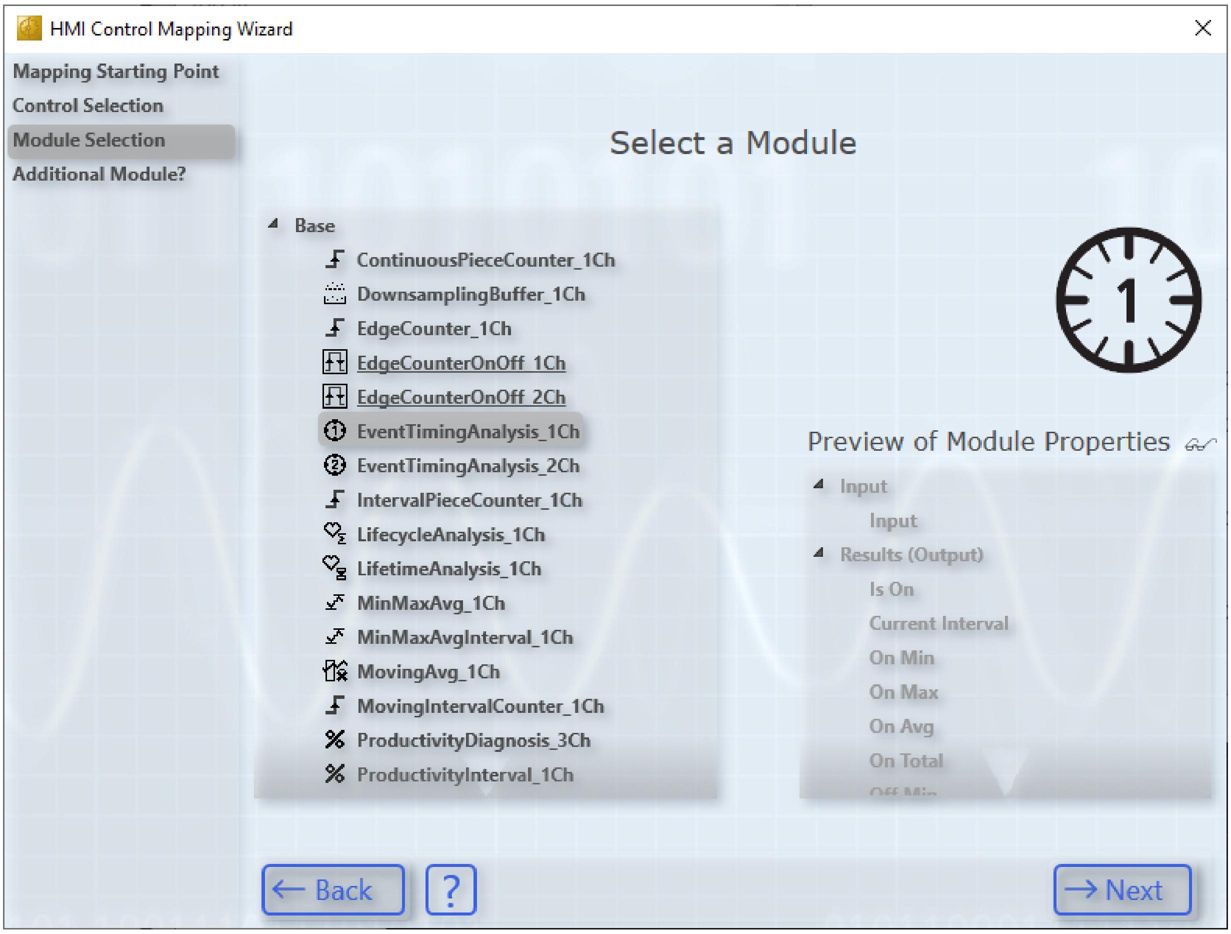
Width:
150 px

Height:
150 px

Preview of Control Properties

- Analytics
  - State
  - Title
  - ShowTitle
  - FontsizeTitle
  - Color Gradient
  - ShowReset
- Colors
  - OffColor
  - OnColor

3. Select the module from which you want to display the data. The inputs, outputs and parameters are displayed as a preview, which can then be linked to the properties of the selected control. Modules that have already been mapped are underlined. These mappings can also be edited.



HMI Control Mapping Wizard

Mapping Starting Point
Control Selection
Module Selection
Additional Module?

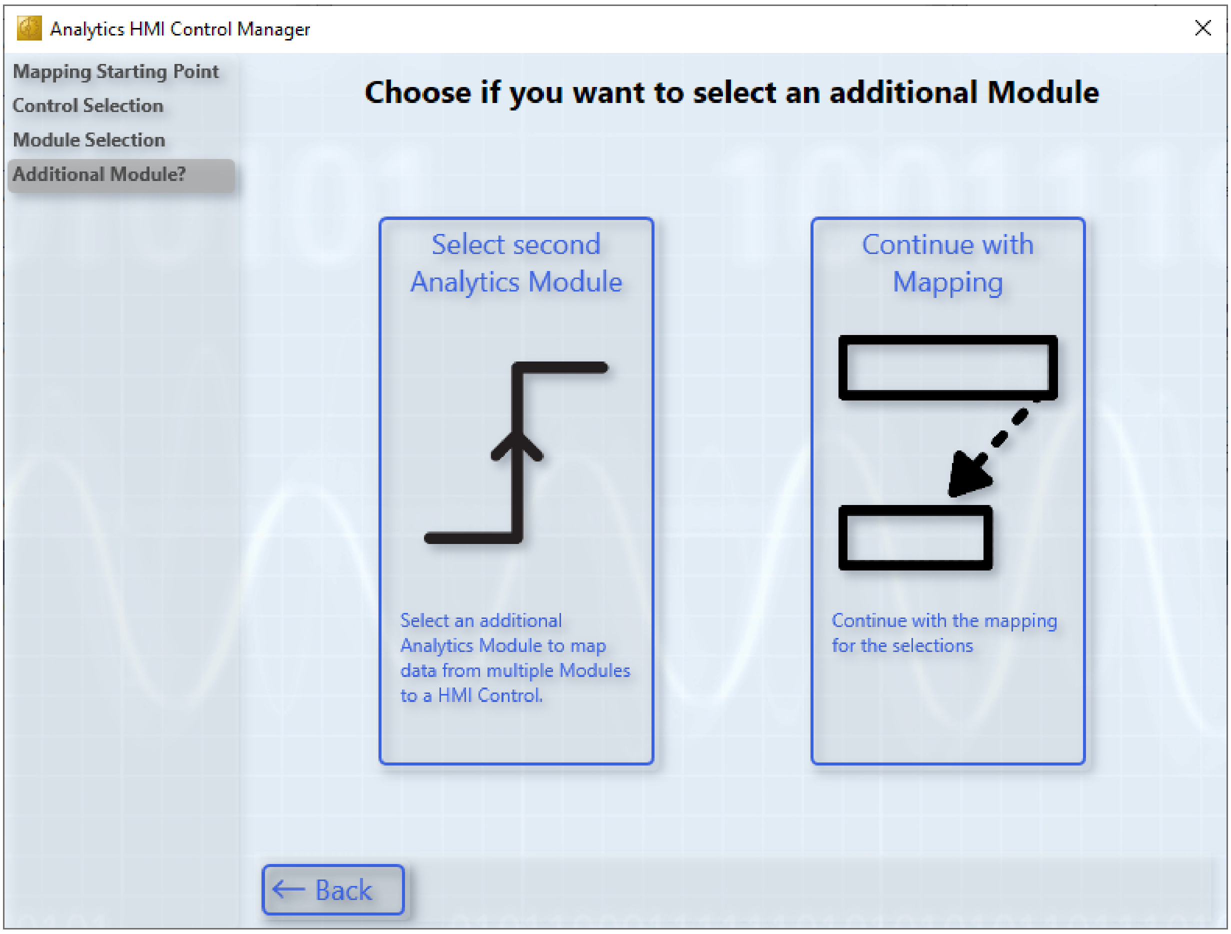
Select a Module

- Base
  - ContinuousPieceCounter_1Ch
  - DownsamplingBuffer_1Ch
  - EdgeCounter_1Ch
  - EdgeCounterOnOff_1Ch
  - EdgeCounterOnOff_2Ch
  - EventTimingAnalysis_1Ch
  - EventTimingAnalysis_2Ch
  - IntervalPieceCounter_1Ch
  - LifecycleAnalysis_1Ch
  - LifetimeAnalysis_1Ch
  - MinMaxAvg_1Ch
  - MinMaxAvgInterval_1Ch
  - MovingAvg_1Ch
  - MovingIntervalCounter_1Ch
  - ProductivityDiagnosis_3Ch
  - ProductivityInterval_1Ch

Preview of Module Properties

- Input
  - Input
- Results (Output)
  - Is On
  - Current Interval
  - On Min
  - On Max
  - On Avg
  - On Total
  - Off Min

4. In the future, a control will be able to display data from different modules. Since this is not currently possible, select **Continue with Mapping**.



Choose if you want to select an additional Module

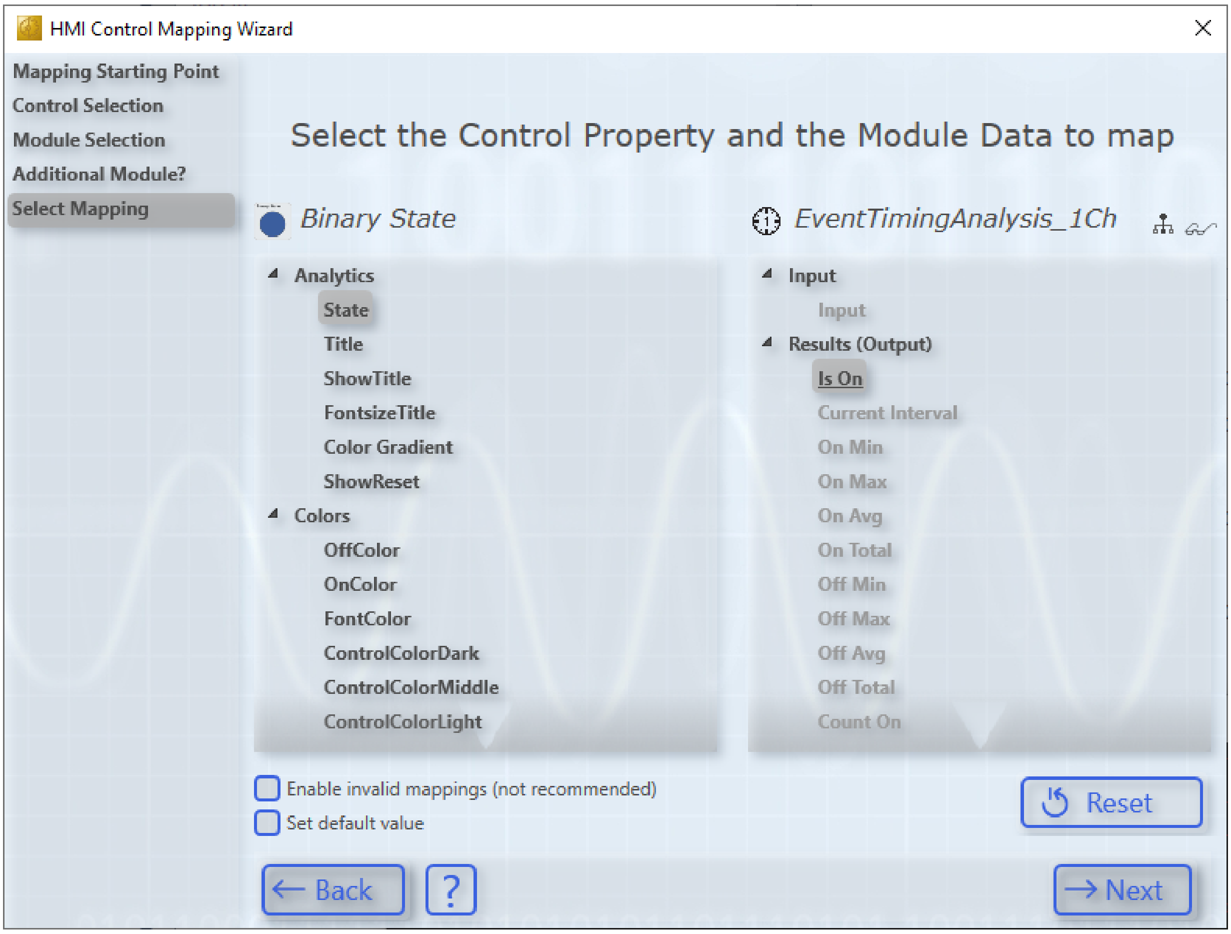
Select second Analytics Module

Select an additional Analytics Module to map data from multiple Modules to a HMI Control.

Continue with Mapping

Continue with the mapping for the selections

5. Next, select a control property to be mapped. All impossible variables that cannot be assigned due to their data type are grayed out. All others can be mapped.
   Now select the module variable you want to map with the control property.



HMI Control Mapping Wizard

Mapping Starting Point
Control Selection
Module Selection
Additional Module?
Select Mapping

Select the Control Property and the Module Data to map

Binary State

- Analytics
  - State
  - Title
  - ShowTitle
  - FontsizeTitle
  - Color Gradient
  - ShowReset
- Colors
  - OffColor
  - OnColor
  - FontColor
  - ControlColorDark
  - ControlColorMiddle
  - ControlColorLight

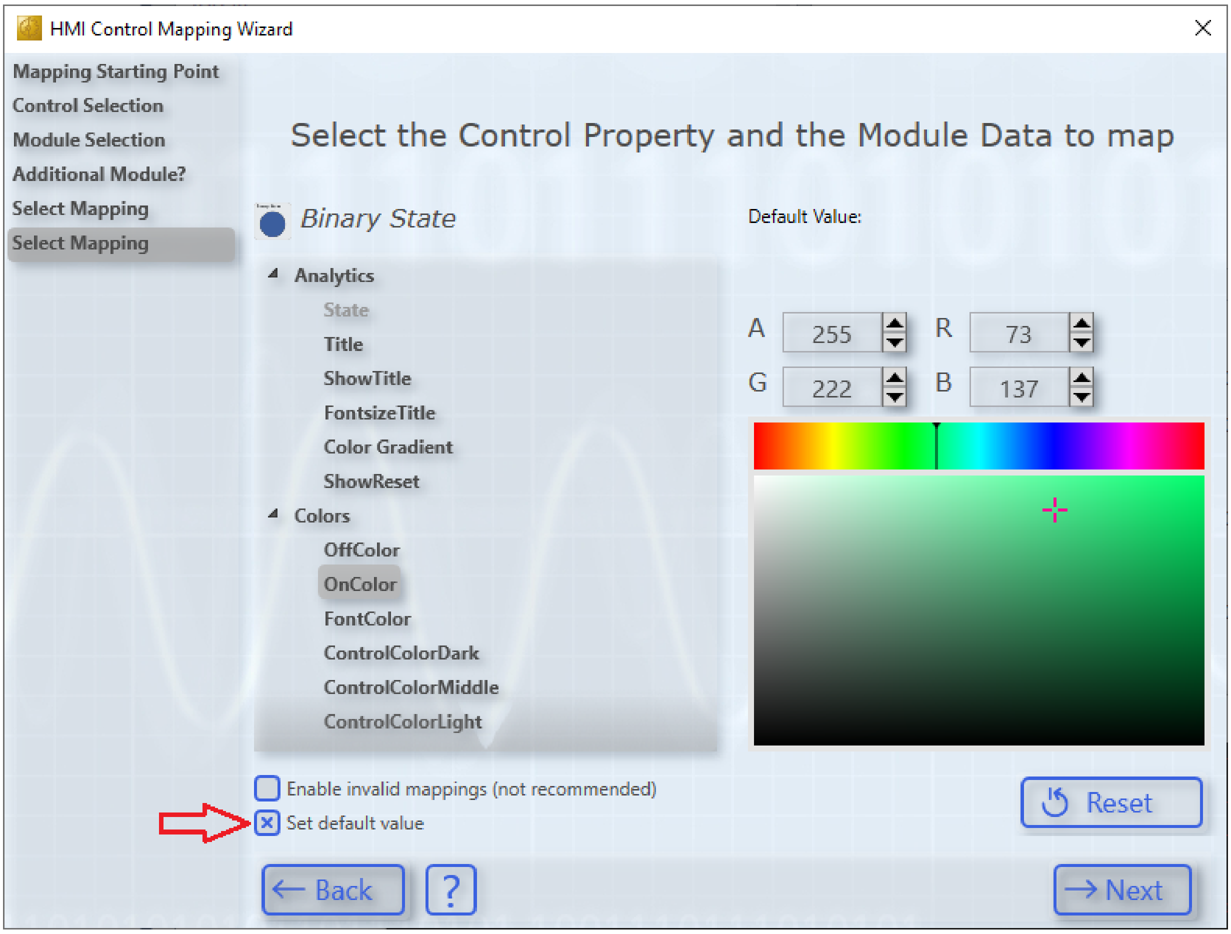
EventTimingAnalysis_1Ch

- Input
  - Input
- Results (Output)
  - Is On
  - Current Interval
  - On Min
  - On Max
  - On Avg
  - On Total
  - Off Min
  - Off Max
  - Off Avg
  - Off Total
  - Count On

□ Enable invalid mappings (not recommended)
□ Set default value

6. Likewise, you can check the **Set default value** checkbox to assign a default value to the control. This can be used, for example, to change default colors or to set Boolean values such as "ShowTitle" to "False" if no title is to be displayed in the Analytics control.

In this example, the "OnColor" is changed. Based on the data type, a specific selection option is provided.



HMI Control Mapping Wizard

Select the Control Property and the Module Data to map

Binary State

Default Value:

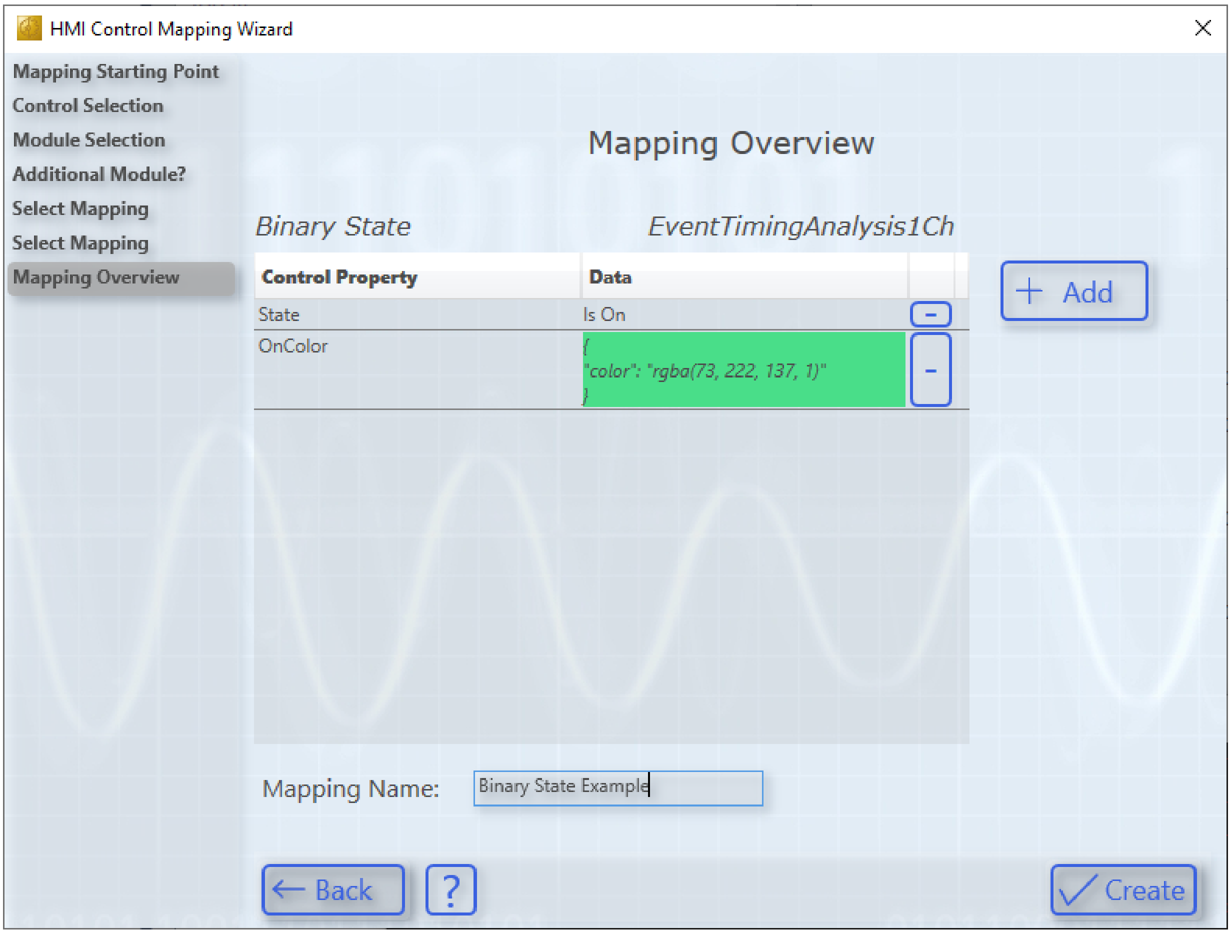
A 255 R 73

G 222 B 137

□ Enable invalid mappings (not recommended)

☒ Set default value

7. As far as you do not want to add any more entries via **Add**, you can complete the mapping via **Create**. It is recommended to assign a meaningful mapping name.

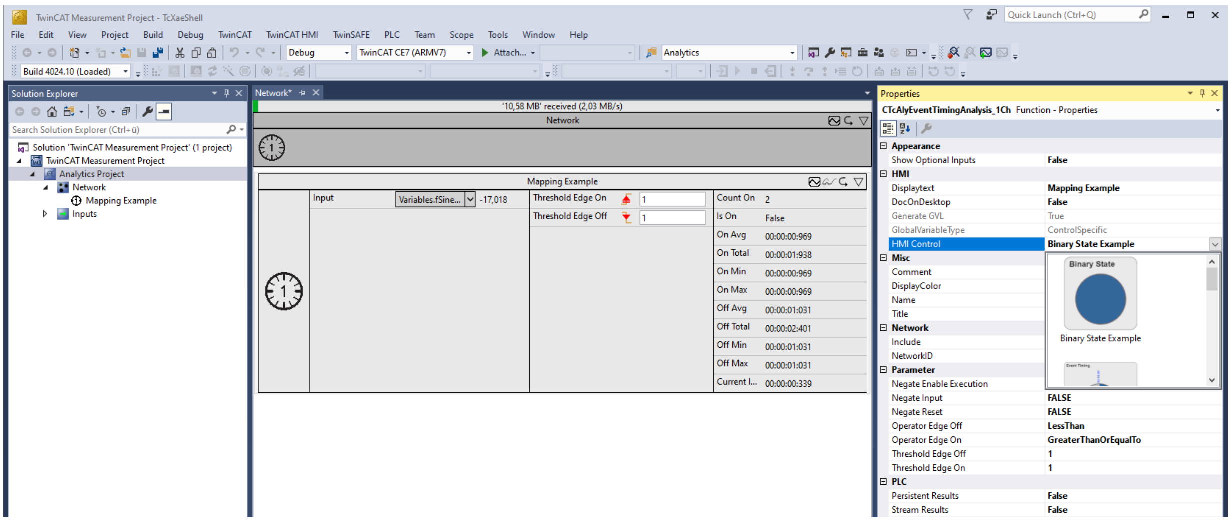


Control Property	Data
State	Is On

OnColor	{
	"color": "rgba(73, 222, 137, 1)"
	}

Mapping Name: Binary State Example

8. You can close the wizard and the mapping will automatically be available for the module. Select it and generate your dashboard.

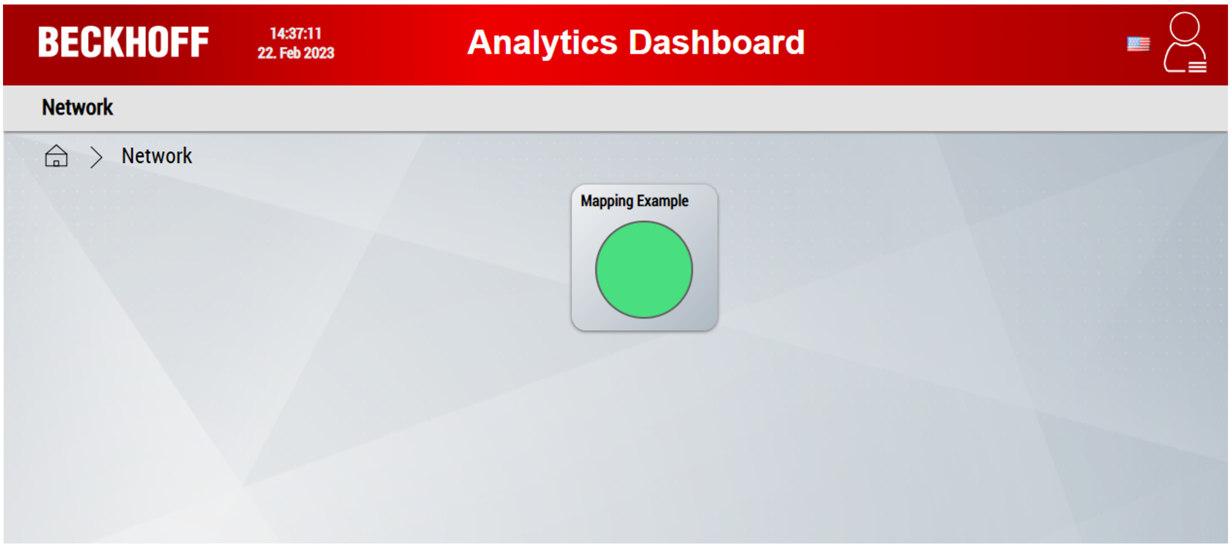

TwinCAT Measurement Project - TcXaeShell
File Edit View Project Build Debug TwinCAT TwinCAT HMI TwinSAFE PLC Team Scope Tools Window Help
Debug
TwinCAT CE7 (ARMV7)
Attach...
Analytics
Quick Launch (Ctrl+Q)
Build 4024.10 (Loaded)
Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'TwinCAT Measurement Project' (1 project)
TwinCAT Measurement Project
Analytics Project
Network
Mapping Example
Inputs
Network*
'10,58 MB' received (2,03 MB/s)
Network
Mapping Example

Input	Variables.fSine...	-17,018	Threshold Edge On	1	Count On 2
			Threshold Edge Off	1	Is On False
					On Avg 00:00:00:969
					On Total 00:00:01:938
					On Min 00:00:00:969
					On Max 00:00:00:969
					Off Avg 00:00:01:031
					Off Total 00:00:02:401
					Off Min 00:00:01:031
					Off Max 00:00:01:031
					Current I... 00:00:00:339

Properties

CTcAlyEventTimingAnalysis_1Ch Function - Properties

Appearance	
Show Optional Inputs	False
HMI	
Displaytext	Mapping Example
DocOnDesktop	False
Generate GVL	True
GlobalVariableType	ControlSpecific
HMI Control	Binary State Example
Misc	
Comment	
DisplayColor	
Name	
Title	
Network	
Include	
NetworkID	
Parameter	
Negate Enable Execution	
Negate Input	FALSE
Negate Reset	FALSE
Operator Edge Off	LessThan
Operator Edge On	GreaterThanOrEqualTo
Threshold Edge Off	1
Threshold Edge On	1
PLC	
Persistent Results	False
Stream Results	False

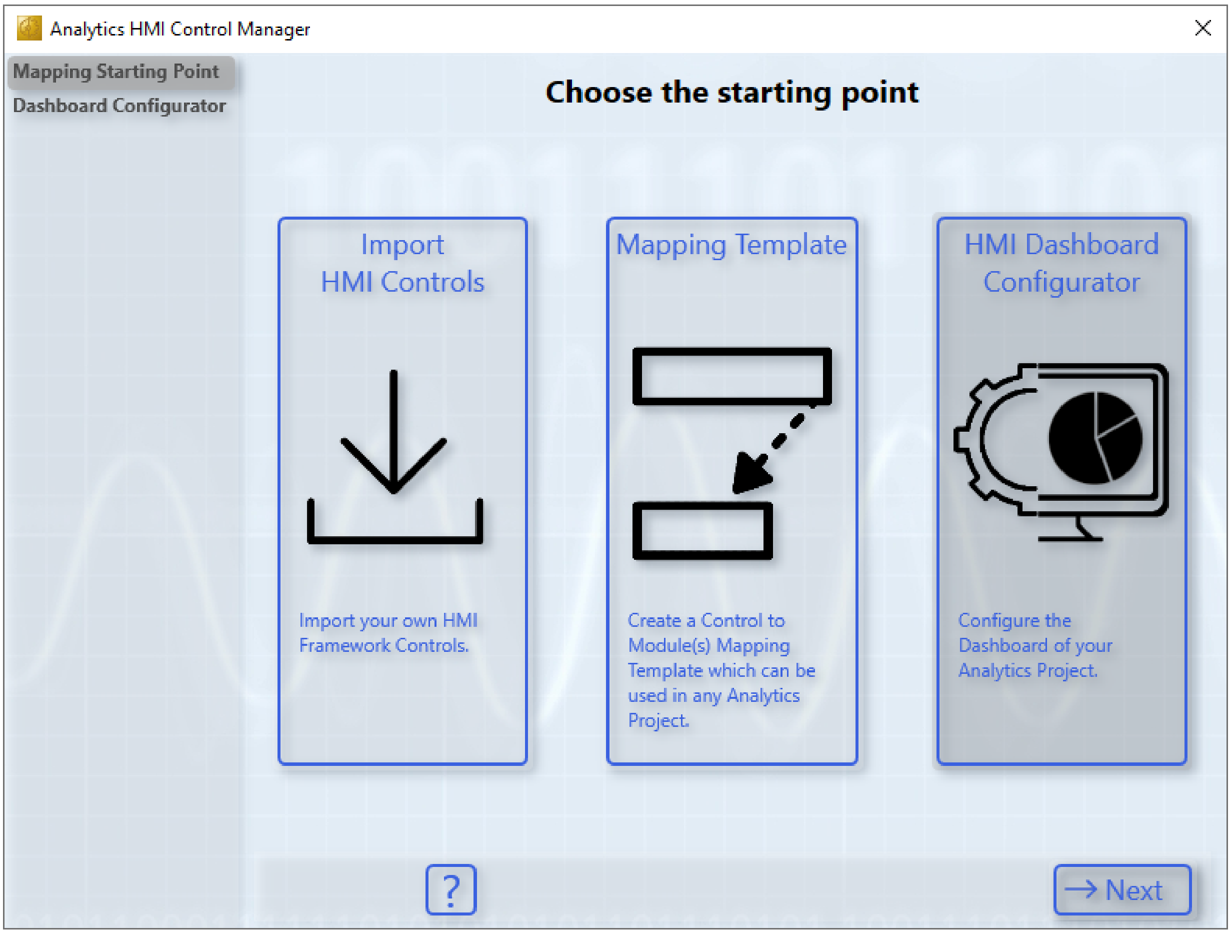


⇨ The dashboard is ready.

3. Mapping of controls to module instances

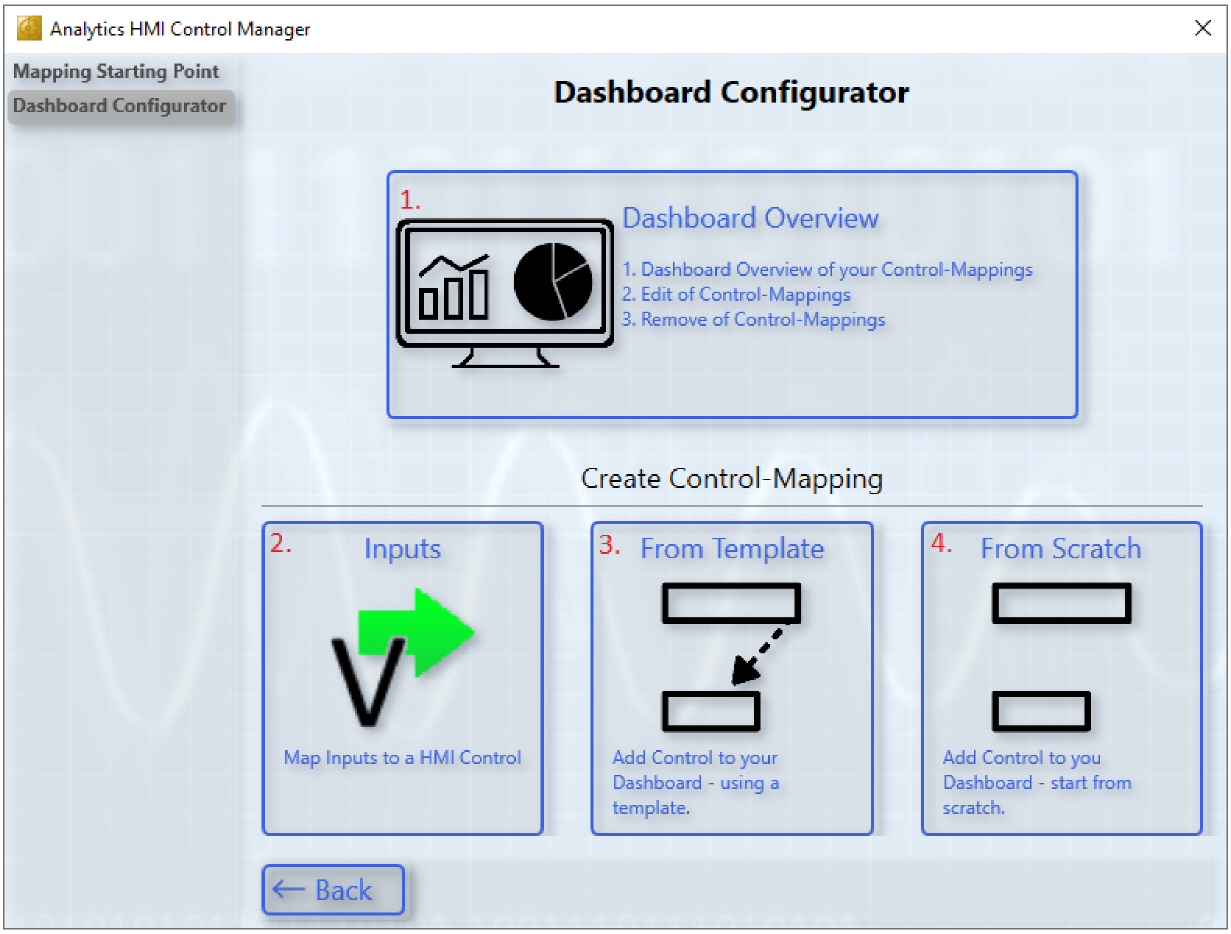
In addition to mapping controls to module classes, the module instances of a project can also be directly linked to controls. This is possible via the Manage dashboard structure and content in Analytics project [▶ 422] as well as via the wizard.

1. At the beginning of the wizard, click **HMI Dashboard Configurator**.



Analytics HMI Control Manager
Mapping Starting Point
Dashboard Configurator
Choose the starting point
Import HMI Controls
Import your own HMI Framework Controls.
Mapping Template
Create a Control to Module(s) Mapping Template which can be used in any Analytics Project.
HMI Dashboard Configurator
Configure the Dashboard of your Analytics Project.
?
Next

2. There you have several choices.



Analytics HMI Control Manager
Mapping Starting Point
Dashboard Configurator
Dashboard Configurator
1.
Dashboard Overview
1. Dashboard Overview of your Control-Mappings
2. Edit of Control-Mappings
3. Remove of Control-Mappings
Create Control-Mapping
2.
Inputs
Map Inputs to a HMI Control
3.
From Template
Add Control to your Dashboard - using a template.
4.
From Scratch
Add Control to you Dashboard - start from scratch.
Back

1. Overview and editing options for all controls
    - This function is also available by double-clicking a control in the Solution Explorer.
2. Adding a control that displays your input data
    - This function is also possible via the Manage dashboard structure and content in Analytics project [▶ 422].
3. Adding a control that displays module data (selection via a template)
    - This function can also be set for an individual module via the *Properties* window. This is the only way to display data from several modules.
    - You can see directly which data from the template are linked and can adjust them directly.
4. Adding a control that displays data from modules (fully manual)
    - This function is also possible via the Manage dashboard structure and content in Analytics project [▶ 422].

For the individual points you have to follow the steps of the wizard. The steps are the same or very similar to section "2. Mapping of controls to modules".

6.8.3 Configure user management and access rights

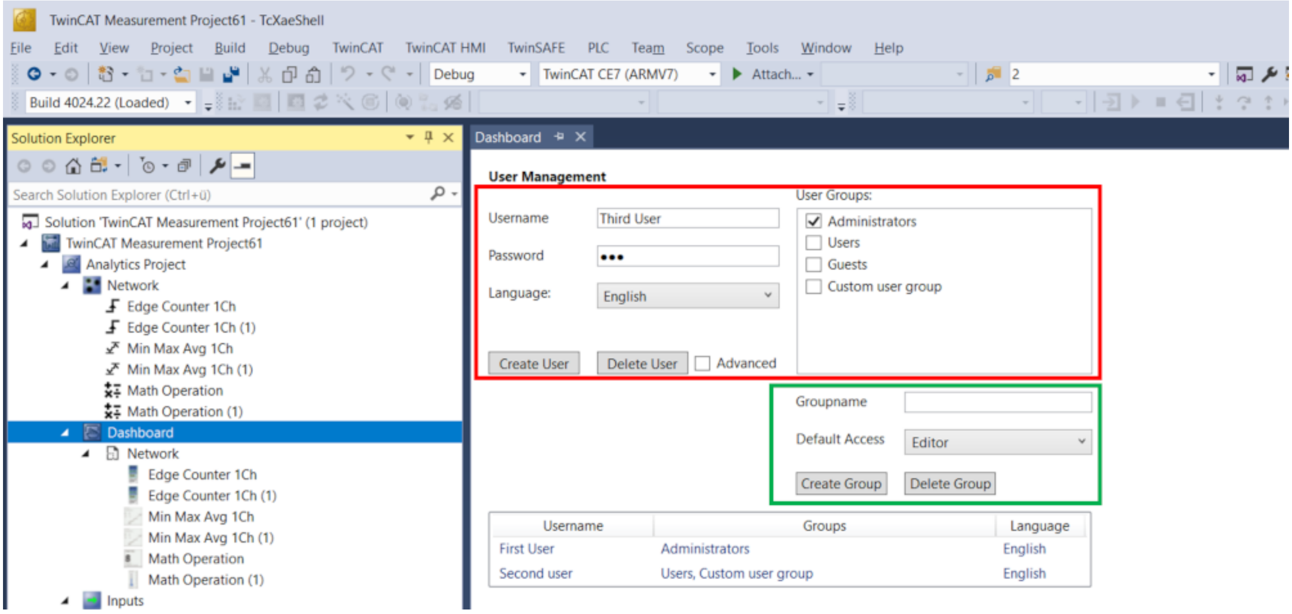
Users and user groups can be generated individually via the user management. Access rights to contents and controls can be defined for user groups.

Note that user management and access rights are only transferred to the HMI project from Dashboard version 2.0.

6.8.3.1 Definition in the Analytics Project

Defined users, user groups and access rights are automatically transferred to the HMI project during code generation. Changes to users and access rights can be made in the web browser after generation.

6.8.3.1.1 User management



Username: Third User

Password: •••

Language: English

User Groups:
- ☑ Administrators
- ☐ Users
- ☐ Guests
- ☐ Custom user group

☐ Advanced

Groupname:

Default Access: Editor

Username	Groups	Language
First User	Administrators	English
Second user	Users, Custom user group	English

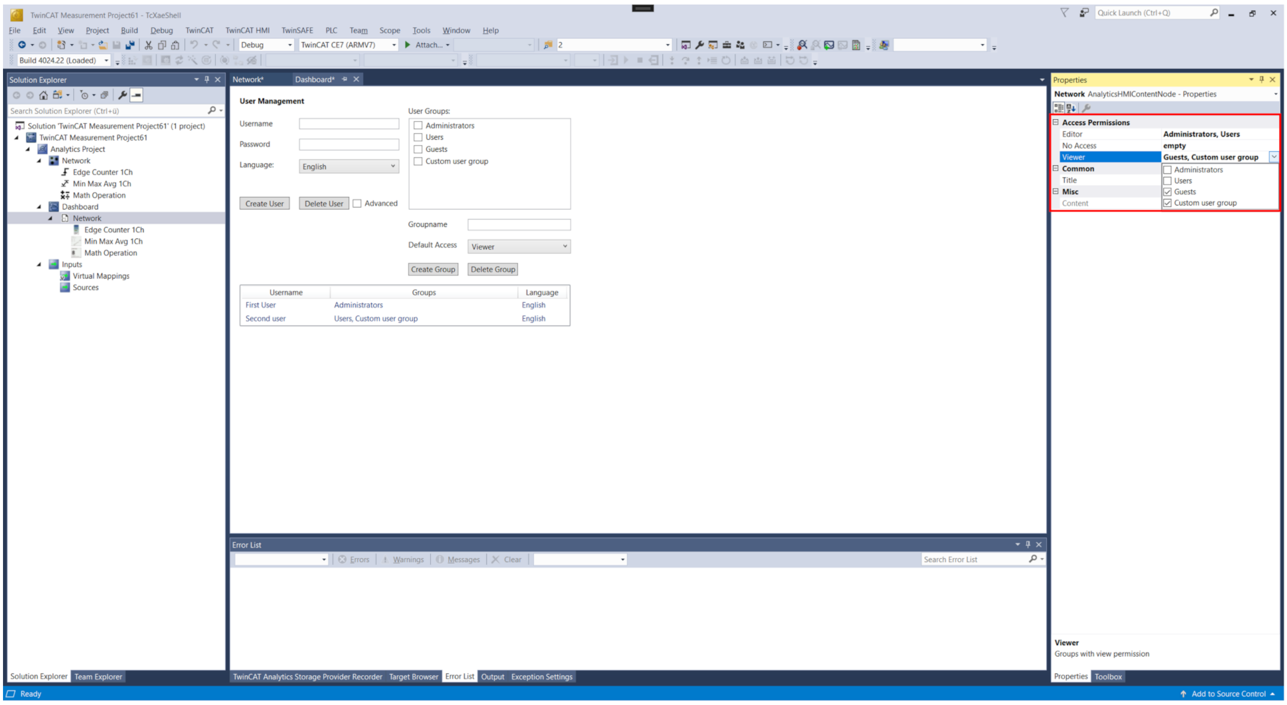
Users (outlined in red) can be created under the Analytics project dashboard node. For a new user, a username and password must be set, as well as a language. In addition, it is necessary to assign one or more user groups to the user, which gives the user the access rights of the respective group(s).

By default, there are the following user groups:

- Administrators
- Users
- Guests

For a more precise classification, additional user groups can be created (outlined in green). For this purpose, a name and the access authorization must be entered.

6.8.3.1.2 Configure access rights



User Management

Username

Password

Language: English

Create User | Delete User | □ Advanced

User Groups:

□ Administrators
□ Users
□ Guests
□ Custom user group

Groupname

Default Access: Viewer

Create Group | Delete Group

Username	Groups	Language
First User	Administrators	English
Second user	Users, Custom user group	English

Network AnalyticsHMIContentNode - Properties

Access Permissions	
Editor	**Administrators, Users**
No Access	**empty**
Viewer	**Guests, Custom user group**
Common	
Title	
Misc	
Content	

□ Administrators
□ Users
☑ Guests
☑ Custom user group

Viewer
Groups with view permission

Access rights can be configured for objects below the dashboard node (contents and controls). Access permissions (outlined in red) can be set for each user group.

The following access rights are available:

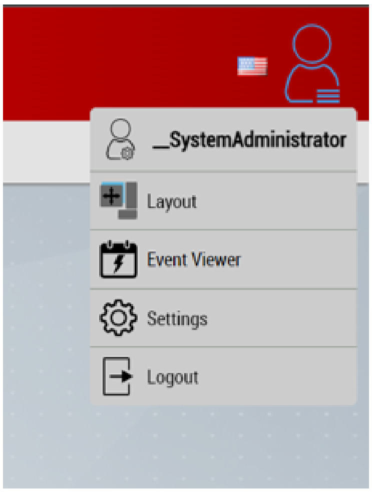
- Editor: Access to the object is permitted, and changes (such as layout adjustments for content objects) can also be made.
- Viewer: Access to the object allowed - but no changes can be made.
- No Access: No access allowed to the object and the objects subordinate to it in the Solution Explorer.

Access rights for content objects can be subsequently adjusted via the web browser.

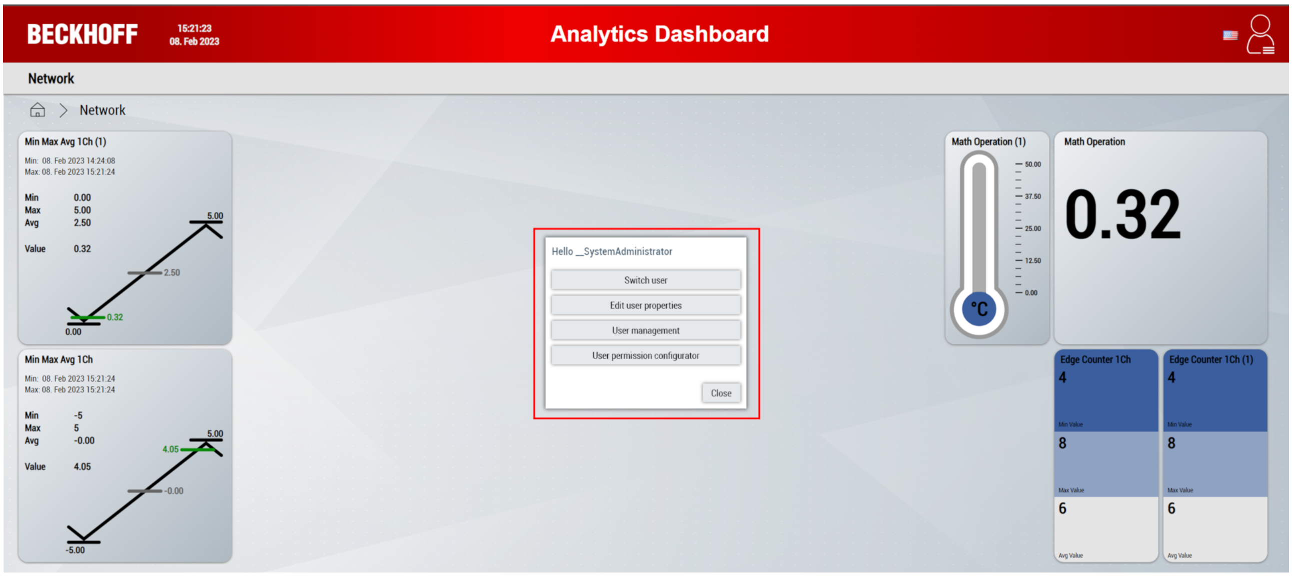
6.8.3.2 Customize users and access rights in the web browser

Note that the menu control and the configuration of users and access rights are only available from Dashboard version 2.0.

Configure user



The menu control can be used to make changes to the users and access rights. To do this, please click on your username (at the top of the menu).



The following options are available via the popup (outlined in red):

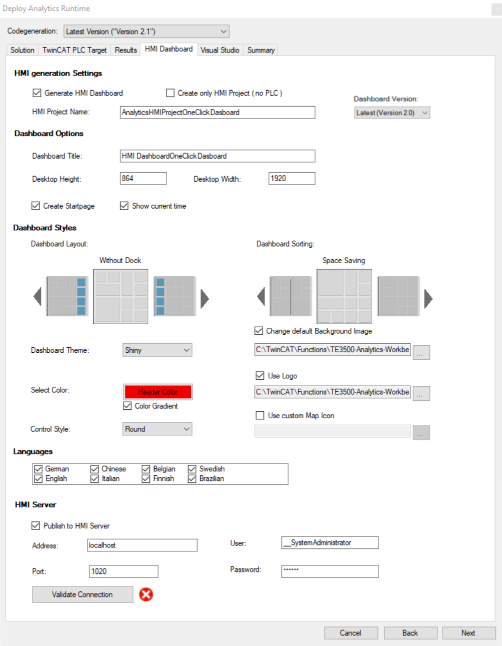
- **Switch user:** Change of the logged in user
- **Edit user properties:** Change the properties of the own user
- **User management:** Create new users, remove existing users and set user properties
- **User permission configurator:** Configuring the access rights of user groups to content objects

Note that when creating new users, individually configured layouts of the creating user are taken over.

6.8.4 Dashboard Configuration

HMI Dashboard tab

The **HMI Dashboard** tab contains all the configurations for the dashboard.



Codegeneration: Latest Version ("Version 2.1")

HMI generation Settings

☑ Generate HMI Dashboard ☐ Create only HMI Project (no PLC)

Dashboard Version: Latest (Version 2.0)

HMI Project Name: AnalyticsHMIProjectOneClickDasboard

Dashboard Options

Dashboard Title: HMI DashboardOneClickDasboard

Desktop Height: 864 Desktop Width: 1920

☑ Create Startpage ☑ Show current time

Dashboard Styles

Dashboard Layout:

Dashboard Sorting:

☑ Change default Background Image

C:\TwinCAT\Functions\TE3500-Analytics-Workbe

Dashboard Theme: Shiny

☑ Use Logo

C:\TwinCAT\Functions\TE3500-Analytics-Workbe

Select Color: Header Color

☑ Color Gradient

☐ Use custom Map Icon

Control Style: Round

Languages

☑ German ☑ Chinese ☑ Belgian ☑ Swedish
☑ English ☑ Italian ☑ Finnish ☑ Brazilian

HMI Server

☑ Publish to HMI Server

Address: localhost User: __SystemAdministrator

Port: 1020 Password: ******

Configuration

HMI generation Settings

Generate HMI Dashboard	Enables the generation of the HMI Dashboard, if enabled. This automatically activates/deactivates the checkboxes **Create Bootproject** and **Activate PLC Runtime** on the **TwinCAT PLC Target** tab.
Create only HMI Project (No PLC)	Creates only one HMI project and not a PLC project, if enabled.
HMI Project Name	Name of the TwinCAT HMI project.
Dashboard version	Selection of the dashboard version.

Dashboard Options

Dashboard Title	Title of the HMI Dashboard, which is displayed in the dashboard header.
Desktop Width	Width of the target screen in pixels.
Desktop Height	Height of the target screen in pixels.
Create Start-Page	Creates a start page for the dashboard that displays a map with all machine locations. The location data is adopted from the machine management data.
Show current time	Creates a clock in the dashboard header that shows the current local time.

Dashboard Styles

Dashboard Layout	Defines the layout of the dashboard. Dock requires the "Dock" property of a module to be TRUE. Dock Left: fixed left column Dock Right: fixed right column Without Dock: no fixed column
Dashboard Sorting	Defines the sorting of the dashboard. Space Saving: arranges the controls without free space. Control Type: arranges the controls by Control type. Control Size: arranges the controls by size from large to small. Filled: arranges the controls so that the entire screen is filled. Network Groups: groups the controls by network and summarizes them on a screen. Grouping begins at the defined level (0 = All, 1 = First Level...)
Control Distance	Determines the minimum distance between controls in the grid.
Dashboard Theme	Defines the topic of the HMI dashboard, affects controls and backgrounds: Light: bright skin, especially for day mode. Shiny: similar to the bright skin, color gradient in the controls. Dark: dark skin, especially for night mode.
Header Color	Specifies the color of the dashboard header. This contains a color gradient if Color Gradient is enabled.
Control Style	Defines the style of the controls: Round: The controls have rounded corners. Flat: The controls have angular corners.
Change default Background Image	If enabled, a customer-specific background image can be set for the HMI Dashboard. If nothing is defined, the default image is used.
Use Logo	If enabled, a logo is added to the dashboard header. A customer-specific image can be defined for the logo.

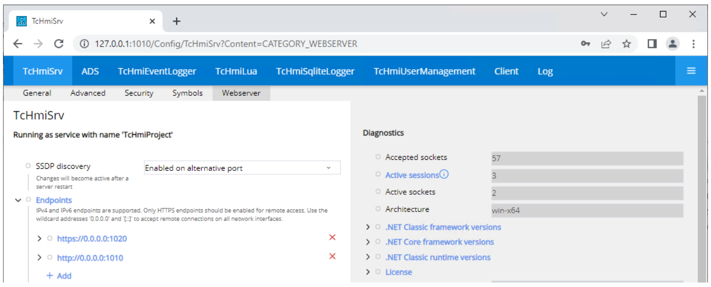
Languages

Languages	Selection of languages available in the HMI Dashboard.
Select Default Language	Default language for the dashboard.

HMI Server

Publish to TwinCAT HMI Server	Publishes the dashboard to a TwinCAT HMI server, if enabled.
Address	Enter the IP address or host name of the TwinCAT HMI server.
Port	Enter the port of the TwinCAT HMI Server (by default 1010 without encryption and 1020 with encryption).
User	Enter the user name.
Password	Enter the password. (The administrator password must be set once on the HMI server via the configuration page).
Validate Connection	Press the button to validate your server connection.

In order to publish to a remote HMI server, the HMI port and IP of the development computer must be shared in the HMI server. Also an inbound rule with the HMI ports must be set up in the Windows firewall from the remote PC.



127.0.0.1:1010/Config/TcHmiSrv?Content=CATEGORY_WEBSERVER

TcHmiSrv

Running as service with name 'TcHmiProject'

SSDP discovery: Enabled on alternative port

Changes will become active after a server restart

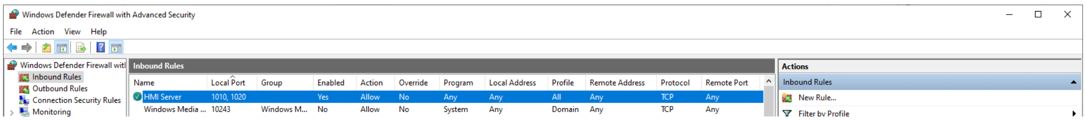
Endpoints

IPv4 and IPv6 endpoints are supported. Only HTTPS endpoints should be enabled for remote access. Use the wildcard addresses '0.0.0.0' and '[::]' to accept remote connections on all network interfaces.

- https://0.0.0.0:1020
- http://0.0.0.0:1010

Diagnostics

Accepted sockets	57
Active sessions	3
Active sockets	2
Architecture	win-x64



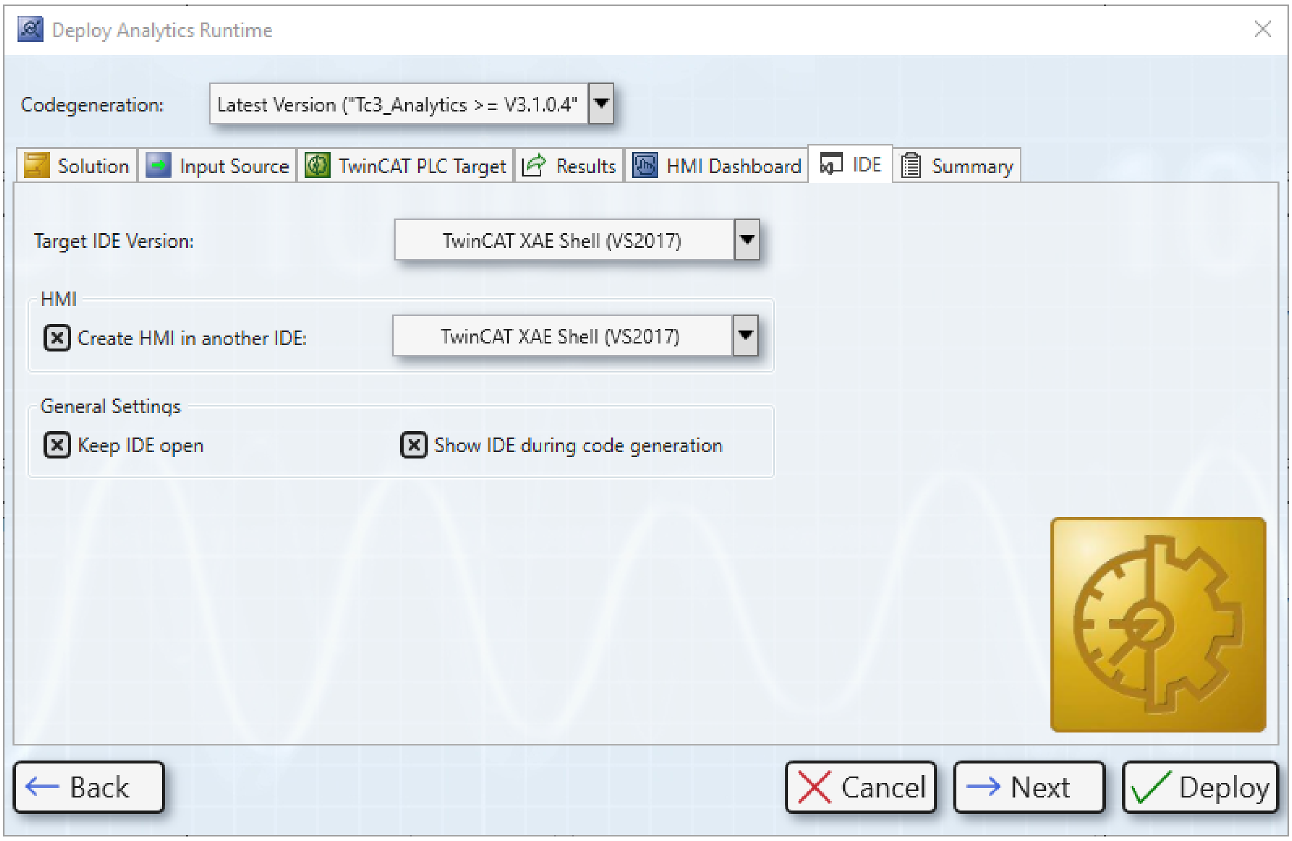
Windows Defender Firewall with Advanced Security

Inbound Rules

Name	Local Port	Group	Enabled	Action	Override	Program	Local Address	Profile	Remote Address	Protocol	Remote Port
HMI Server	1010, 1020		Yes	Allow	No	Any	Any	All	Any	TCP	Any
Windows Media ...	10243	Windows M...	No	Allow	No	System	Any	Domain	Any	TCP	Any

IDE tab

On the **IDE** tab, you can select which IDE (Visual Studio® or TwinCAT XAE Shell) is to be used for generating the PLC and HMI. In addition, it is possible to generate HMI and PLC in two different solutions.



Deploy Analytics Runtime

Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4"

Target IDE Version: TwinCAT XAE Shell (VS2017)

HMI

☒ Create HMI in another IDE: TwinCAT XAE Shell (VS2017)

General Settings

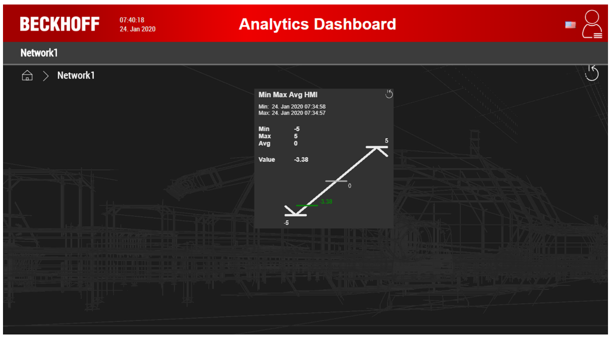
☒ Keep IDE open ☒ Show IDE during code generation

Keep IDE open	Keeps the IDE open after generation, if enabled.
Target IDE Version	The IDE target version for PLC and HMI.
Create HMI in another IDE	If enabled, the HMI is generated into a second solution. A different IDE can be selected.
Requirement: TwinCAT or TwinCAT HMI must be installed in the selected IDE.	

Click **Next** to display the **Summary** tab, then click **Deploy** to start generating the dashboard.

Impressions

Topics



BECKHOFF 07:40:18 24. Jan 2020 Analytics Dashboard

Network1

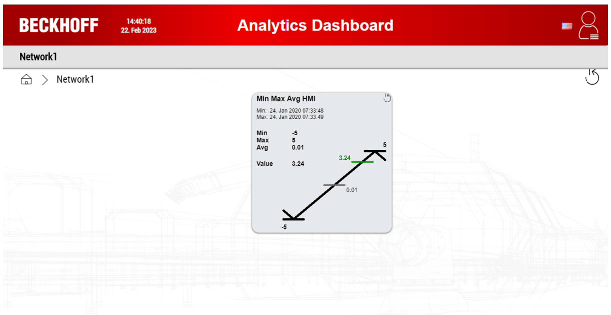
Network1

Min Max Avg HMI

Min: 24. Jan 2020 07:34:58
Max: 24. Jan 2020 07:34:57

Min -5
Max 5
Avg 0

Value -3.38



BECKHOFF 14:40:18 22. Feb 2023 Analytics Dashboard

Network1

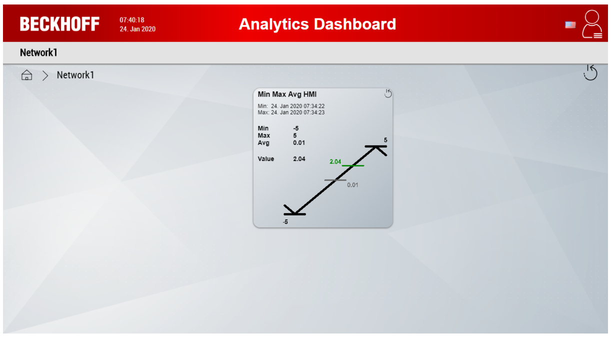
Network1

Min Max Avg HMI

Min: 24. Jan 2020 07:33:48
Max: 24. Jan 2020 07:33:49

Min -5
Max 5
Avg 0.01

Value 3.24



BECKHOFF 07:40:18 24. Jan 2020 Analytics Dashboard

Network1

Network1

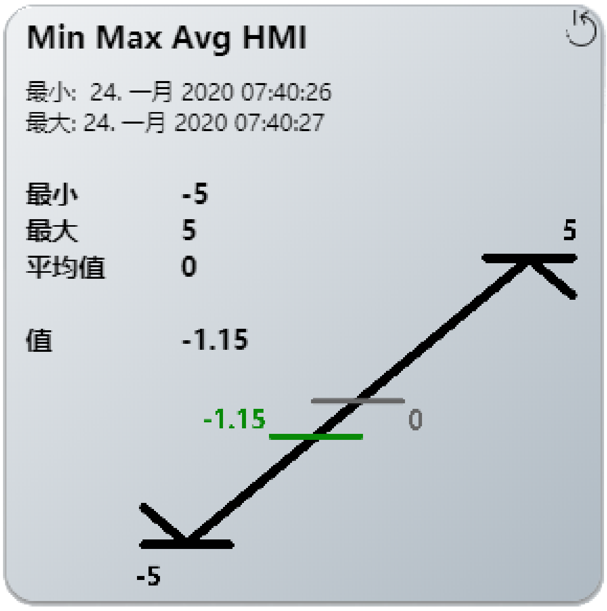
Min Max Avg HMI

Min: 24. Jan 2020 07:34:22
Max: 24. Jan 2020 07:34:23

Min -5
Max 5
Avg 0.01

Value 2.04

Styles

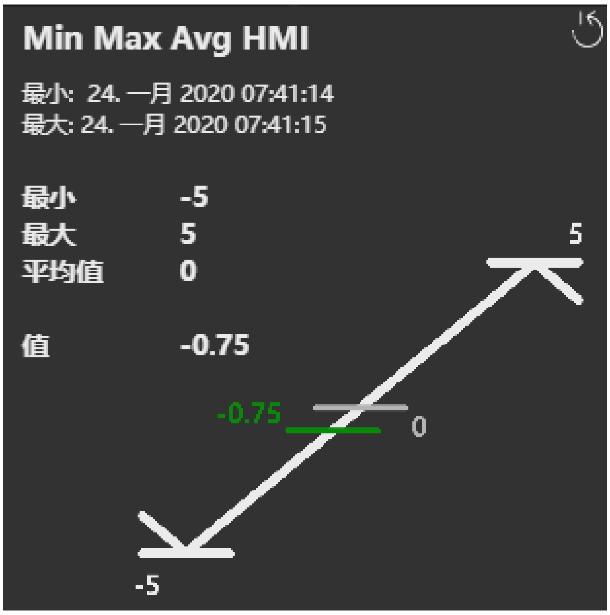


Min Max Avg HMI

最小: 24. 一月 2020 07:40:26
最大: 24. 一月 2020 07:40:27

最小 -5
最大 5
平均值 0

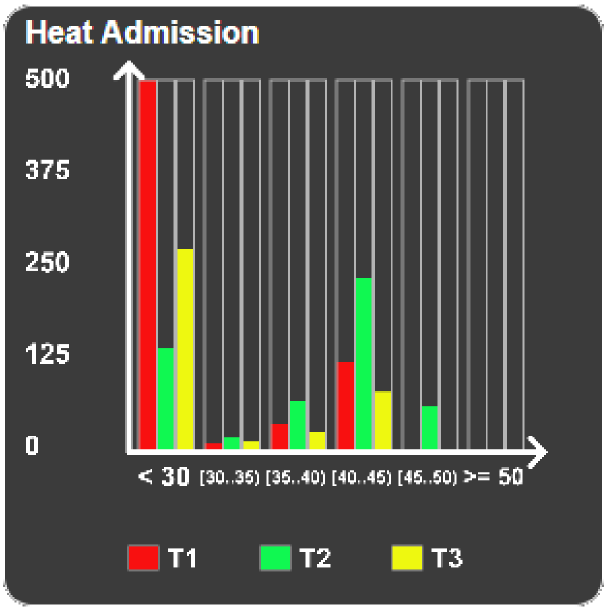
值 -1.15



Min Max Avg HMI

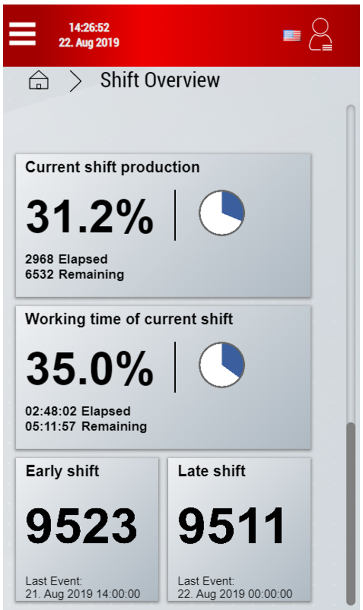
最小: 24. 一月 2020 07:41:14
最大: 24. 一月 2020 07:41:15

最小	-5
最大	5
平均值	0
值	-0.75



Heat Admission

Views for mobile devices



14:26:52
22. Aug 2019

Shift Overview

Current shift production

31.2%

2968 Elapsed
6532 Remaining

Working time of current shift

35.0%

02:48:02 Elapsed
05:11:57 Remaining

Early shift

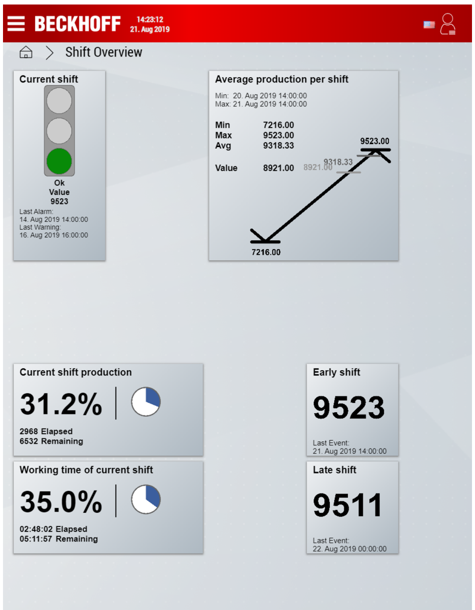
9523

Last Event:
21. Aug 2019 14:00:00

Late shift

9511

Last Event:
22. Aug 2019 00:00:00


BECKHOFF 14:23:12 21. Aug 2019

Shift Overview

Current shift

Ok
Value
9523
Last Alarm:
14. Aug 2019 14:00:00
Last Warning:
16. Aug 2019 16:00:00

Average production per shift

Min: 20. Aug 2019 14:00:00
Max: 21. Aug 2019 14:00:00

Min 7216.00
Max 9523.00
Avg 9318.33

Value 8921.00

Current shift production

31.2%

2968 Elapsed
6532 Remaining

Early shift

9523

Last Event:
21. Aug 2019 14:00:00

Working time of current shift

35.0%

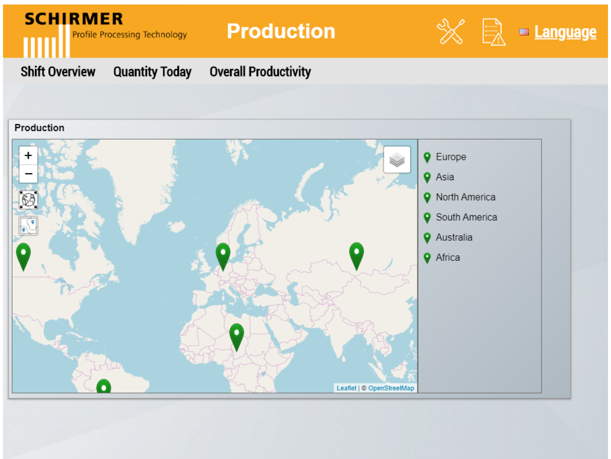
02:48:02 Elapsed
05:11:57 Remaining

Late shift

9511

Last Event:
22. Aug 2019 00:00:00

Map on customer-specific dashboard


SCHIRMER Profile Processing Technology

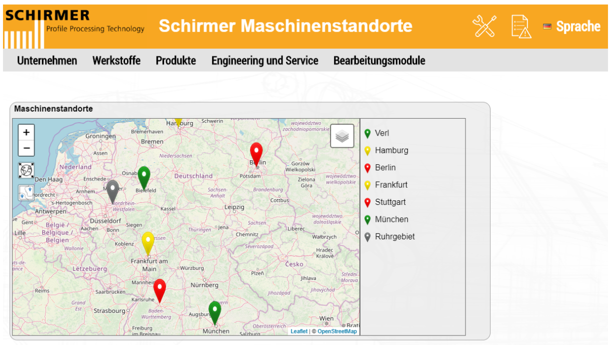
Production

Language

Shift Overview Quantity Today Overall Productivity

Production

- Europe
- Asia
- North America
- South America
- Australia
- Africa


SCHIRMER Profile Processing Technology

Schirmer Maschinenstandorte

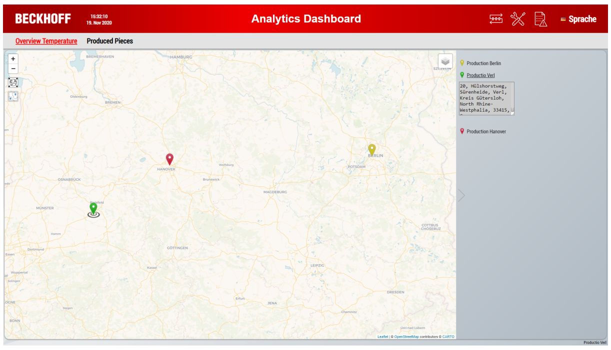
Sprache

Unternehmen Werkstoffe Produkte Engineering und Service Bearbeitungsmodule

Maschinenstandorte

- Verl
- Hamburg
- Berlin
- Frankfurt
- Stuttgart
- München
- Ruhrgebiet

Historical data and machine switching


BECKHOFF 15:32:10 19. Nov 2020

Analytics Dashboard

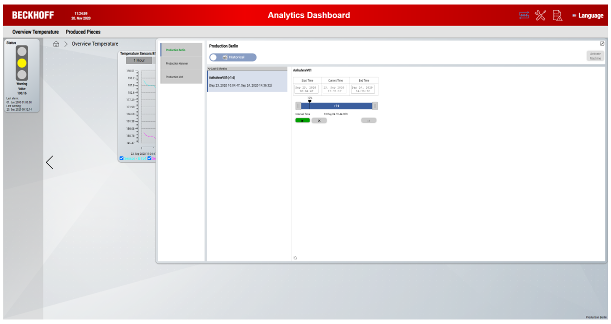
Sprache

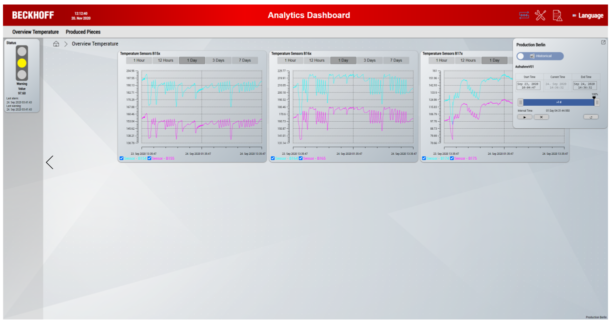
Overview Temperature Produced Pieces

- Production Berlin
- Productio Verl

20, Hülshorstweg, Sürenheide, Verl, Kreis Gütersloh, North Rhine-Westphalia, 33415,

- Production Hanover

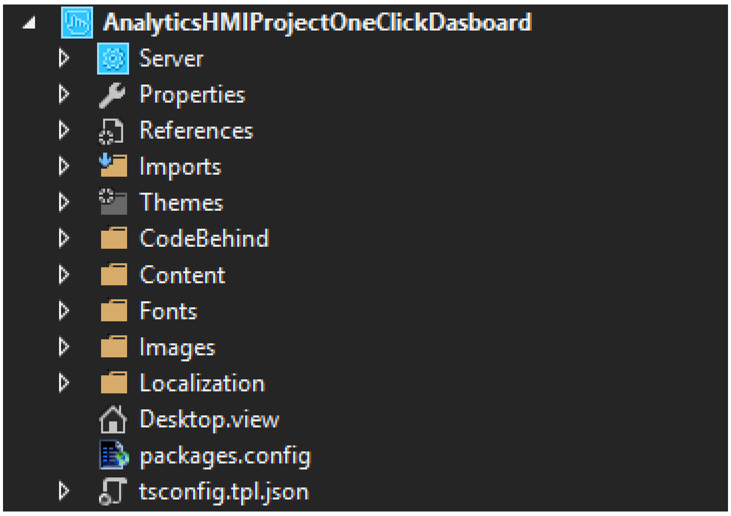




6.8.5 Modifying a generated dashboard in HMI engineering

TwinCAT 3 HMI project

The result of dashboard generation is a complete TwinCAT 3 HMI project. Therefore, all options offered by TwinCAT 3 HMI Engineering can be used and included.



- AnalyticsHMIProjectOneClickDasboard
  - Server
  - Properties
  - References
  - Imports
  - Themes
  - CodeBehind
  - Content
  - Fonts
  - Images
  - Localization
  - Desktop.view
  - packages.config
  - tsconfig.tpl.json

Fig. 5:

Note that many dashboard customizations can also be made in the web browser at runtime.

6.8.6 Configuration of the dashboard at runtime in the web browser

At runtime, different configurations can be made in the web browser:

- Create individual user-specific layouts
- Manage users (see Configuring user management and access rights)
- Customize access rights to contents (see Configuring user management and access rights)
- Changing control properties in the browser

- Customize parameters
- Changing global dashboard options

Note that not all configuration options are available with Dashboard version 1.0.

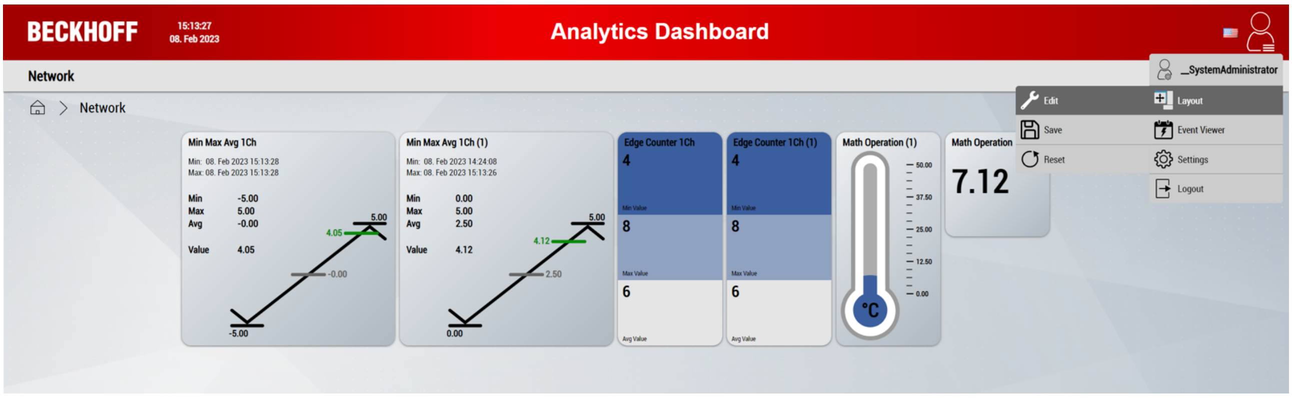
6.8.6.1 Customizing layouts with the Interactive function

With the interactive function, controls can be arranged within a content in a user-specific way. The customized layout is stored centrally in the HMI server extension "TcHmiAnalytics".

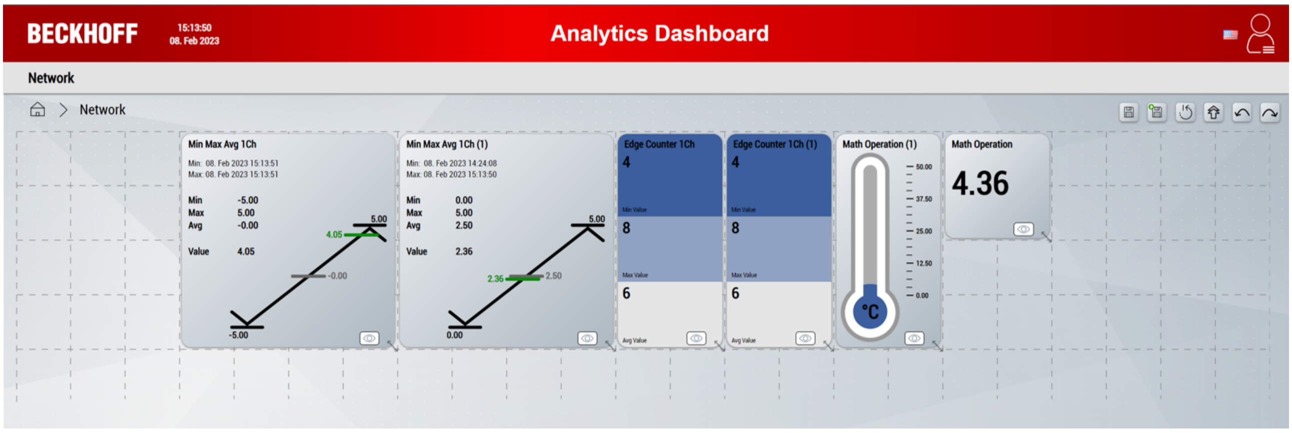
Note that layout editing is available only from Dashboard version 2.0. In addition, the layout can only be edited if the user has the necessary access rights.

Edit mode

Editing a layout is done in a special editing mode. The editing mode can be called via the menu control. Alternatively, the edit mode can also be started with the key combination "CTRL" + "E".



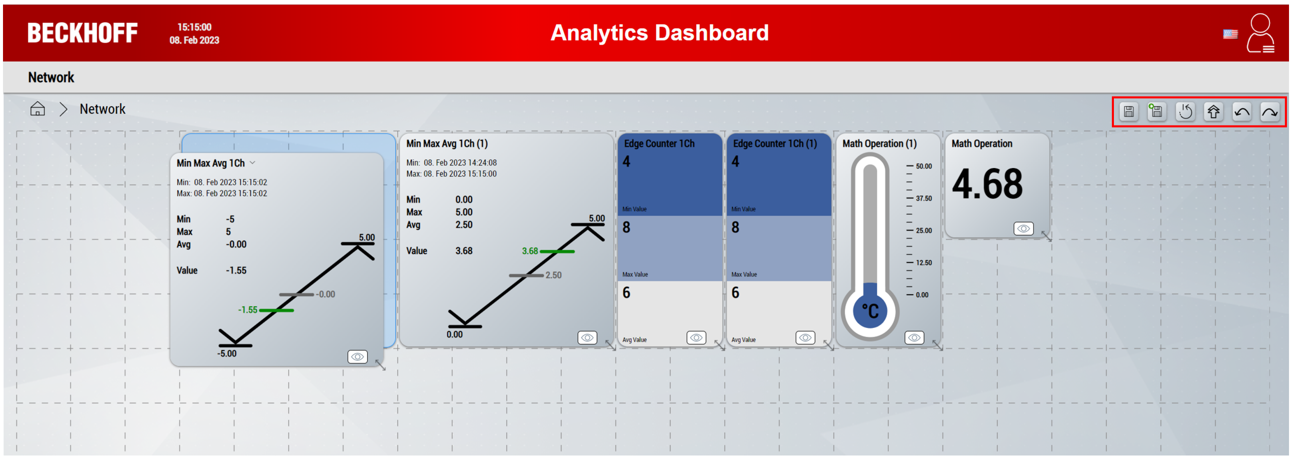
In edit mode, a grid is visible in the background, on which the controls can be arranged as desired.



Edit layouts

The following options are available:

- **Positioning:** For positioning, a control must be clicked or pressed. Thus, it is released from its anchorage and can be moved. The light blue frame indicates the new position within the layout. When the control is released, it anchors itself in its new position.
- **Scale:** Using the arrow in the lower right corner of a control, its size can be changed.
- **Hide controls:** By using the visibility button (bottom right inside a control) it can be shown or hidden. Hidden controls are displayed semi-transparently in edit mode.



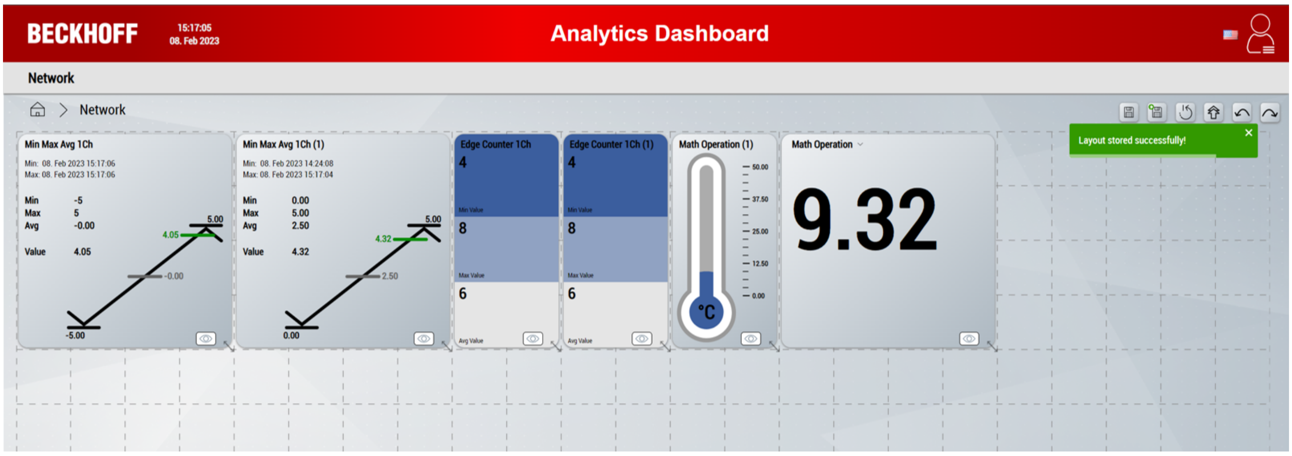
Editing menu

In edit mode, different actions can be performed via an additional menu (outlined in red):

- Save the layout (alternatively key combination "CTRL" + "S")
- Saving the layout and exiting the editing mode (alternatively key combination "CTRL" + "Q")
- Reset the changes made (alternatively key combination "CTRL" + "R")
- Move controls upwards so that free lines are removed (alternatively key combination "CTRL" + "U")
- Undo last action (alternatively key combination "CTRL" + "Z")
- Repeat last action (alternatively key combination "CTRL" + "Y")

Save layouts

Customized layouts can be saved via the edit menu or via the menu control. A popup shows the result of the save operation.

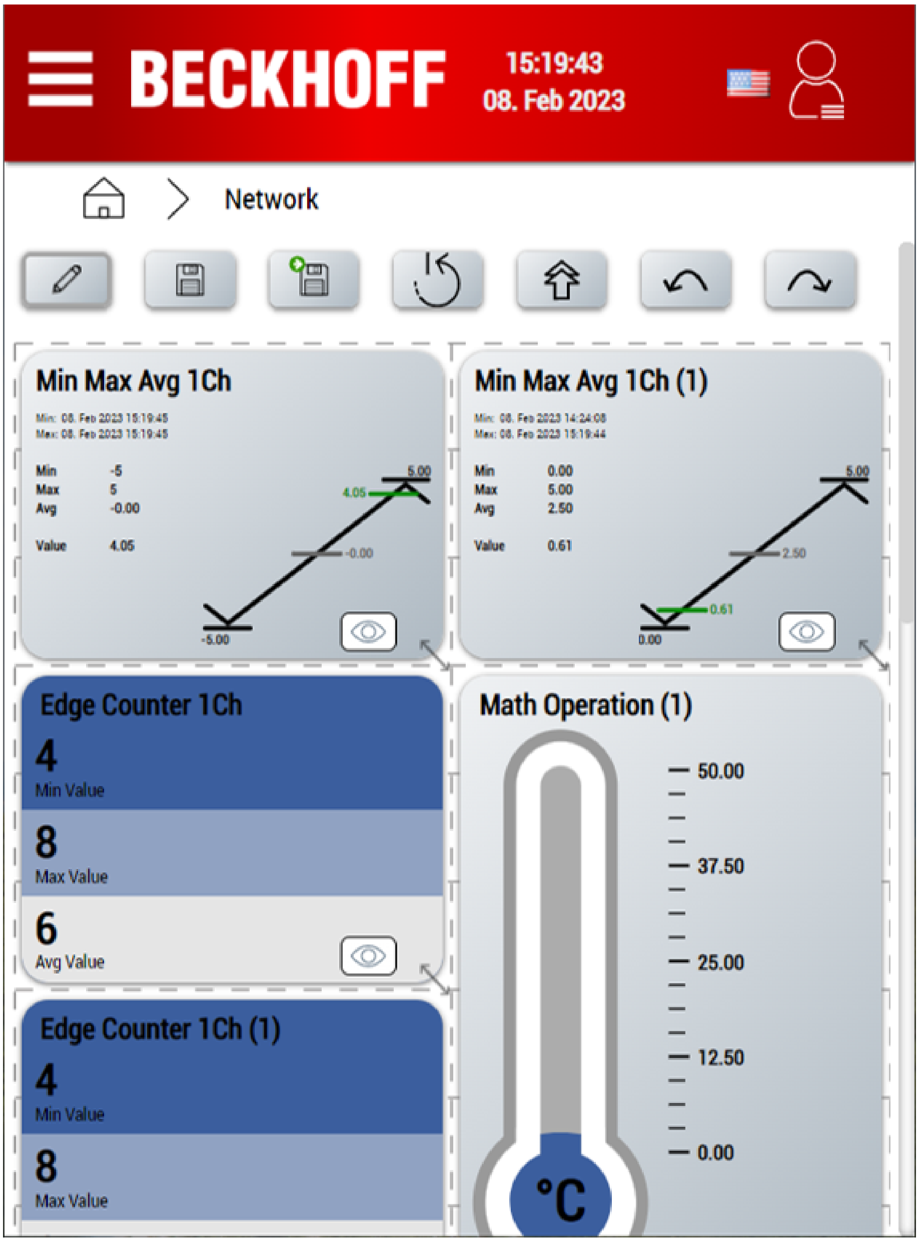


Layout stored successfully!

If an error occurs during saving, check in the server configuration whether the "TcHmiAnalytics" extension is activated.

Mobile layout

For mobile devices, a user-specific mobile layout can additionally be generated. This is independent from the normal layout and has only 2 columns. Editing and saving is done in the same way.



6.8.6.2 Changing control properties in the browser

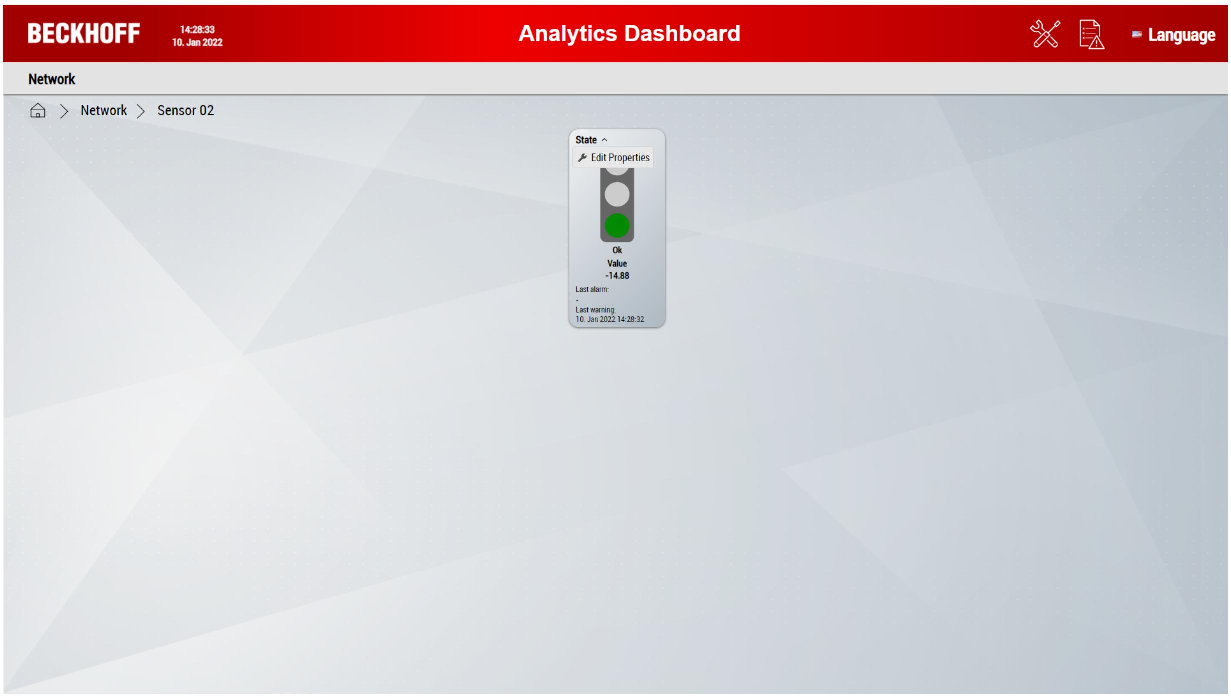
Recommendation: make changes in the Analytics Workbench project

Usually, the changes made here can already be made in the Analytics Workbench before you generate the dashboard (see Mapping Wizard or also Dashboard node). This has the advantage that the changes are saved when the dashboard is generated again.

Each Analytics Control has selected properties that can be changed dynamically in the browser. This option is available from Analytics version 3.4.3145.0 or with Control Package version 1.1.31. You can use it to easily update the NuGet package for older HMI projects.

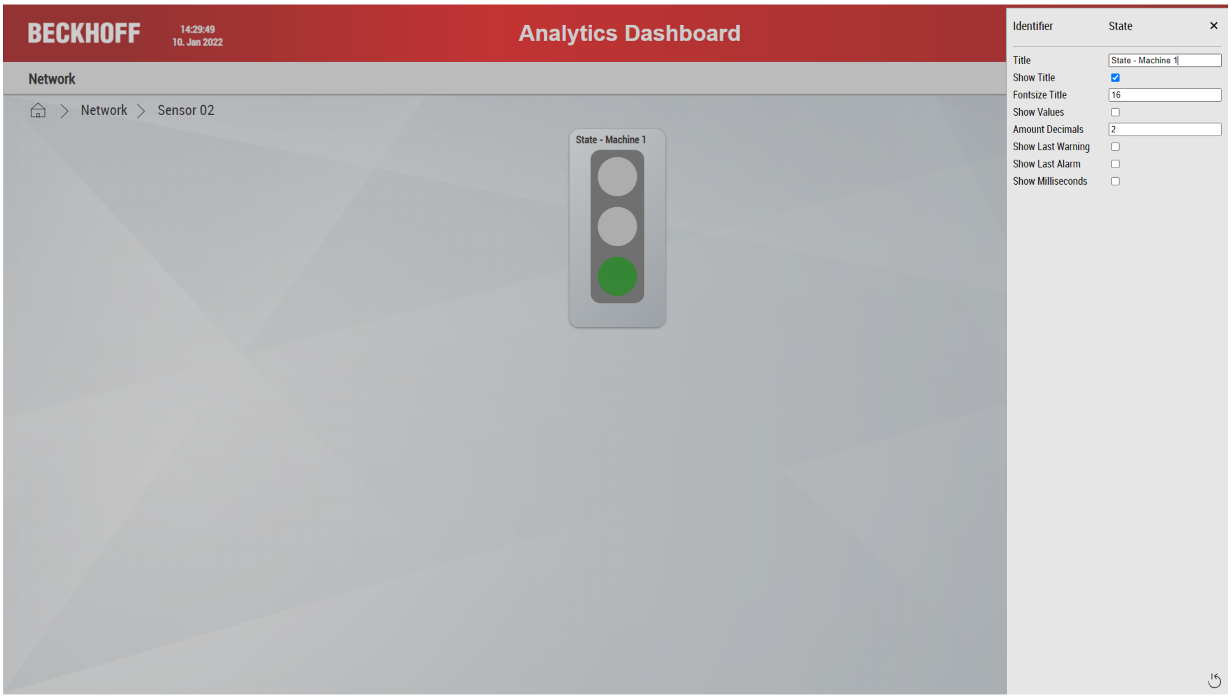
1. Open a generated HMI project and click on the title of a control where you want to change a property.

2. There a menu opens where you can click on **Edit Properties**.



⇨ Each control has different properties that can be customized. Here, for demonstration purposes, the properties of the *Traffic Light* control are changed

3. Change the title and uncheck the bottom three selection boxes to display only the traffic light with the changed title.



Identifier State ×

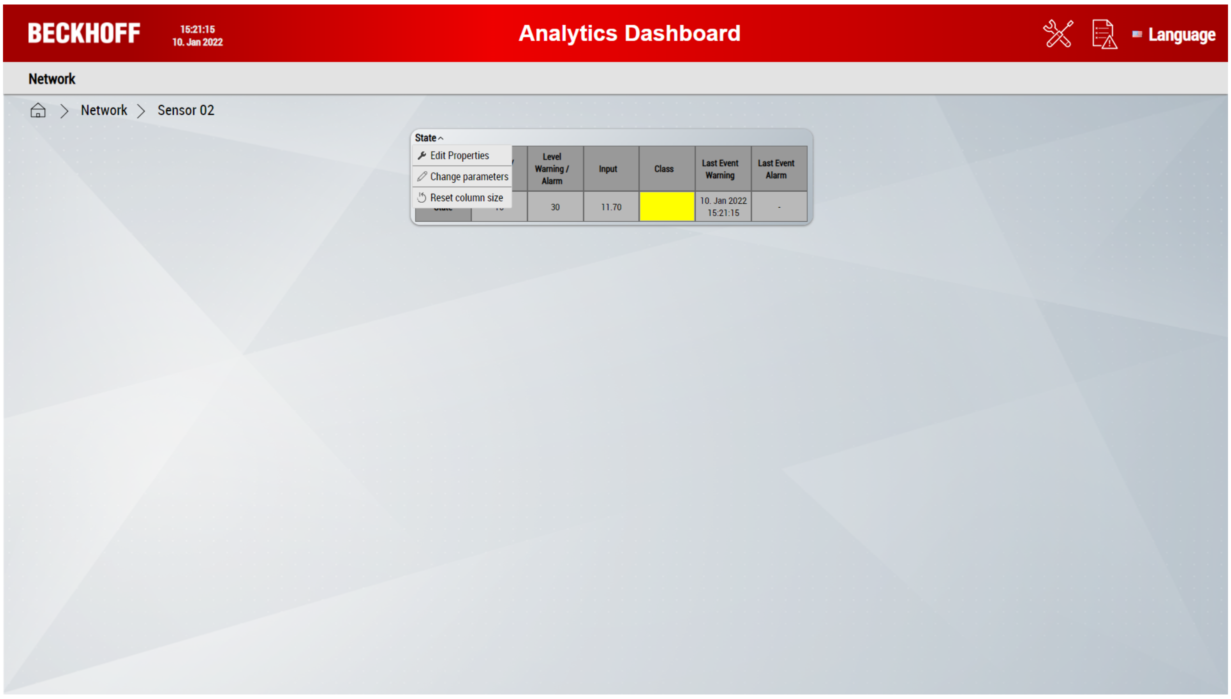
Title: State - Machine 1
Show Title: ☑
Fontsize Title: 16
Show Values: ☐
Amount Decimals: 2
Show Last Warning: ☐
Show Last Alarm: ☐
Show Milliseconds: ☐

⇨ The changes made are saved client-side (on the device where you open the dashboard) and are only displayed there. In the future, storage will be performed via the HMI server, which means that changes will be made globally on each device.

6.8.6.3 Changing parameters in the browser

Numerical parameters such as limit values can be changed dynamically in the dashboard. This is possible with the Data Table Control. The values are persistently written back to the PLC there and are thus also available after a restart.

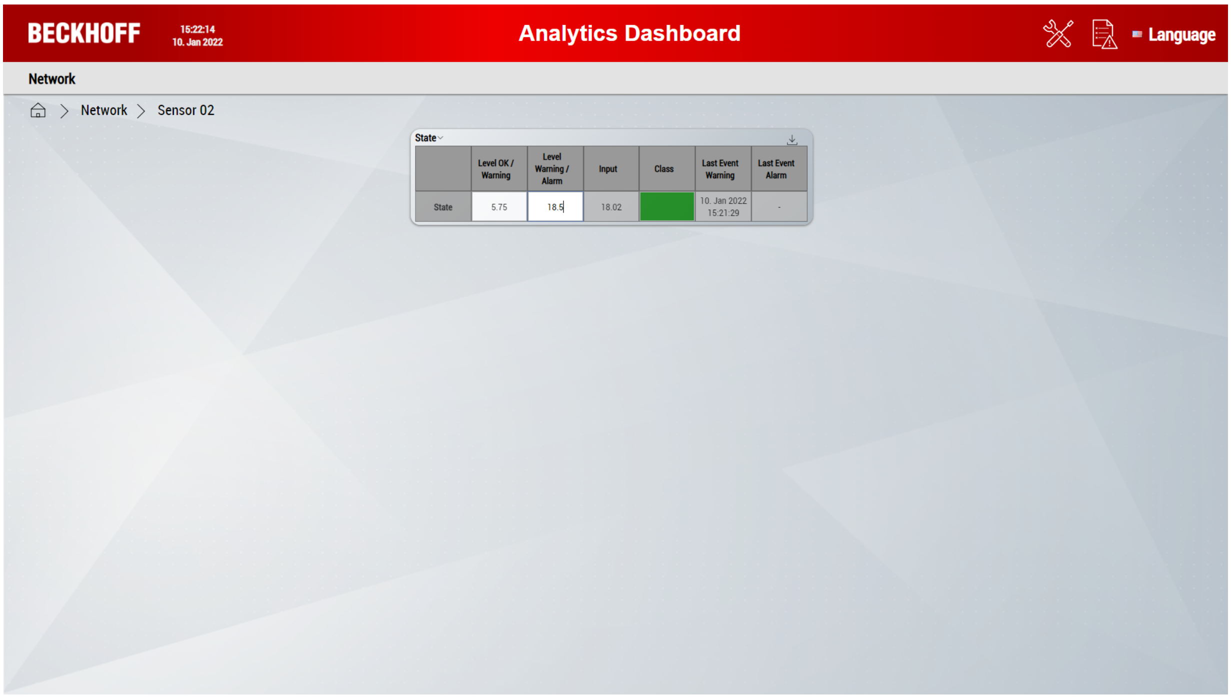
1. Open a generated HMI project and click on the title of the table control where you want to change a parameter.
2. There a menu opens where you click on **Change parameters**. A module without parameters does not have this menu item.



		Level Warning / Alarm	Input	Class	Last Event Warning	Last Event Alarm
State	10	30	11.70	🟨	10. Jan 2022 15:21:15	-

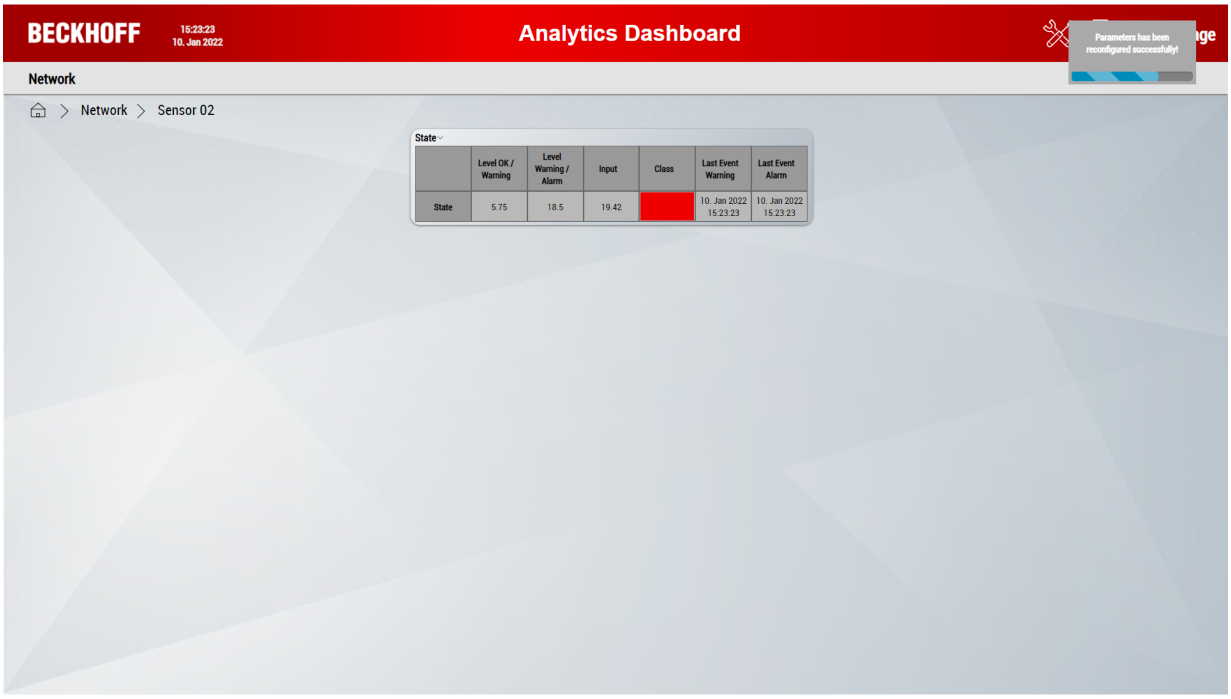
⇨ The changeable fields change to an input field with a white background.

3. Change the values there and then click the save icon in the top right corner.



	Level OK / Warning	Level Warning / Alarm	Input	Class	Last Event Warning	Last Event Alarm
State	5.75	18.5	18.02	🟩	10. Jan 2022 15:21:29	-

⇨ After saving the parameters, you will see a message in the upper right area of the dashboard. There you will be told if the reconfiguration worked.



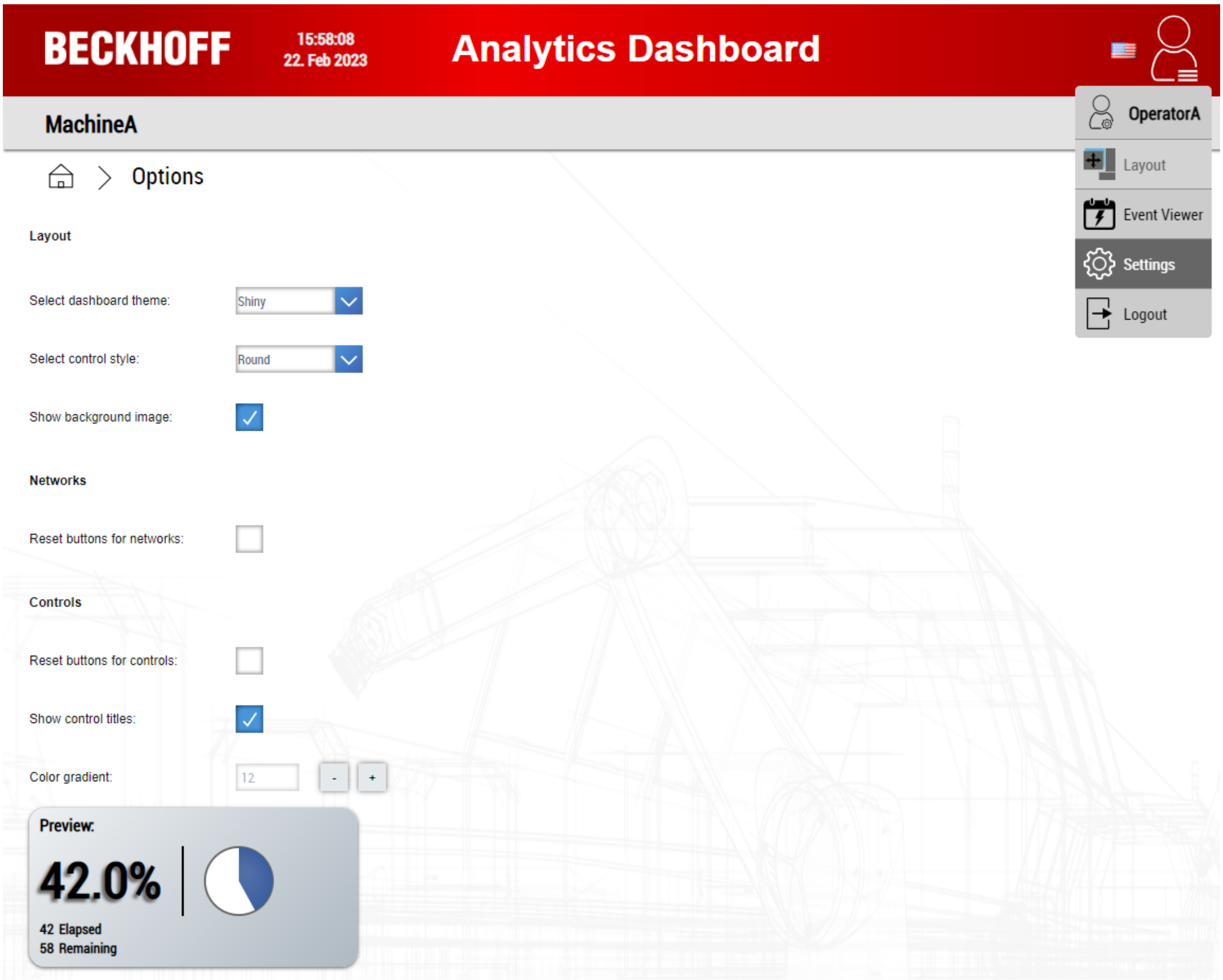
Parameters has been reconfigured successfully!

State

	Level OK / Warning	Level Warning / Alarm	Input	Class	Last Event Warning	Last Event Alarm
State	5.75	18.5	19.42	🟥	10. Jan 2022 15:23:23	10. Jan 2022 15:23:23

6.8.6.4 Customizing global dashboard options

On the Options page, general settings for the dashboard can be edited.



Options

Layout

Select dashboard theme: Shiny

Select control style: Round

Show background image: ☑

Networks

Reset buttons for networks: □

Controls

Reset buttons for controls: □

Show control titles: ☑

Color gradient: 12 - +

Layout	
Select dashboard theme	Change the dashboard theme between shiny, dark and light.
Select control style	Change the control style between flat and round.
Show background image	Displays the standard or customer-specific background image.
Networks	
Reset buttons for networks	Enables reset buttons for entire networks. Reset all controls within the selected network.
Controls	
Reset buttons for controls	Enables reset buttons for controls. Reset a single control.
Show control titles	Enables control titles for all controls.

6.8.7 Creating charts in the web browser

With the new Interactive Chart function, you can add charts to your dashboards as required. Charts can be added and configured directly in the web browser. Engineering is not required for this. Data that has been historized with the Analytics Storage Provider can be used as a data source for the charts. During configuration, you can select the variables to display, the time ranges and the chart type.

Analytics Storage Provider Update required

A **TwinCAT Analytics Storage Provider >= version 1.0.12.3** is required so that the data can be loaded correctly.

The following NuGet packages are required to use the Interactive Chart function:

Beckhoff.TwinCAT.Analytics.Controls >= 1.35.0

Beckhoff.TwinCAT.Analytics.ControlsExtension >= 1.35.0

The Interactive Chart function can be integrated into already generated HMI projects by updating the NuGet packages.

6.8.7.1 Managing the data sources

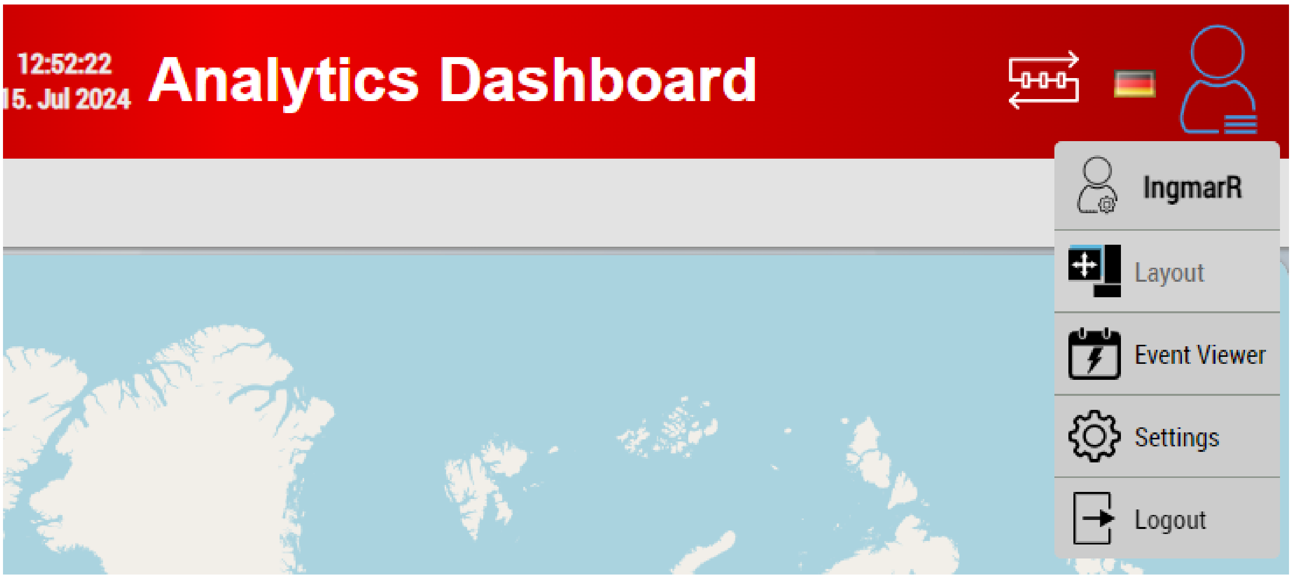
Data that has been historized with the Analytics Storage Provider can be used as a data source for the charts. The data sources can be managed directly via the dashboard.

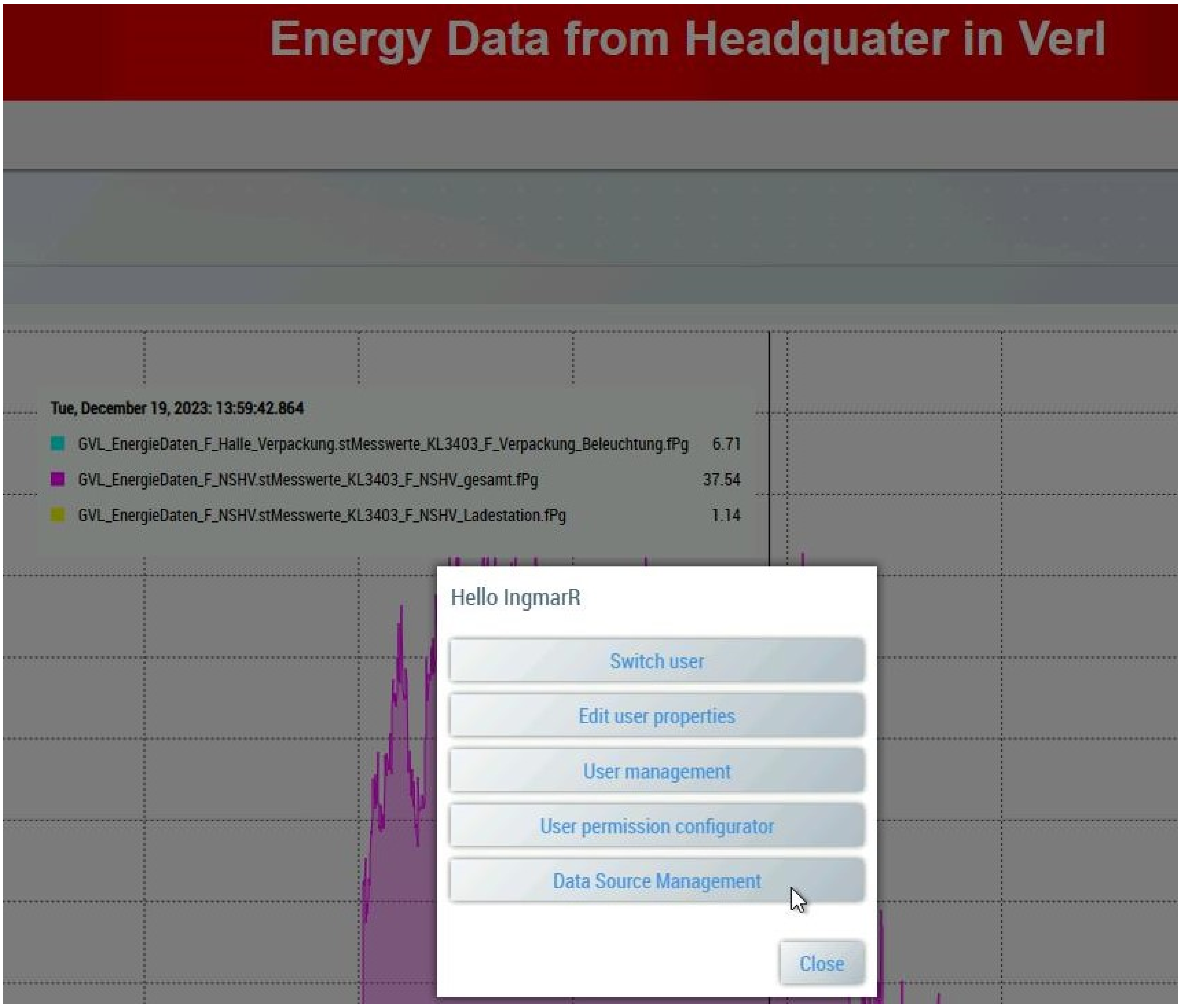
Automatic addition of historical sources during code generation

From TwinCAT Analytics Workbench version >= 49.0, all historical data sources from the project are automatically transferred to the dashboard during code generation. It does not matter whether these are actively used in the project or have merely been added as sources. Please note that live data sources cannot be used.

Open of the Data Source Management

Data sources can only be managed by users in the "__SystemAdministrators" or "Administrators" group. You can open the Data Source Management via the menu icon in the top right-hand corner: **Menu - <username> - Data Source Management**.



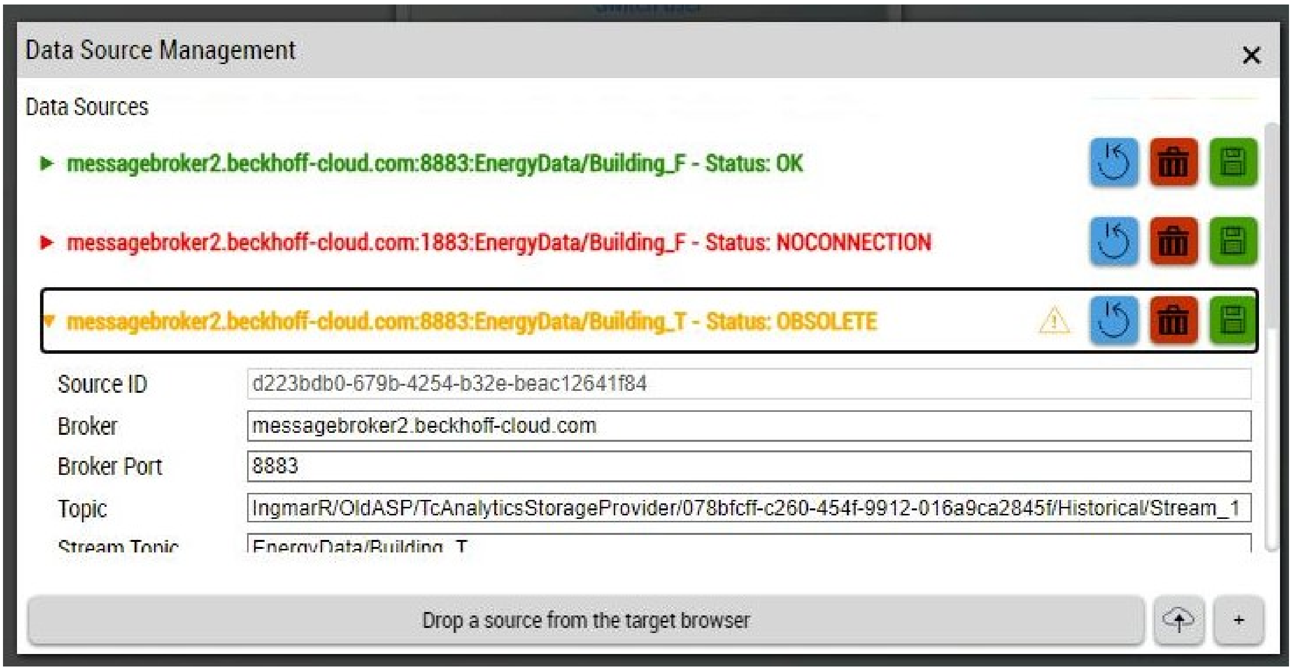


Energy Data from Headquater in Verl

Tue, December 19, 2023: 13:59:42.864

🟦	GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fPg	6.71
🟪	GVL_EnergieDaten_F_NSHV.stMesswerte_KL3403_F_NSHV_gesamt.fPg	37.54
🟨	GVL_EnergieDaten_F_NSHV.stMesswerte_KL3403_F_NSHV_Ladestation.fPg	1.14

Hello IngmarR



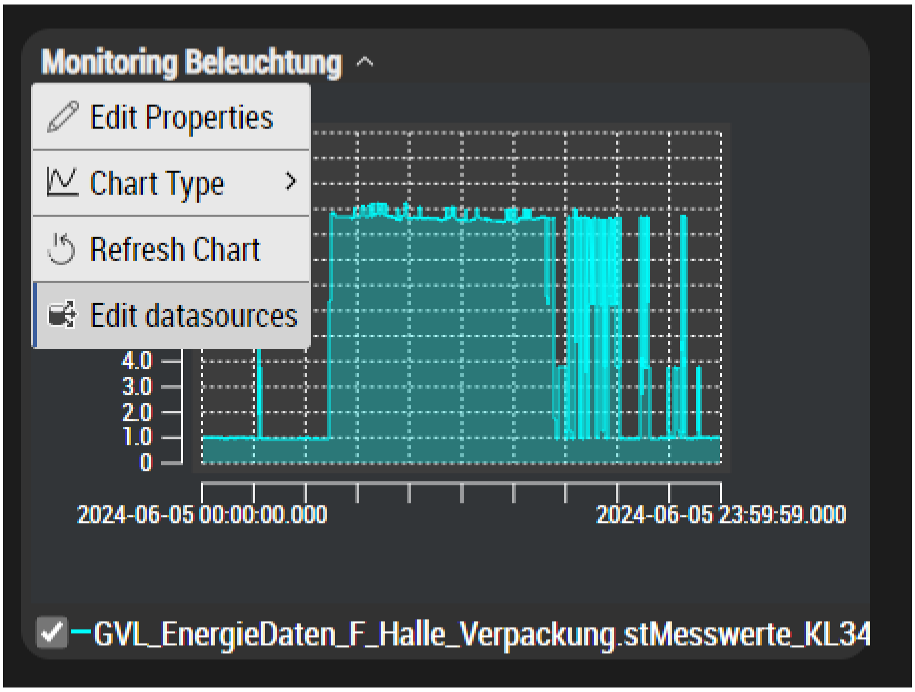
Data Source Management

Data Sources

- ▶ messagebroker2.beckhoff-cloud.com:8883:EnergyData/Building_F - Status: OK
- ▶ messagebroker2.beckhoff-cloud.com:1883:EnergyData/Building_F - Status: NOCONNECTION
- ▼ messagebroker2.beckhoff-cloud.com:8883:EnergyData/Building_T - Status: OBSOLETE ⚠

Source ID	d223bdb0-679b-4254-b32e-beac12641f84
Broker	messagebroker2.beckhoff-cloud.com
Broker Port	8883
Topic	IngmarR/OldASP/TcAnalyticsStorageProvider/078bfcff-c260-454f-9912-016a9ca2845f/Historical/Stream_1
Stream Topic	EnergyData/Building_T

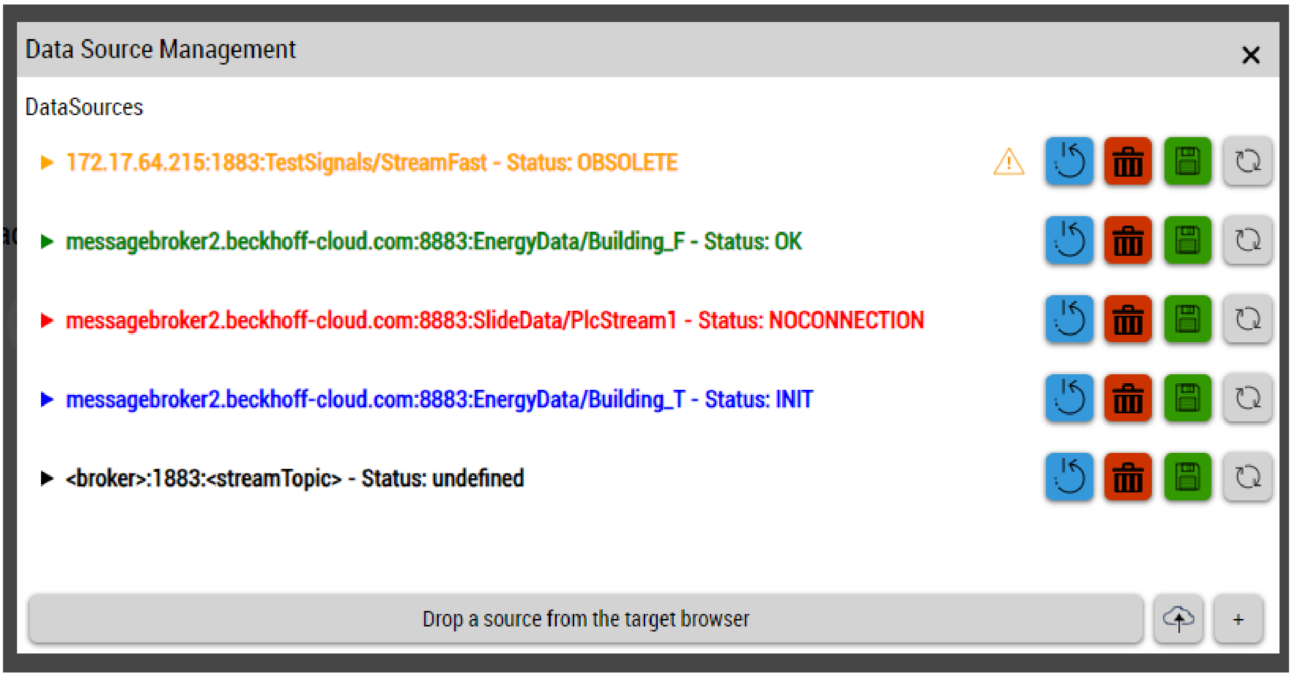
Alternatively, you can also open Data Source Management via an existing **Interactive Chart**.



Overview of data sources

All existing data sources are displayed in the Data Source Management. The status of each data source is also displayed. This has the following meanings:

- OK: The data source has been configured correctly.
- NOCONNECTION: The data source was configured incorrectly. In this case, the connection parameters should be checked and corrected.
- OBSOLETE: The Storage Provider of the data is outdated and should be updated to at least version >= 1.0.12.3. Otherwise, data cannot be queried correctly.
- INIT: A new configuration has been sent to the server extension. This attempts to connect to the data source.
- UNDEFINED: The configuration has only been created locally and has not yet been saved in the server extension.
- NOSOURCEHANDLER: The configuration could not be loaded correctly in the server extension and should be saved again.


Data Source Management

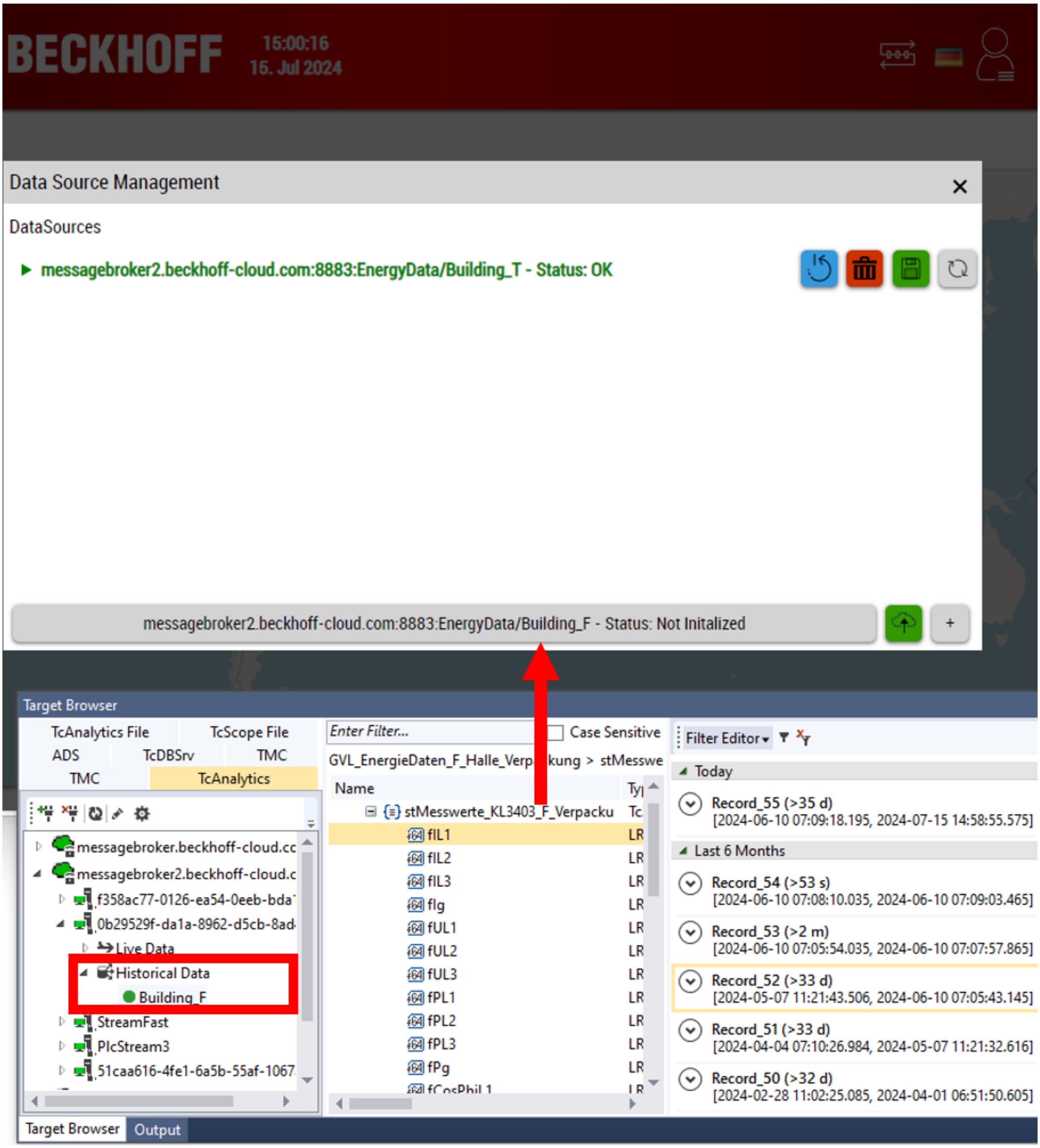
DataSources

- 172.17.64.215:1883:TestSignals/StreamFast - Status: OBSOLETE
- messagebroker2.beckhoff-cloud.com:8883:EnergyData/Building_F - Status: OK
- messagebroker2.beckhoff-cloud.com:8883:SlideData/PlcStream1 - Status: NOCONNECTION
- messagebroker2.beckhoff-cloud.com:8883:EnergyData/Building_T - Status: INIT
- <broker>:1883:<streamTopic> - Status: undefined

Creating a data source

New data sources can be added simply by drag and drop from the TargetBrowser. In the **TcAnalytics** tab, select Historical Data sources.
All symbols are automatically added to the data source.

If the parameters have been read successfully, the upload icon lights up green. Click on the upload icon to send the configuration to the server extension.



Data Source Management

DataSources

▸ messagebroker2.beckhoff-cloud.com:8883:EnergyData/Building_T - Status: OK

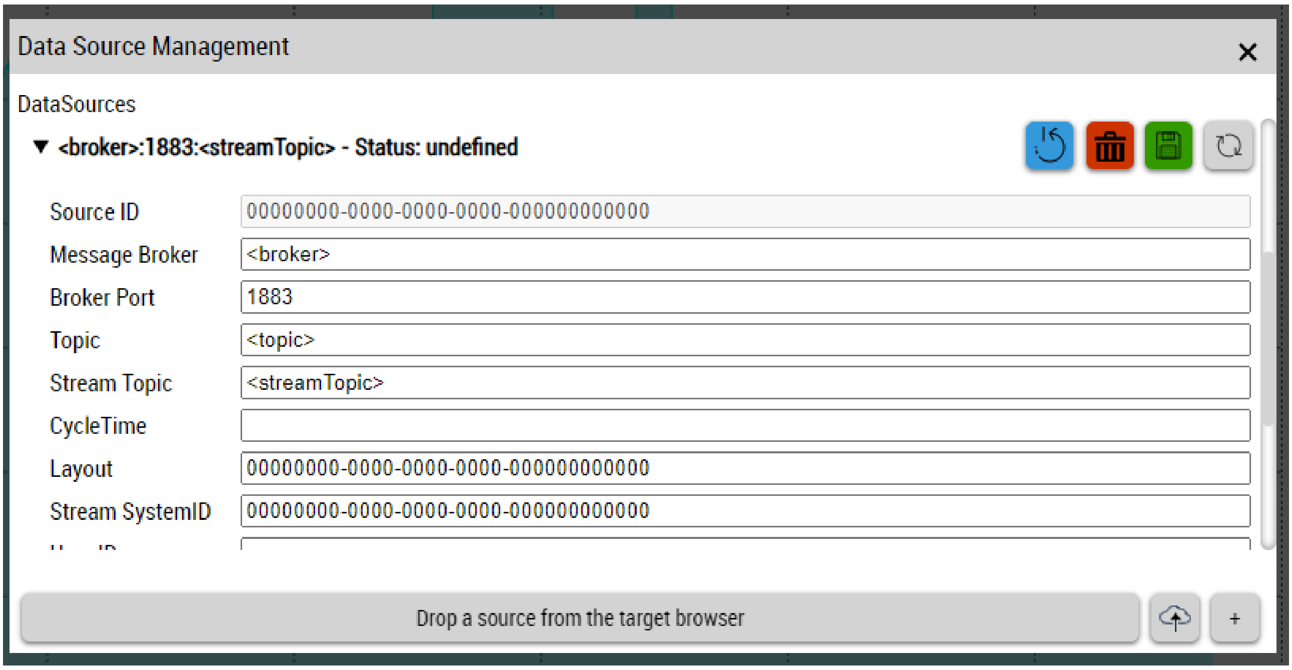
messagebroker2.beckhoff-cloud.com:8883:EnergyData/Building_F - Status: Not Initalized

BECKHOFF
15:00:16
15. Jul 2024
+
Target Browser
TcAnalytics File
TcScope File
ADS
TcDBSrv
TMC
TMC
TcAnalytics
messagebroker.beckhoff-cloud.cc
messagebroker2.beckhoff-cloud.c
f358ac77-0126-ea54-0eeb-bda
0b29529f-da1a-8962-d5cb-8ad
Live Data
Historical Data
Building_F
StreamFast
PlcStream3
51caa616-4fe1-6a5b-55af-1067
Enter Filter...
Case Sensitive
GVL_EnergieDaten_F_Halle_Verpackung > stMesswe
Name
Ty
stMesswerte_KL3403_F_Verpacku
Tc
fIL1
fIL2
fIL3
fIg
fUL1
fUL2
fUL3
fPL1
fPL2
fPL3
fPg
fCosPhil 1
LR
Filter Editor
Today
Record_55 (>35 d) [2024-06-10 07:09:18.195, 2024-07-15 14:58:55.575]
Last 6 Months
Record_54 (>53 s) [2024-06-10 07:08:10.035, 2024-06-10 07:09:03.465]
Record_53 (>2 m) [2024-06-10 07:05:54.035, 2024-06-10 07:07:57.865]
Record_52 (>33 d) [2024-05-07 11:21:43.506, 2024-06-10 07:05:43.145]
Record_51 (>33 d) [2024-04-04 07:10:26.984, 2024-05-07 11:21:32.616]
Record_50 (>32 d) [2024-02-28 11:02:25.085, 2024-04-01 06:51:50.605]
Target Browser
Output

If the Target Browser is not available on the device, you can create an empty data source using the plus icon in the bottom right-hand corner. This must be configured manually.

Configuring and changing a data source

You can use the arrow to the left of a data source to expand and edit it.



Data Source Management

DataSources

▼ <broker>:1883:<streamTopic> - Status: undefined

Source ID	00000000-0000-0000-0000-000000000000
Message Broker	<broker>
Broker Port	1883
Topic	<topic>
Stream Topic	<streamTopic>
CycleTime	
Layout	00000000-0000-0000-0000-000000000000
Stream SystemID	00000000-0000-0000-0000-000000000000

The following parameters can be set:

Parameter	Description
Source ID	Unique ID of a data source. This is unique and is assigned by the server extension.
Message Broker	Address of the message broker
Broker Port	Port of the message broker
Topic	MQTT topic of the historical stream „…/TcAnalyticsStorageProvider/…/Historical/Stream…“
StreamTopic	Original topic of the recorded stream.
Cycle Time	Cycle time of the data source in 100 nanoseconds.
Layout	Layout of the stream
Stream SystemID	SystemID of the recorded stream.
UserID	User name for accessing the message broker.
Password	Password for access to the message broker.
With certificates	If the value is checked, the selected certificate files are used.
CA	CA file (can be uploaded to the server extension).
CERT	CERT file (can be uploaded to the server extension).
KEY	KEY file (can be uploaded to the server extension).
Key Password	Password for the KEY file.

The changes can be sent to the server extension via the save icon (green). Alternatively, the changes made locally can be undone using the reset icon (blue). The status of the data source can be queried again via the refresh icon (gray).

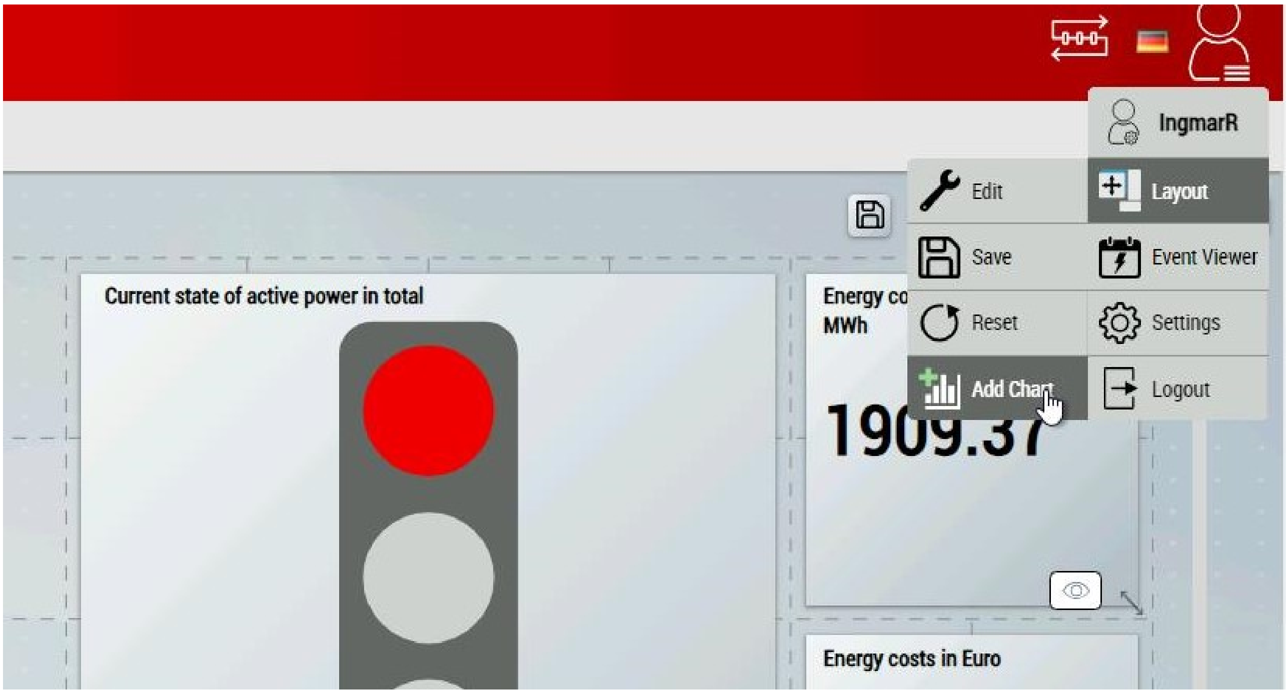
Deleting a data source

A data source can be deleted using the corresponding delete icon (red). Please note that all charts using this data source will no longer work. If parameters are changed, such as the broker address or the renewal of certificates, it is advisable to adapt existing data sources and not to delete them.

6.8.7.2 Adding and removing charts

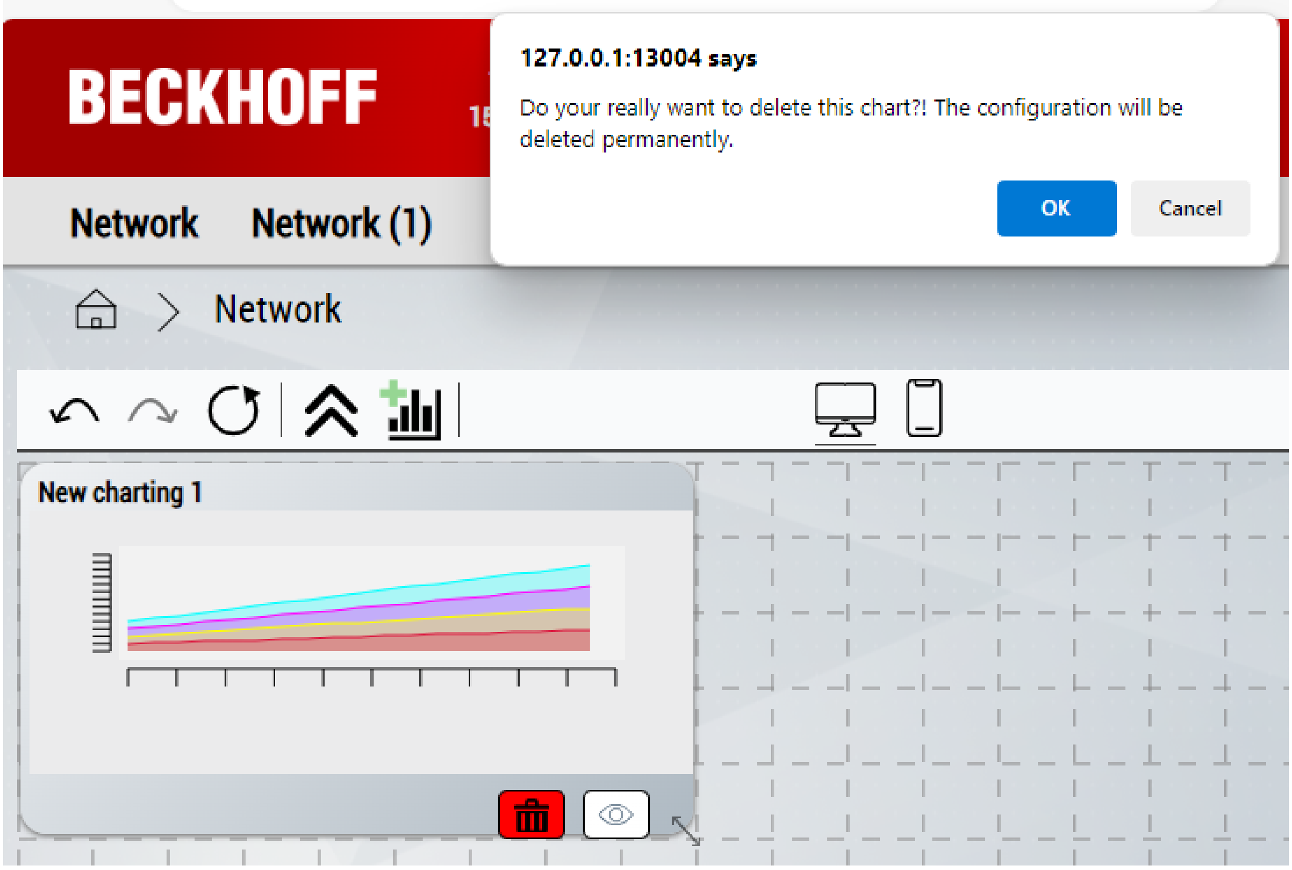
In the web browser

Add new charts via the menu icon. Click on **Add Chart** under the Layout category. The new chart is inserted at the first free position in the layout.



Charts can only be added if the user has authorization to edit the layout.

The added chart is user-specific and can only be viewed by the current user. Its position and size can be moved within the layout as required. You can remove a chart in Edit mode using the recycle bin icon in the bottom right-hand corner.

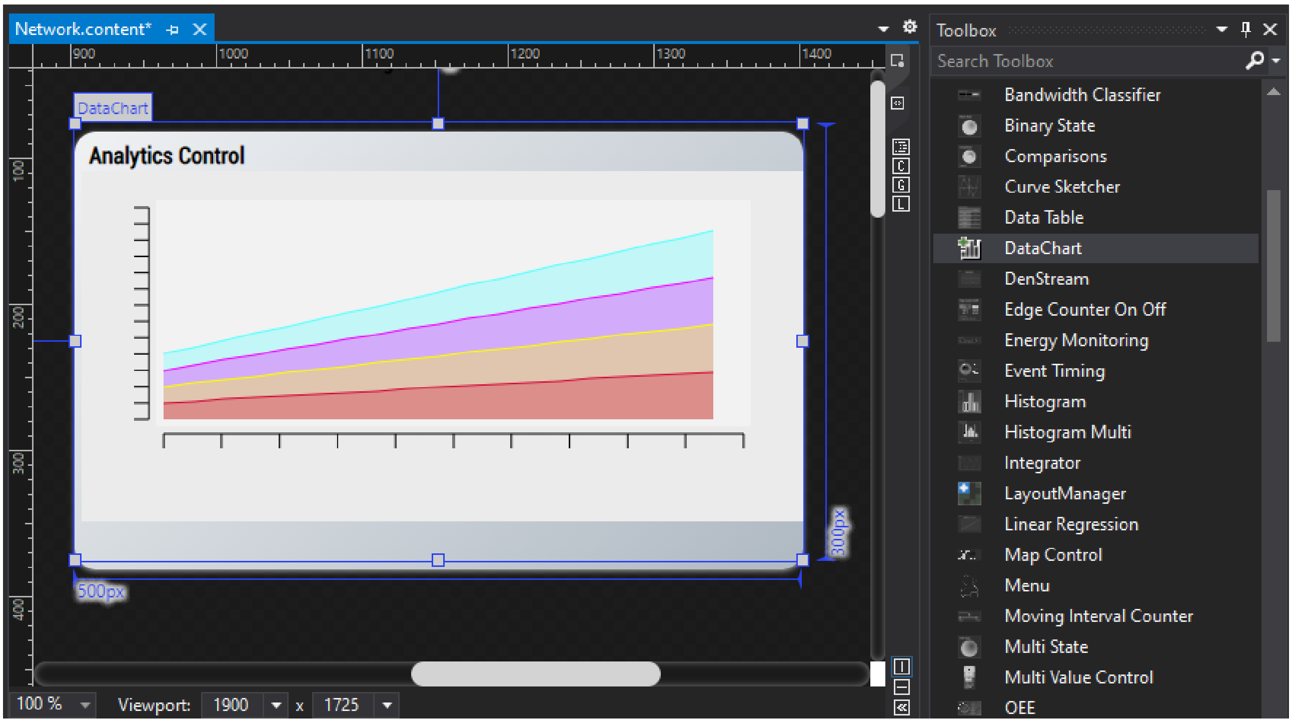


127.0.0.1:13004 says

Do your really want to delete this chart?! The configuration will be deleted permanently.

In HMI engineering

Charts can also be added via the HMI engineering. These charts are available to all users by default and cannot be removed. The configuration [▶ 476] of charts added in Engineering is carried out in the web browser.

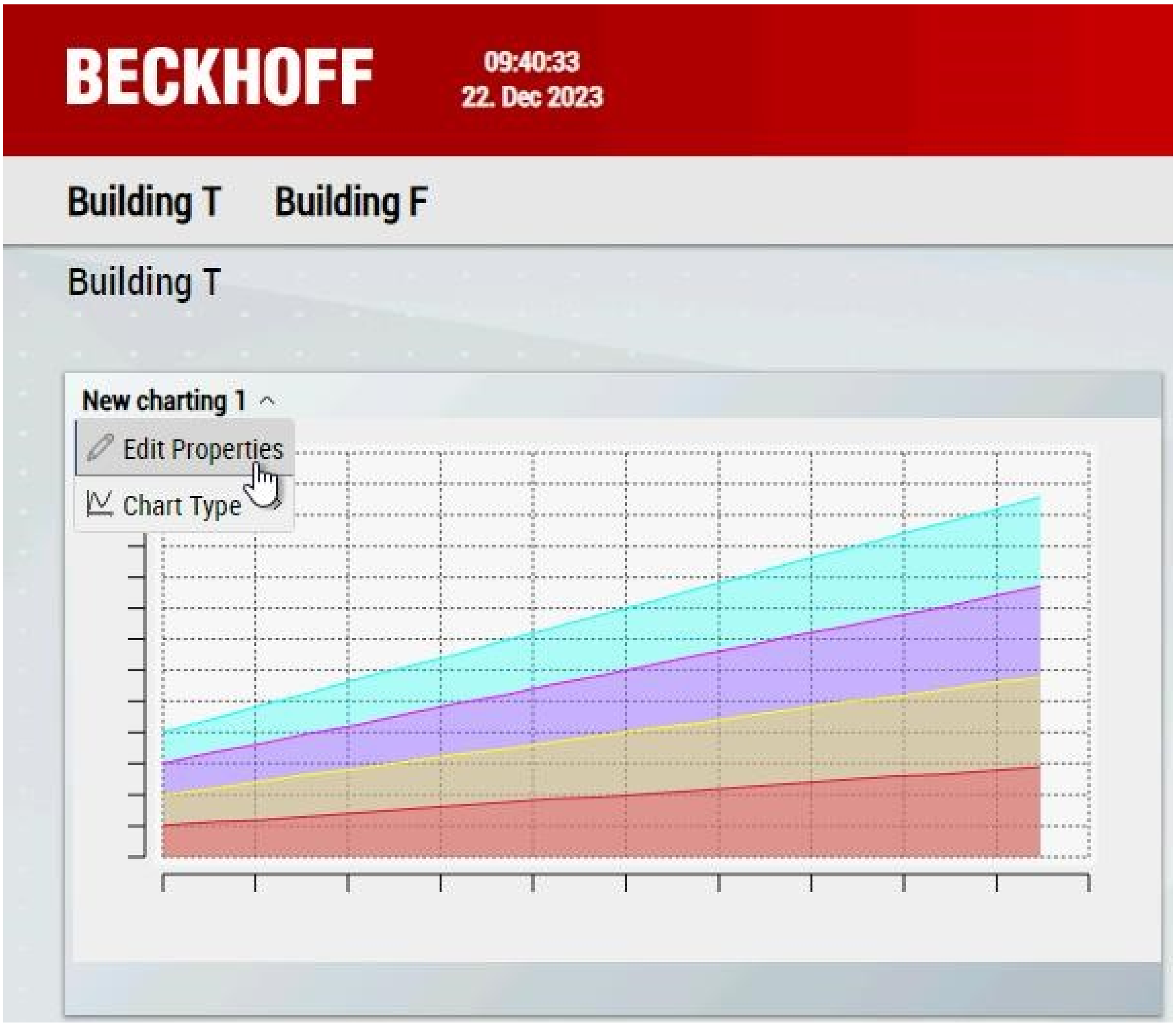


6.8.7.3 Configuration of a chart

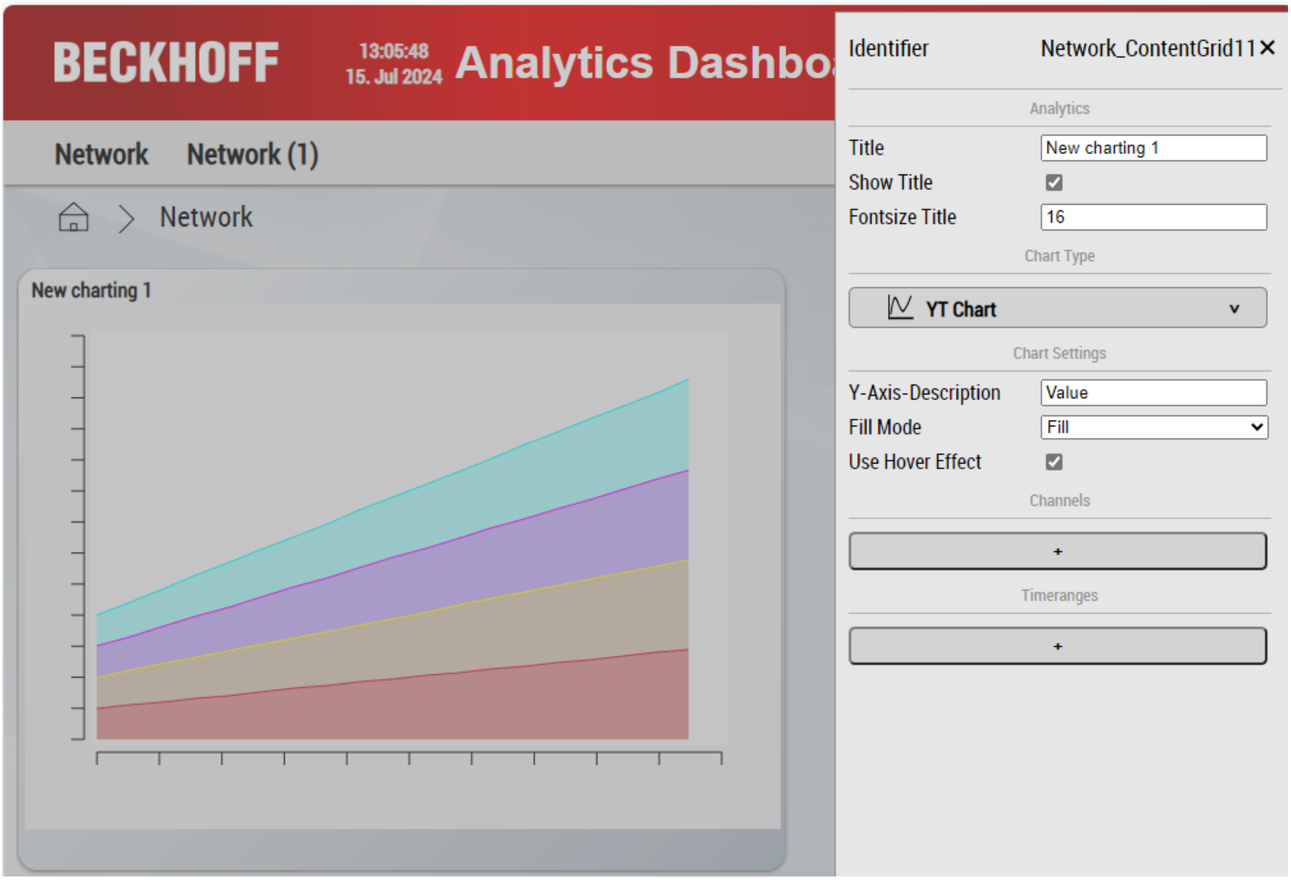
A chart is configured in the web browser. You can edit all settings (such as chart type, variables and time periods) via the properties window of the chart.

Open of the chart properties

Open the chart properties by clicking on the title of the Chart control.



Click on **Edit Properties** to open the chart properties.



Identifier Network_ContentGrid11 ✕

Analytics

Title: New charting 1

Show Title ☑

Fontsize Title: 16

Chart Type

YT Chart

Chart Settings

Y-Axis-Description: Value

Fill Mode: Fill

Use Hover Effect ☑

Channels

+

Timeranges

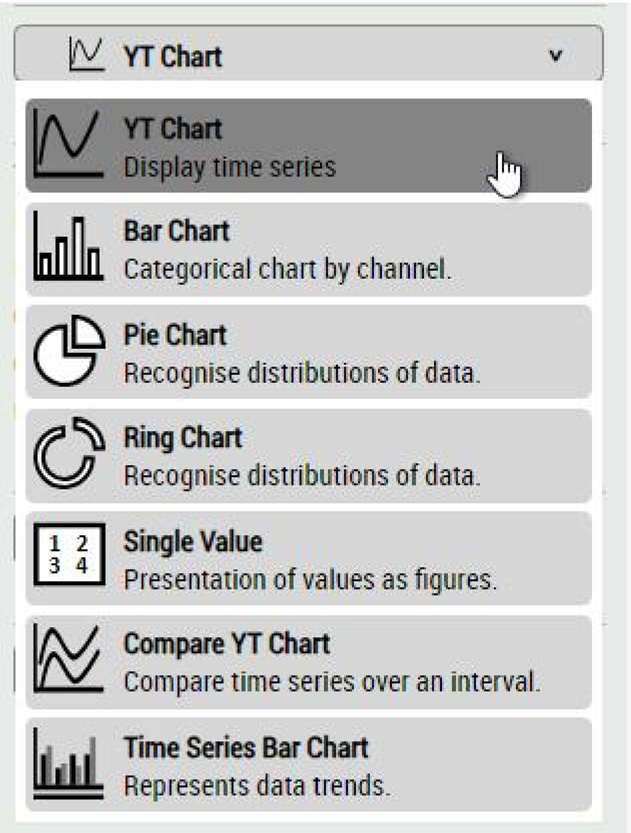
+

The chart properties are divided into 4 categories.

- Chart Type: Selection of the chart type
- Chart Settings: Configuration of the chart
- Channels: Selection of variables and their display.
- Timeranges: Selection of time ranges to be displayed.

6.8.7.3.1 Chart type

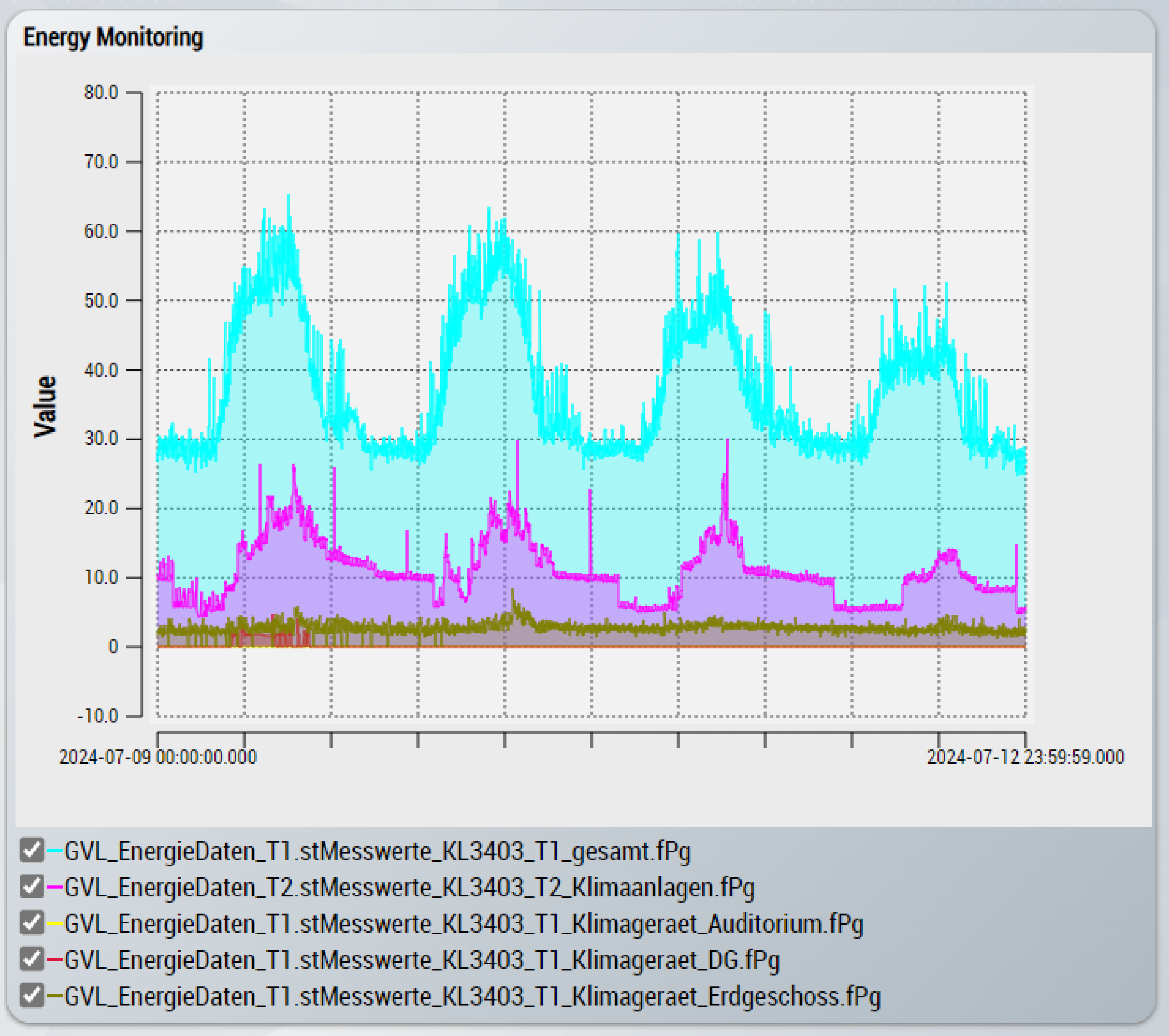
You can set the chart type via the properties of the chart.



YT Chart
YT Chart
Display time series
Bar Chart
Categorical chart by channel.
Pie Chart
Recognise distributions of data.
Ring Chart
Recognise distributions of data.
Single Value
Presentation of values as figures.
Compare YT Chart
Compare time series over an interval.
Time Series Bar Chart
Represents data trends.

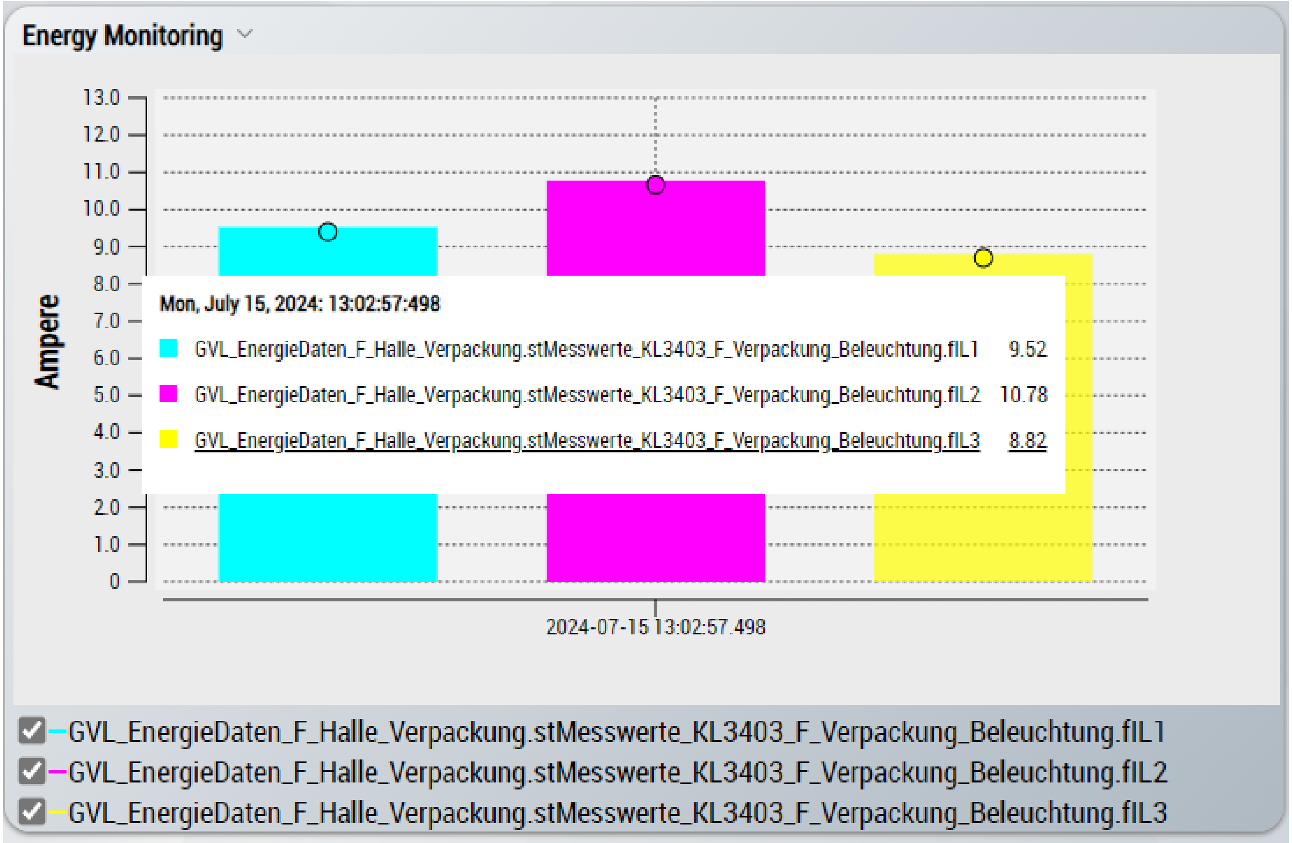
You can use the following chart types:

YT-Chart



Display of the time on the X-axis and the corresponding values on the Y-axis.

Bar-Chart

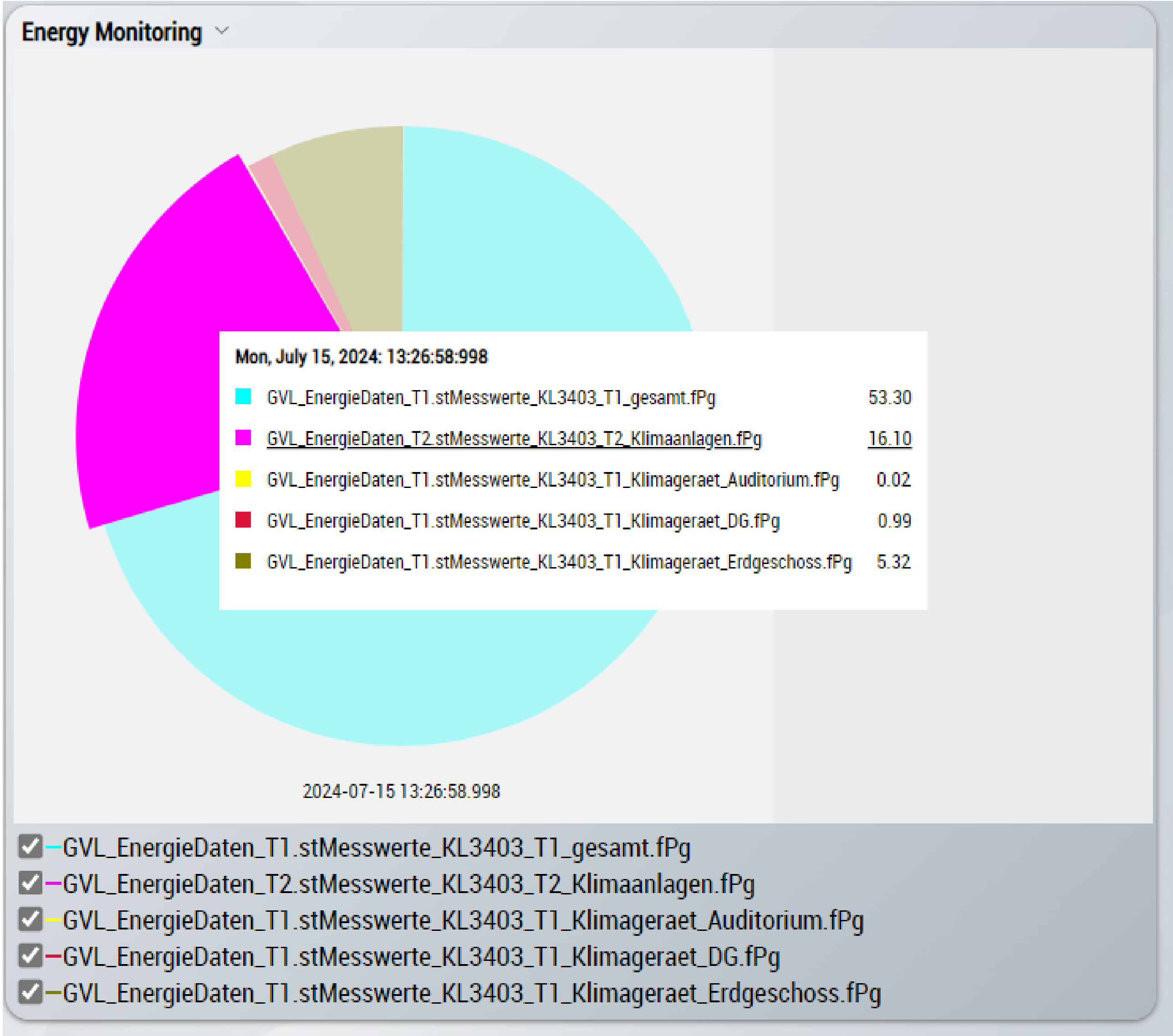

Energy Monitoring
Ampere
13.0
12.0
11.0
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0
2024-07-15 13:02:57.498
☑ GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fIL1
☑ GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fIL2
☑ GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fIL3

Mon, July 15, 2024: 13:02:57:498

🟦	GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fIL1	9.52
🟪	GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fIL2	10.78
🟨	GVL_EnergieDaten_F_Halle_Verpackung.stMesswerte_KL3403_F_Verpackung_Beleuchtung.fIL3	8.82

One bar is displayed per selected time range and per variable. The bar represents the last value of the time range.

Pie-Chart



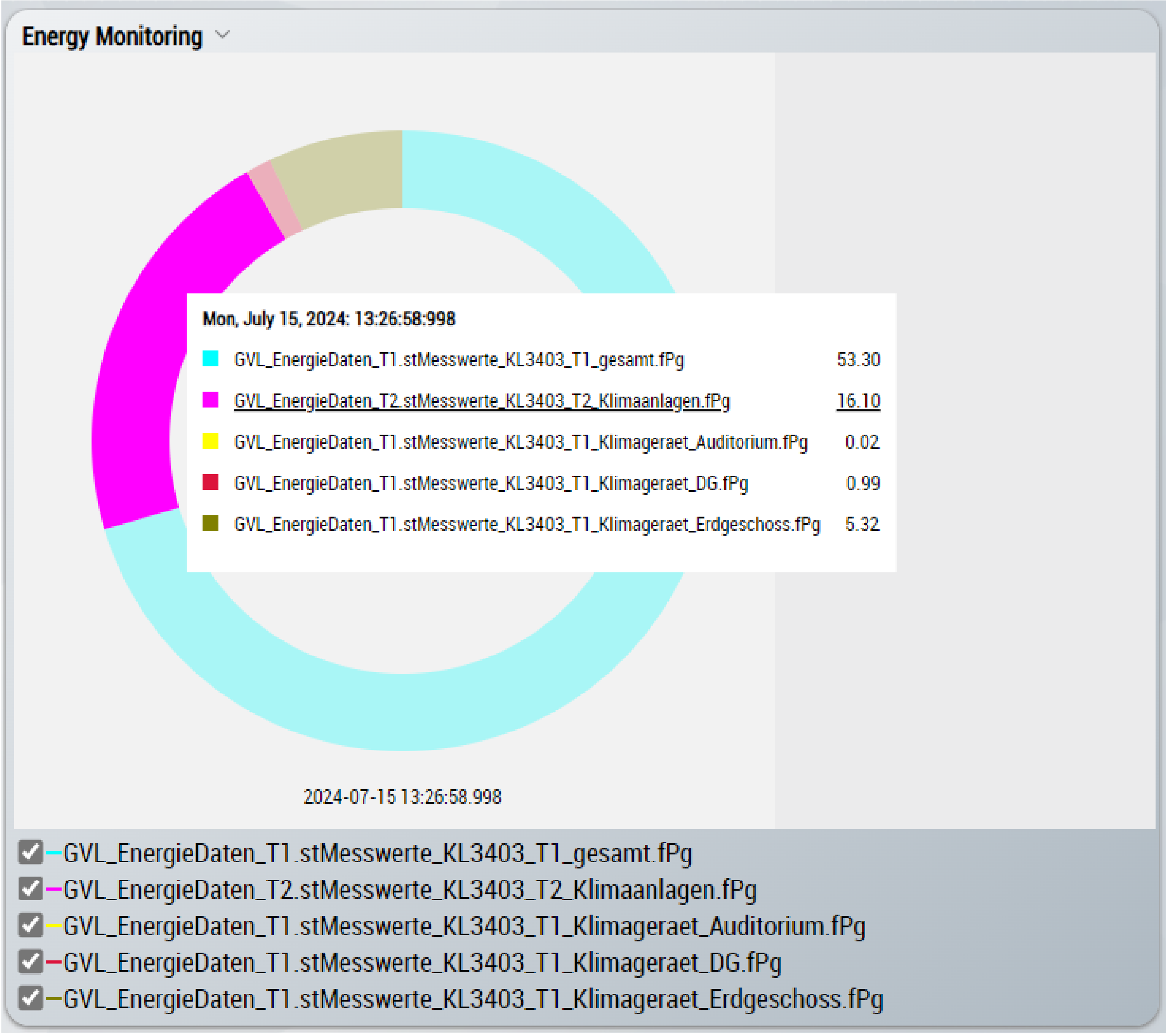
Mon, July 15, 2024: 13:26:58:998

🟦	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	53.30
🟪	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	16.10
🟨	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Auditorium.fPg	0.02
🟥	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_DG.fPg	0.99
🟫	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Erdgeschoss.fPg	5.32

☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg
☑ GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Auditorium.fPg
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_DG.fPg
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Erdgeschoss.fPg

A pie is displayed for each selected time range, which represents the last sample of the time range. Negative variable values are not displayed.

Ring-Chart



Energy Monitoring

Mon, July 15, 2024: 13:26:58:998

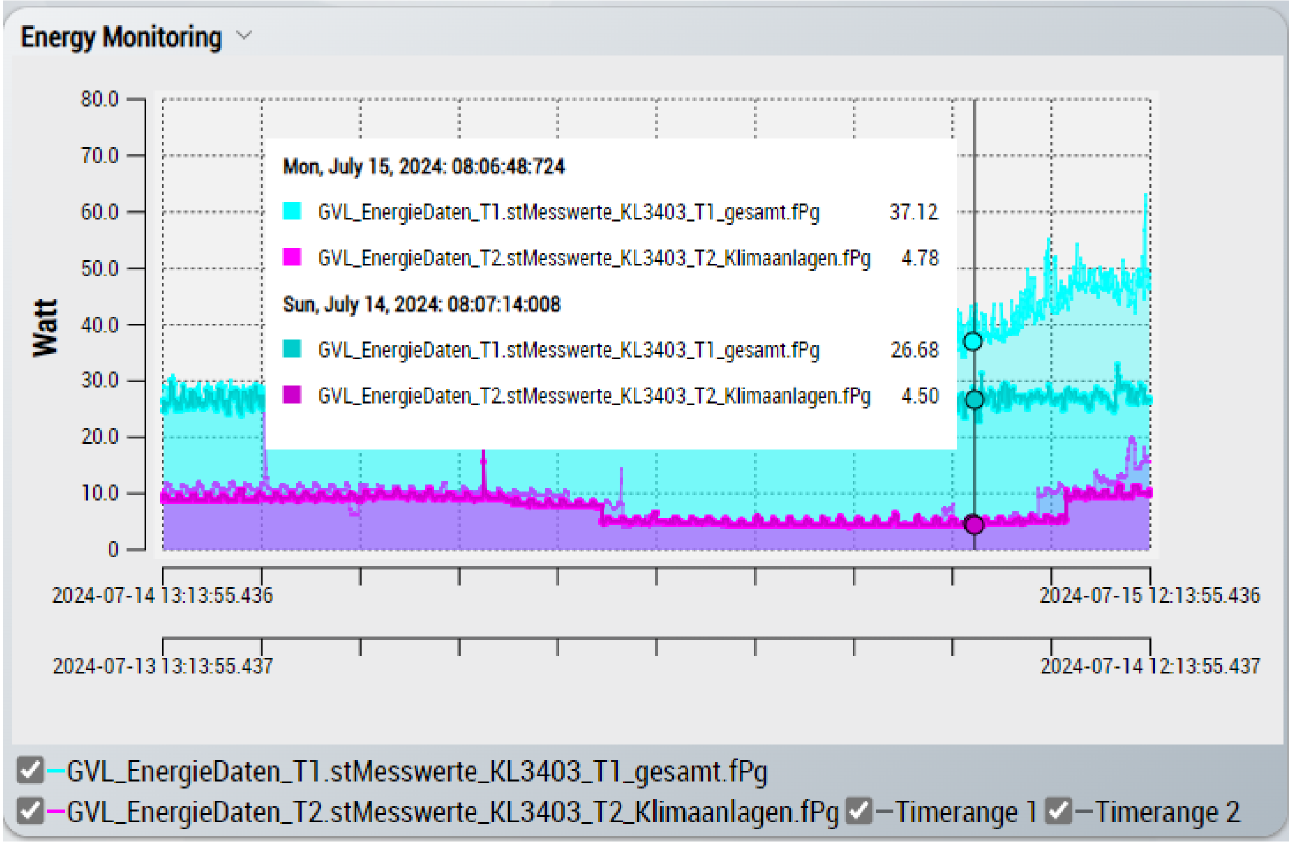
🟦	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	53.30
🟪	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	16.10
🟨	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Auditorium.fPg	0.02
🟥	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_DG.fPg	0.99
🟫	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Erdgeschoss.fPg	5.32

2024-07-15 13:26:58.998

☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg
☑ GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Auditorium.fPg
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_DG.fPg
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_Klimageraet_Erdgeschoss.fPg

A ring is displayed for each selected time range, which represents the last sample of the time range. Negative variable values are not displayed.

Compare-YT-Chart



Mon, July 15, 2024: 08:06:48:724

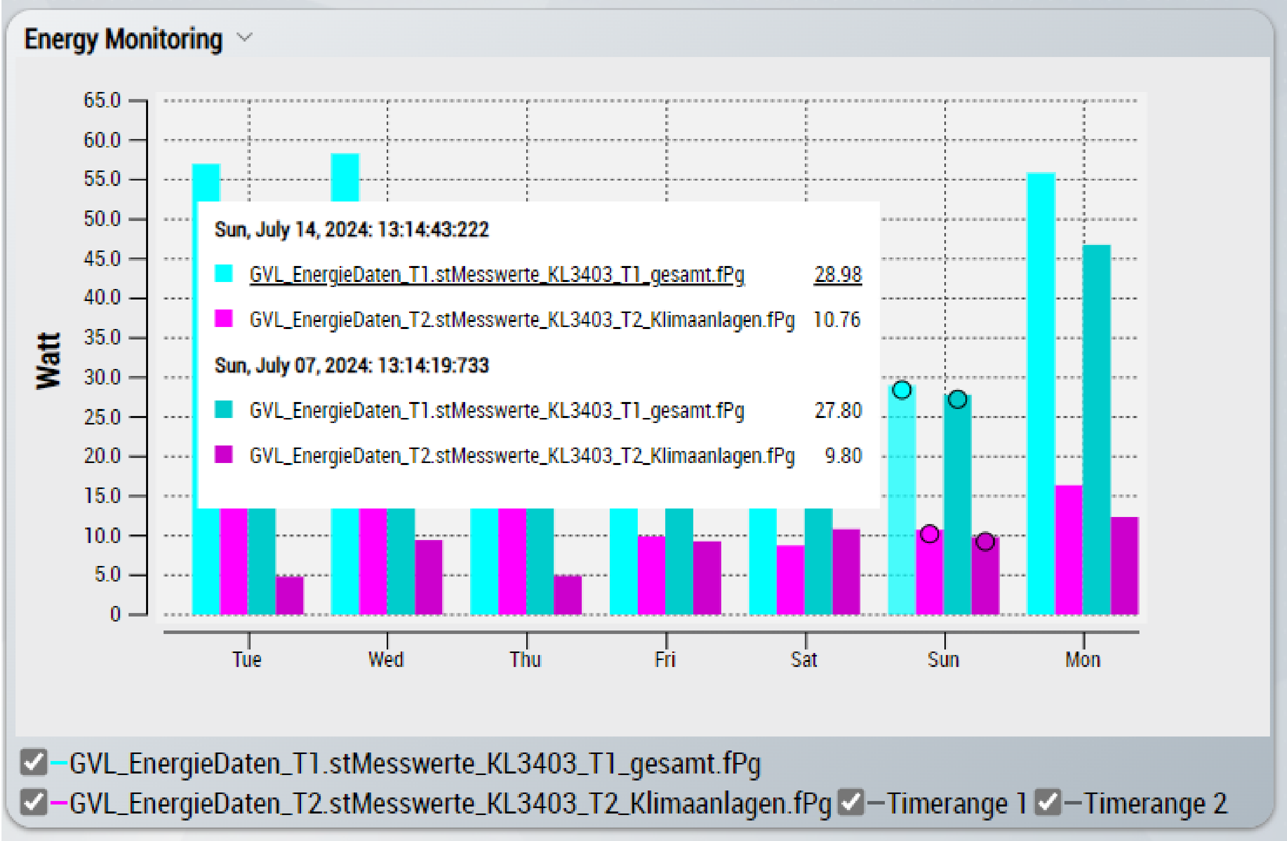
■	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	37.12
■	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	4.78

Sun, July 14, 2024: 08:07:14:008

■	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	26.68
■	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	4.50

Suitable for comparing several time series. This involves comparing the data over the selected comparison interval. Several comparison modes are available for the comparison. The comparison period is shown on the X-axis.

Time-Series-Bar-Chart



Sun, July 14, 2024: 13:14:43:222

■	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	28.98
■	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	10.76

Sun, July 07, 2024: 13:14:19:733

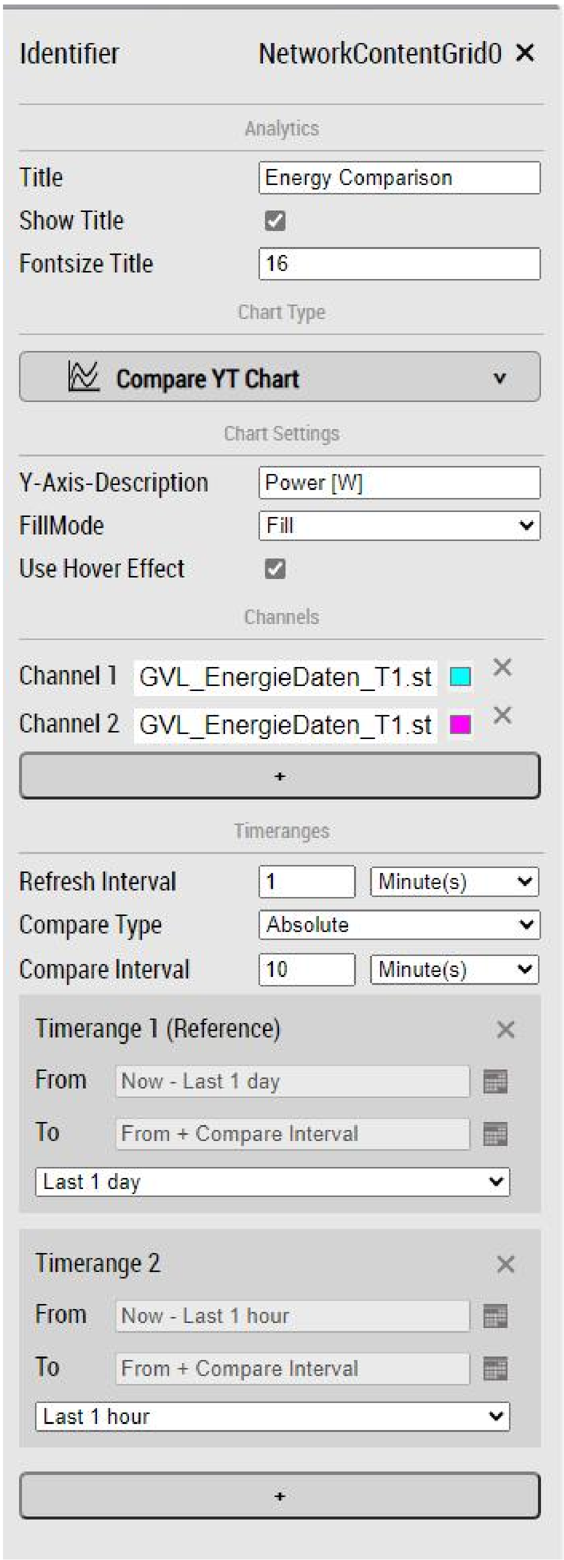
■	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	27.80
■	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	9.80

Energy Monitoring
Watt
65.0
60.0
55.0
50.0
45.0
40.0
35.0
30.0
25.0
20.0
15.0
10.0
5.0
0
Tue
Wed
Thu
Fri
Sat
Sun
Mon
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg
☑ GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg ☑ Timerange 1 ☑ Timerange 2

Suitable for comparing several time series. This involves comparing the data over the selected comparison interval. Several comparison modes are available for the comparison. Bar charts are generated per selected timespan and per variable based on the comparison timespan.

6.8.7.3.2 General settings

You can make general chart settings in the **Chart Settings** area.



Identifier NetworkContentGrid0 ✕

Analytics

Title: Energy Comparison

Show Title: ☑

Fontsize Title: 16

Chart Type

Compare YT Chart

Chart Settings

Y-Axis-Description: Power [W]

FillMode: Fill

Use Hover Effect: ☑

Channels

Channel 1: GVL_EnergieDaten_T1.st 🟦 ✕

Channel 2: GVL_EnergieDaten_T1.st 🟪 ✕

+

Timeranges

Refresh Interval: 1 Minute(s)

Compare Type: Absolute

Compare Interval: 10 Minute(s)

Timerange 1 (Reference) ✕

From: Now - Last 1 day

To: From + Compare Interval

Last 1 day

Timerange 2 ✕

From: Now - Last 1 hour

To: From + Compare Interval

Last 1 hour

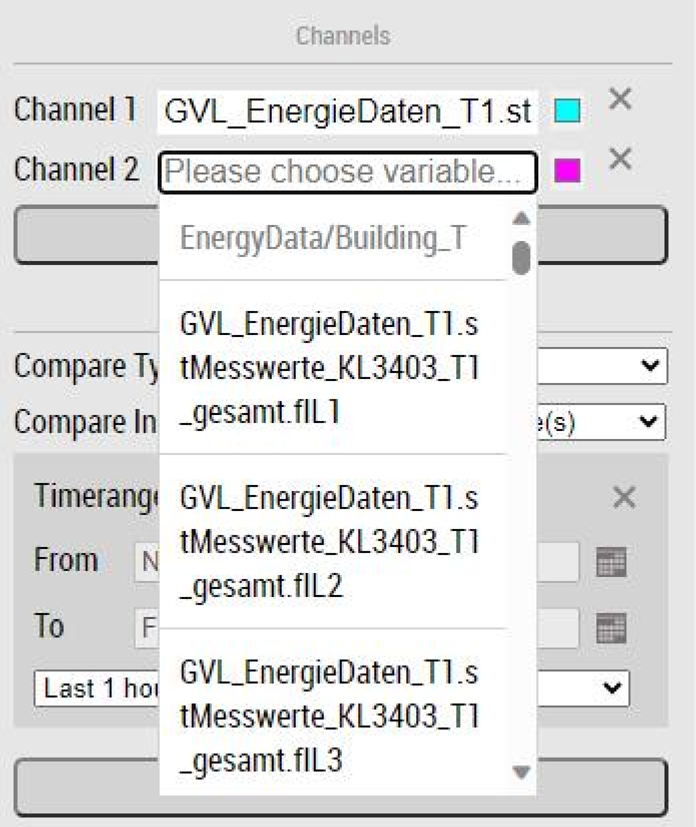
+

The following options can be set here:

Property	Description
Y-Axis-Description	Labeling the Y-axis
FillMode	Selection of whether the graphs should be filled.
Use Hover Effect	If this option is enabled and the mouse pointer is in the chart, the exact time points and variable values are displayed in a window.

6.8.7.3.3 Adding variables

In the **Channels** area, you can select the variables to be displayed. Use the plus icon to add a new channel for the chart. Define a variable for the channel via a drop-down window. Use the color icon to assign an individual color for the channel. Remove the channel using the X icon.



Channels

Channel 1: GVL_EnergieDaten_T1.st

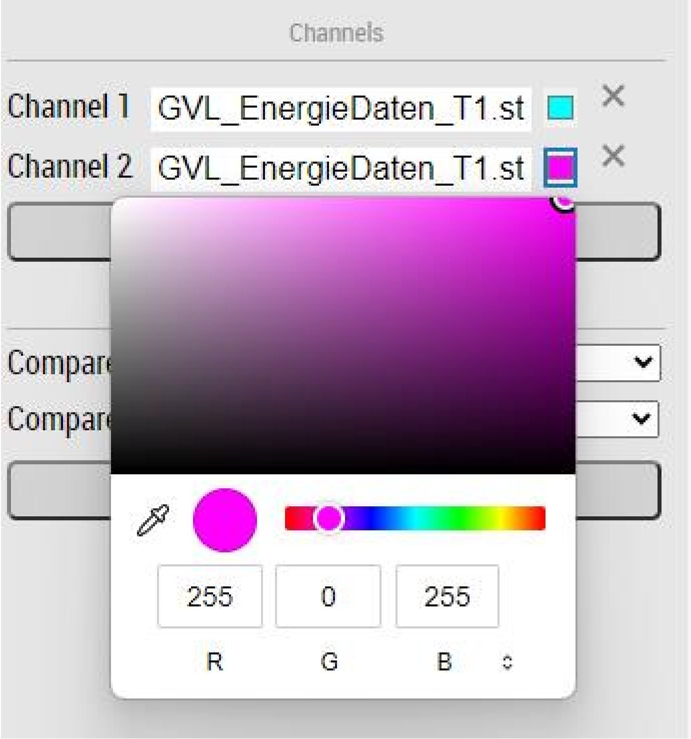
Channel 2: Please choose variable...

EnergyData/Building_T

GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fIL1

GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fIL2

GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fIL3



Channel 1 GVL_EnergieDaten_T1.st
Channel 2 GVL_EnergieDaten_T1.st

255 | 0 | 255

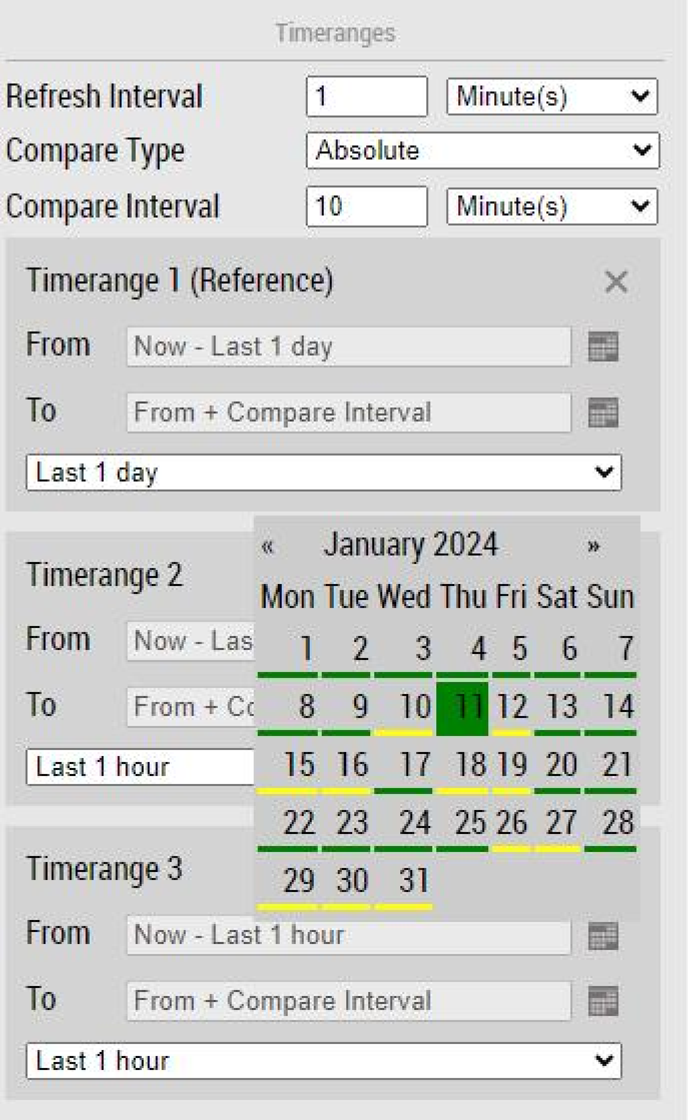
6.8.7.3.4 Adding time ranges

The time ranges to be displayed can be selected in the "Timerranges" area. A new time range can be added to the chart using the plus icon.

Absolute time ranges

For a time range, the start and end time can be selected using the "From" and "To" parameters. The date can be selected from the calendar (opened by clicking on the calendar icon). The days are marked in color in the calendar:

- Green: Day has been fully recorded.
- Yellow: Day was only partially recorded.
- Gray: No data available for this day.



Timeranges

Refresh Interval: 1 Minute(s)

Compare Type: Absolute

Compare Interval: 10 Minute(s)

Timerange 1 (Reference)

From: Now - Last 1 day

To: From + Compare Interval

Last 1 day

Timerange 2

From: Now - Las

To: From + Co

Last 1 hour

Timerange 3

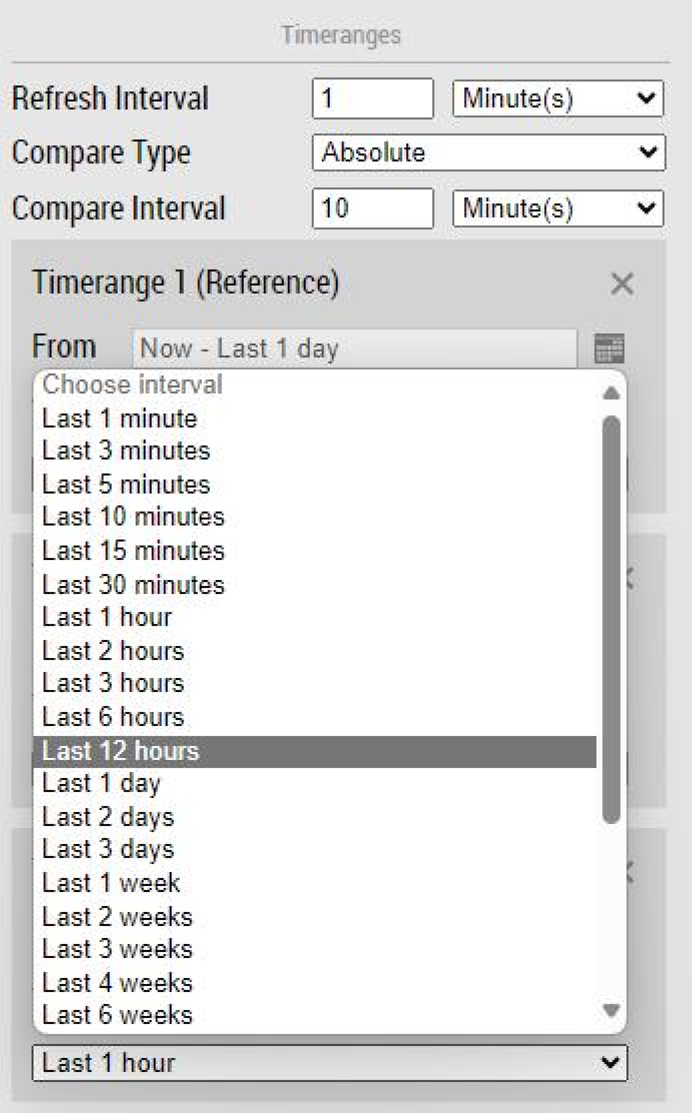
From: Now - Last 1 hour

To: From + Compare Interval

Last 1 hour

Relative time ranges

As an alternative to absolute time ranges, relative time ranges can also be selected. The selection of a relative time range can be made via the drop-down window. As soon as at least one relative time range has been selected, the "Refresh Interval" property can be used to specify the time interval at which the chart is updated.



Timeranges

Refresh Interval: 1 Minute(s)

Compare Type: Absolute

Compare Interval: 10 Minute(s)

Timerange 1 (Reference)

From: Now - Last 1 day

Last 1 hour

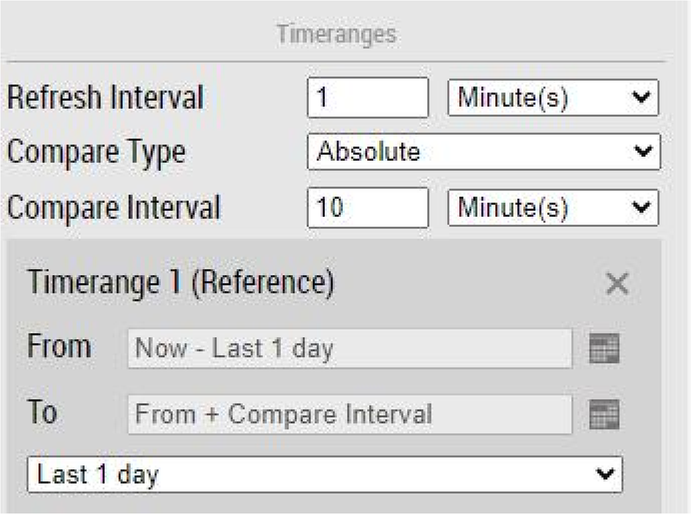
Automatic setting of the end time

For the Compare-YT-Chart and Time-Series-Bar-Chart, the end time of a time range is set based on the configured comparison interval. Manual selection of the end time is not possible in these chart types.

6.8.7.4 Comparing data

The CompareCompare-YT-Chart and TimeSeries-Bar-Chart can be used to compare data from different time ranges. Up to three time ranges can be added in the settings for this purpose. The first time range serves as a reference for the display.

The data is compared over a defined timespan (e.g. 3 hours, 1 week, ...) The length of the comparison period can be configured in the "Timeranges" area if Compare-YT-Chart or TimeSeries-Bar-Chart is selected as the chart type.



Timeranges

Refresh Interval: 1 Minute(s)

Compare Type: Absolute

Compare Interval: 10 Minute(s)

Timerange 1 (Reference)

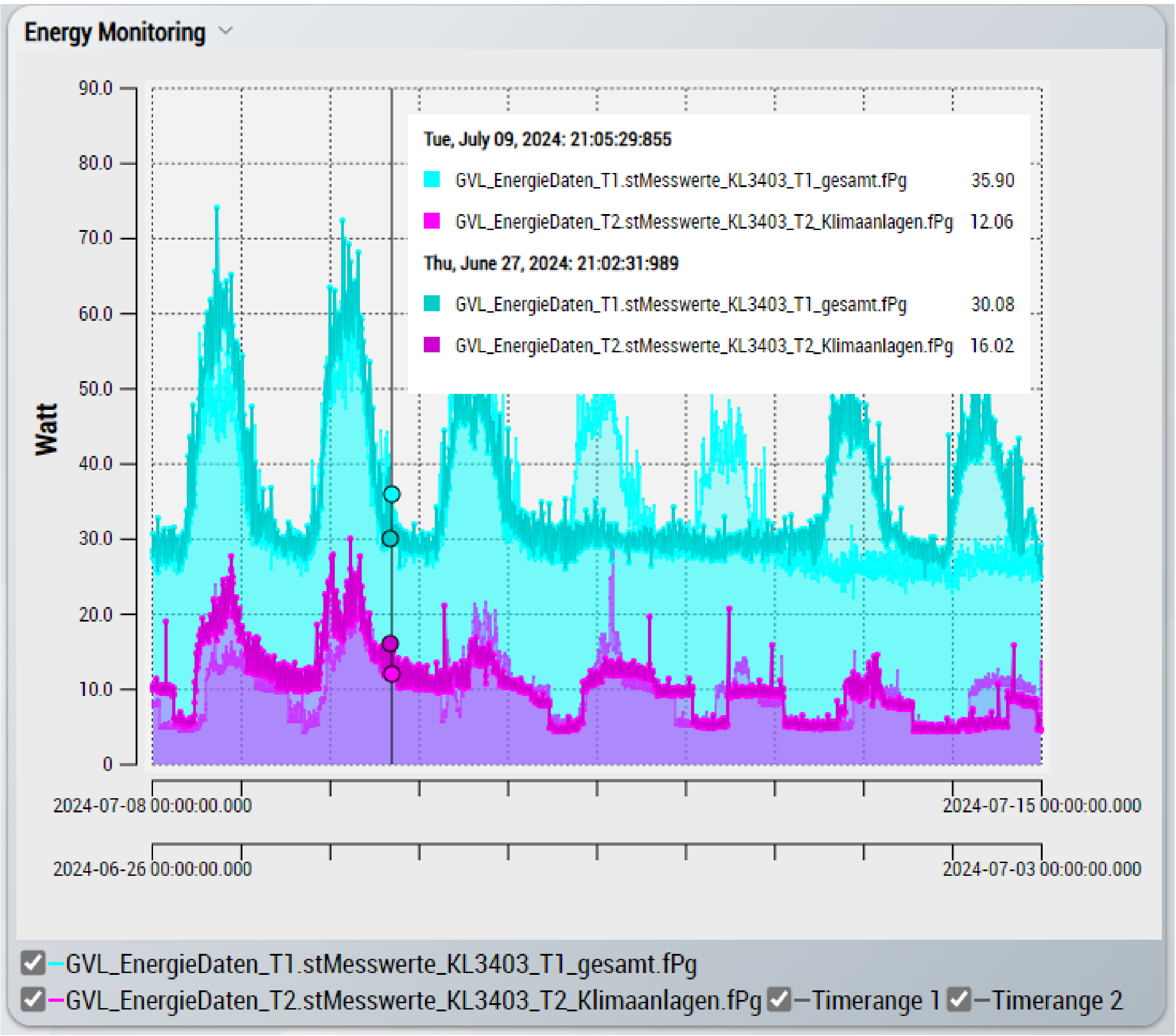
From: Now - Last 1 day

To: From + Compare Interval

Last 1 day

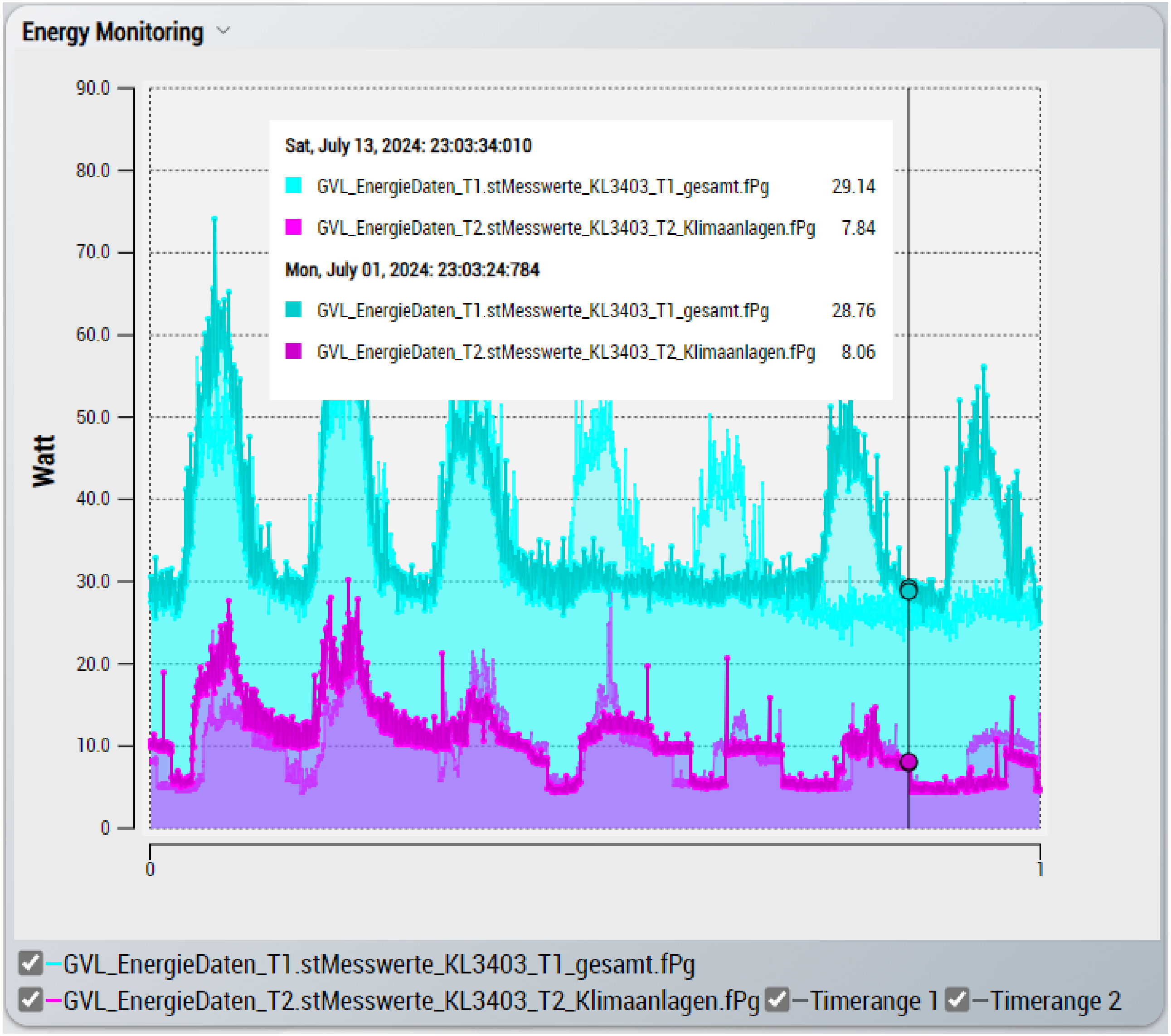
In addition to the timespan, a comparison mode can be set. This can be used to set whether the data should simply be displayed on top of each other or whether they should be displayed synchronized with each other. This corresponds to a shift on the time axis in the diagram. The available comparison modes are:

Absolute



In this mode, all selected time ranges are visualized one above the other. The absolute time is displayed on the X-axis. There is no shift on the time axis. The origin is always the start time of a timespan.

Relative



Energy Monitoring
Watt
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0
0
1
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg
☑ GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg ☑ Timerange 1 ☑ Timerange 2

Sat, July 13, 2024: 23:03:34:010

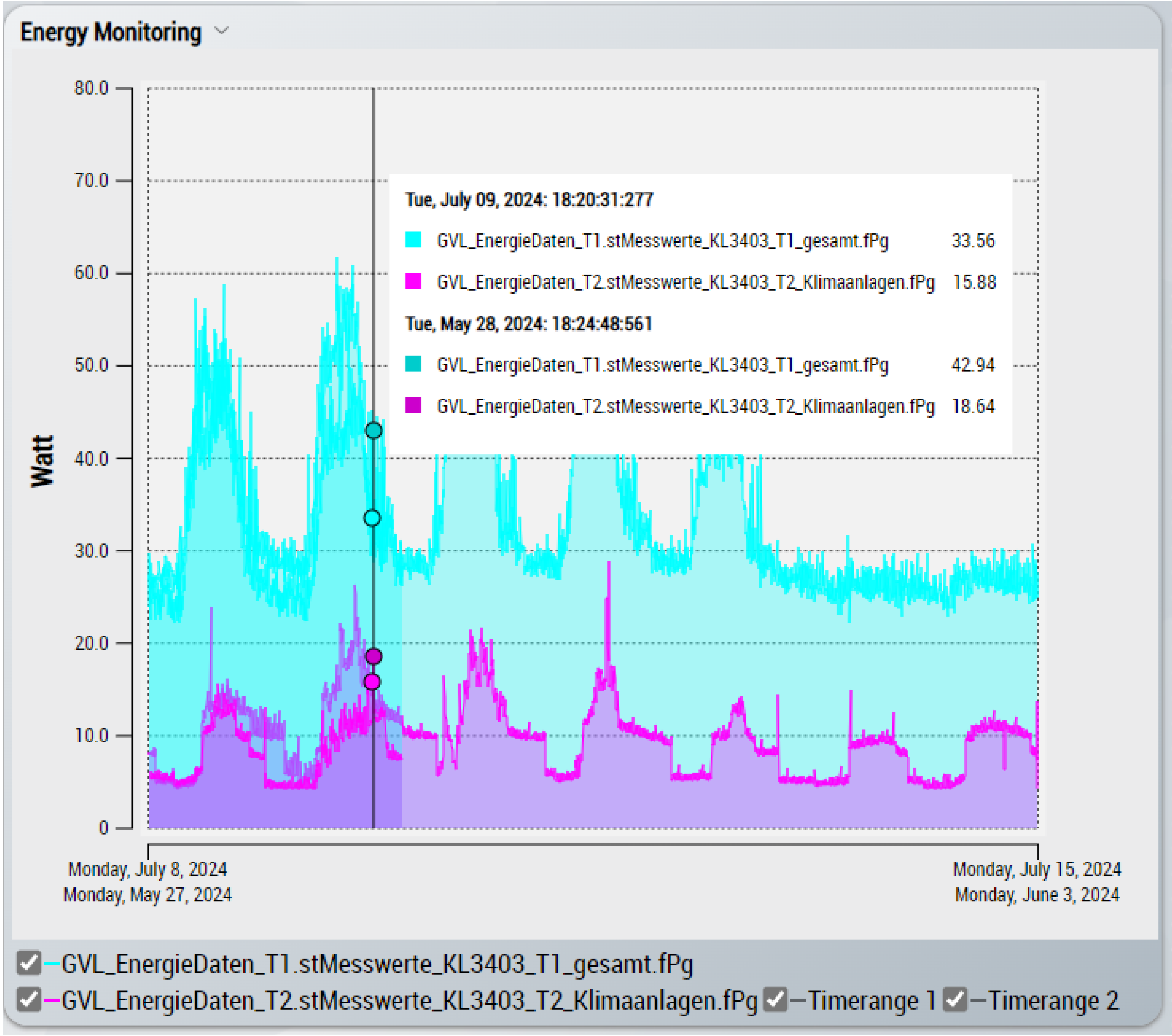
GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	29.14
GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	7.84

Mon, July 01, 2024: 23:03:24:784

GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	28.76
GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	8.06

In this mode, all selected time ranges are visualized one above the other. The relative elapsed time is shown on the X-axis. There is no shift on the time axis.

Match Start


Energy Monitoring
Watt
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0
Monday, July 8, 2024
Monday, May 27, 2024
Monday, July 15, 2024
Monday, June 3, 2024
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg
☑ GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg ☑ Timerange 1 ☑ Timerange 2

Tue, July 09, 2024: 18:20:31:277

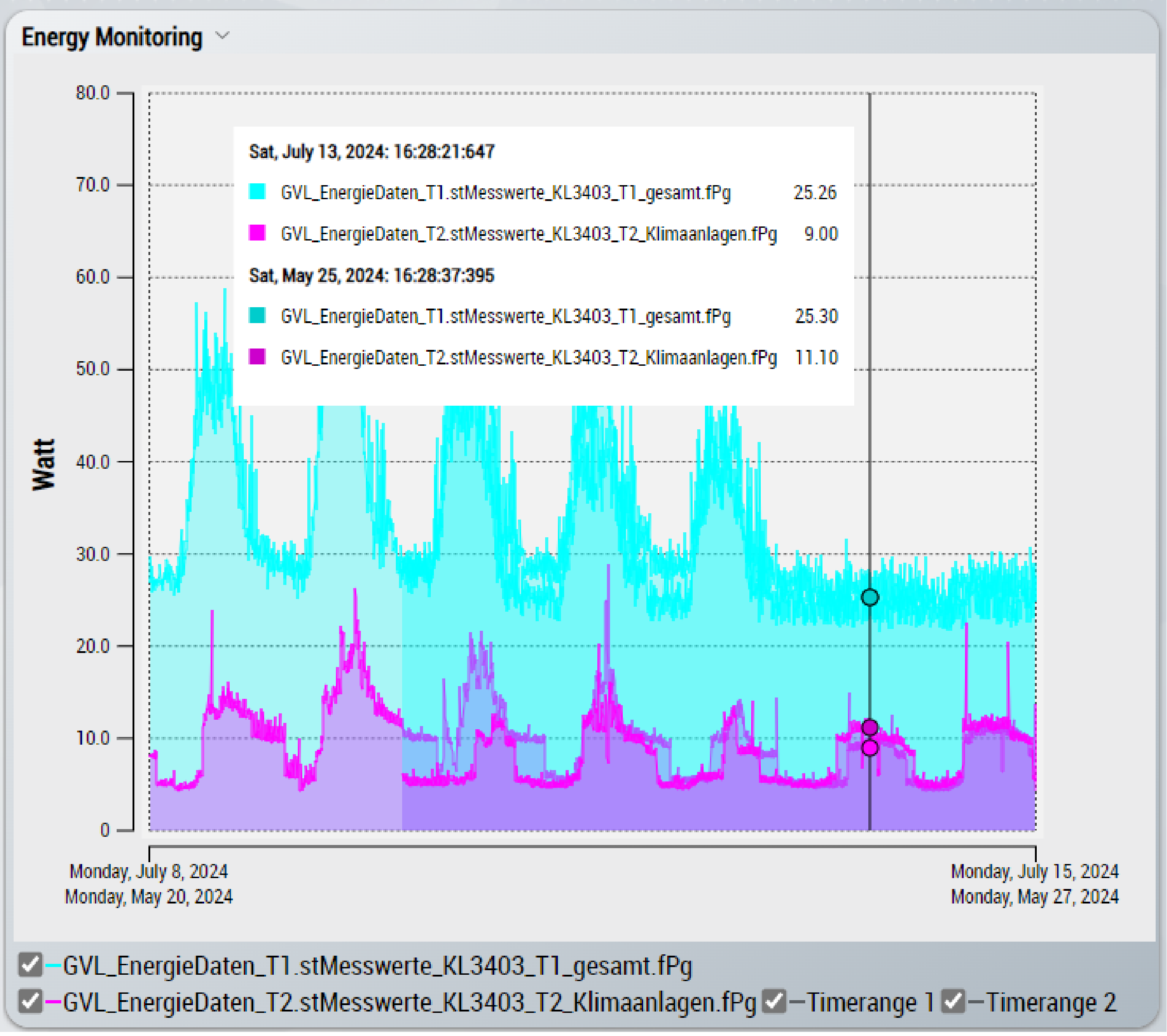
GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	33.56
GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	15.88

Tue, May 28, 2024: 18:24:48:561

GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	42.94
GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	18.64

In this mode, the data is superimposed at the correct time. The start time of the first time range serves as a reference. The data for the other points in time are shifted so that the data at the beginning of the chart are aligned. The basis for this is the selected comparison period (see table).

Match End


Energy Monitoring
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0
Watt
Monday, July 8, 2024
Monday, May 20, 2024
Monday, July 15, 2024
Monday, May 27, 2024
☑ GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg
☑ GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg ☑ Timerange 1 ☑ Timerange 2

Sat, July 13, 2024: 16:28:21:647

🟦	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	25.26
🟪	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	9.00

Sat, May 25, 2024: 16:28:37:395

🟦	GVL_EnergieDaten_T1.stMesswerte_KL3403_T1_gesamt.fPg	25.30
🟪	GVL_EnergieDaten_T2.stMesswerte_KL3403_T2_Klimaanlagen.fPg	11.10

In this mode, the data is superimposed at the correct time. The end time of the first time range serves as a reference. The data for the other points in time are shifted so that the data at the end of the chart overlap in time. The basis for this is the selected comparison period (see table).

Comparison period	Procedure
Second(s)	Comparison of the milliseconds of data. As soon as the same millisecond occurs, the data is superimposed.
Minute(s)	Comparison of the seconds of data. As soon as the same second occurs, the data is superimposed.
Hour(s)	Comparison of the minutes of the data. As soon as the same minute occurs, the data is superimposed.
Day(s)	Comparison of the hour of data. As soon as the same hour occurs, the data is superimposed.
Week(s)	Comparison of the weekday of the data. As soon as the same day of the week occurs, the data is superimposed.
Month(s)	Comparison of the day of the data. As soon as the same day (date) occurs, the data is superimposed. One month always corresponds to 31 days.
Year(s)	Comparison of the month of data. As soon as the same month occurs, the data is superimposed.

Shifting the data

As it is not guaranteed that the data is recorded at the same cycle time or that the data is scanned at the same sampling rate, it is possible that the values do not always overlap exactly.

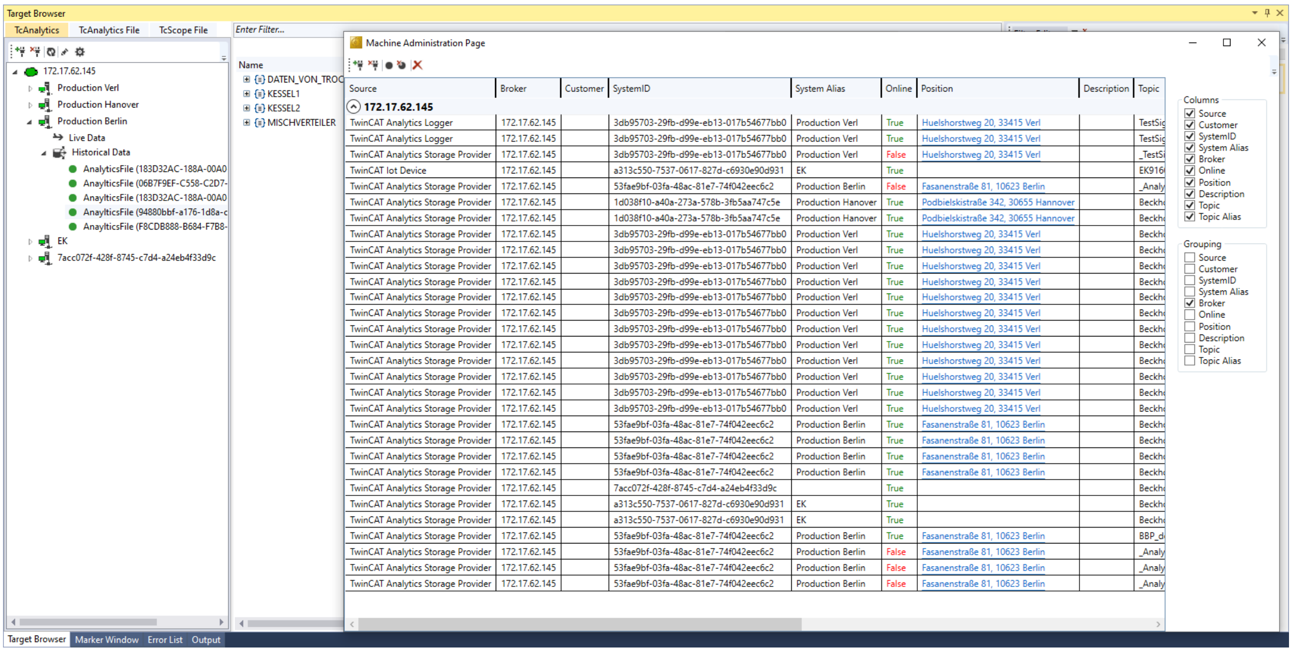
6.8.8 Switching multiple machines in the HMI Dashboard

In TwinCAT Analytics you can use different data streams from several machines, which you can switch in an analysis. This is possible both in the Analytics Workbench via the Virtual Input Source and in the fully generated PLC and HMI. Both live and historical data can be used. For each data stream, you can add a brief description and the location either in the Analytics Logger or in the Machine Administration [▶ 28].

Machine Administration page

If not already done in the Overview, set the metadata of your machine on the Analyse data [▶ 28] page. Open the TwinCAT Target Browser (**TwinCAT > Target Browser > Target Browser**) and click the gear icon. Now you can enter the location of your machine, a short description and the name of your machine. Note that existing data in an Analytics project is not assigned this meta information, since the information is only transferred from the Target Browser during drag and drop. In this case, you can delete the data stream and recreate it.

For each data stream you use in your Analytics configuration, the Runtime deployment [▶ 353] creates a map entry. These map entries are used as input variables for the general map on the start page of your HMI dashboard.



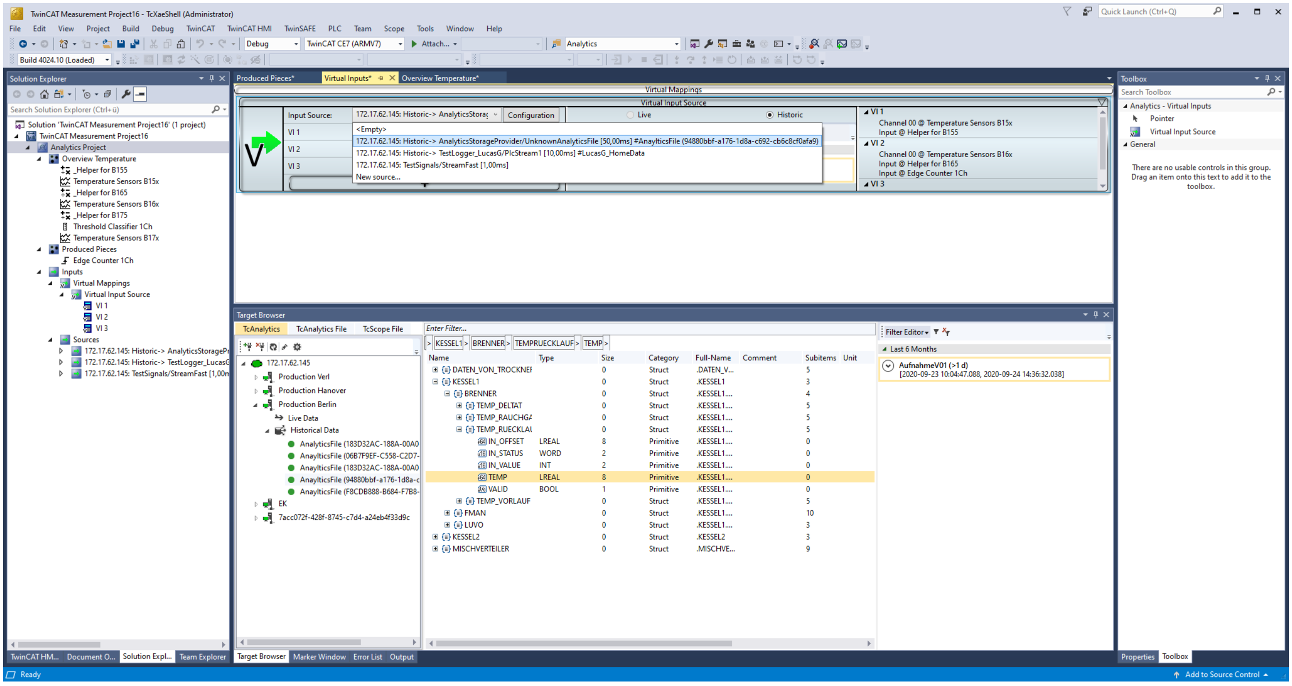
Columns
☑ Source
☑ Customer
☑ SystemID
☑ System Alias
☑ Broker
☑ Online
☑ Position
☑ Description
☑ Topic
☑ Topic Alias

Grouping
☐ Source
☐ Customer
☐ SystemID
☐ System Alias
☑ Broker
☐ Online
☐ Position
☐ Description
☐ Topic
☐ Topic Alias

172.17.62.145

Source	Broker	Customer	SystemID	System Alias	Online	Position	Description	Topic
TwinCAT Analytics Logger	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		TestSig
TwinCAT Analytics Logger	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		TestSig
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	False	Huelshorstweg 20, 33415 Verl		_TestSi
TwinCAT Iot Device	172.17.62.145		a313c550-7537-0617-827d-c6930e90d931	EK	True			EK916
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	False	Fasanenstraße 81, 10623 Berlin		_Analy
TwinCAT Analytics Storage Provider	172.17.62.145		1d038f10-a40a-273a-578b-3fb5aa747c5e	Production Hanover	True	Podbielskistraße 342, 30655 Hannover		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		1d038f10-a40a-273a-578b-3fb5aa747c5e	Production Hanover	True	Podbielskistraße 342, 30655 Hannover		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		3db95703-29fb-d99e-eb13-017b54677bb0	Production Verl	True	Huelshorstweg 20, 33415 Verl		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	True	Fasanenstraße 81, 10623 Berlin		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	True	Fasanenstraße 81, 10623 Berlin		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	True	Fasanenstraße 81, 10623 Berlin		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	True	Fasanenstraße 81, 10623 Berlin		Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		7acc072f-428f-8745-c7d4-a24eb4f33d9c		True			Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		a313c550-7537-0617-827d-c6930e90d931	EK	True			Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		a313c550-7537-0617-827d-c6930e90d931	EK	True			Beckh
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	True	Fasanenstraße 81, 10623 Berlin		BBP_d
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	False	Fasanenstraße 81, 10623 Berlin		_Analy
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	False	Fasanenstraße 81, 10623 Berlin		_Analy
TwinCAT Analytics Storage Provider	172.17.62.145		53fae9bf-03fa-48ac-81e7-74f042eec6c2	Production Berlin	False	Fasanenstraße 81, 10623 Berlin		_Analy

In the Analyse data [▶ 28] the data streams are listed under **Sources**. These are created by using data from various sources from the TwinCAT Target Browser. These sources are listed in the Virtual Input Source. It is possible to switch between the sources. Check that all individual inputs are linked and that none is set to **Empty** by clicking each source once (which corresponds to a switch in the workbench). You can then generate an HMI with PLC.



Input Source: 172.17.62.145: Historic-> AnalyticsStorag | Configuration | ○ Live | ● Historic

- <Empty>
- 172.17.62.145: Historic-> AnalyticsStorageProvider/UnknownAnalyticsFile [50,00ms] #AnayIticsFile (94880bbf-a176-1d8a-c692-cb6c8cf0afa9)
- 172.17.62.145: Historic-> TestLogger_LucasG/PlcStream1 [10,00ms] #LucasG_HomeData
- 172.17.62.145: TestSignals/StreamFast [1,00ms]
- New source...

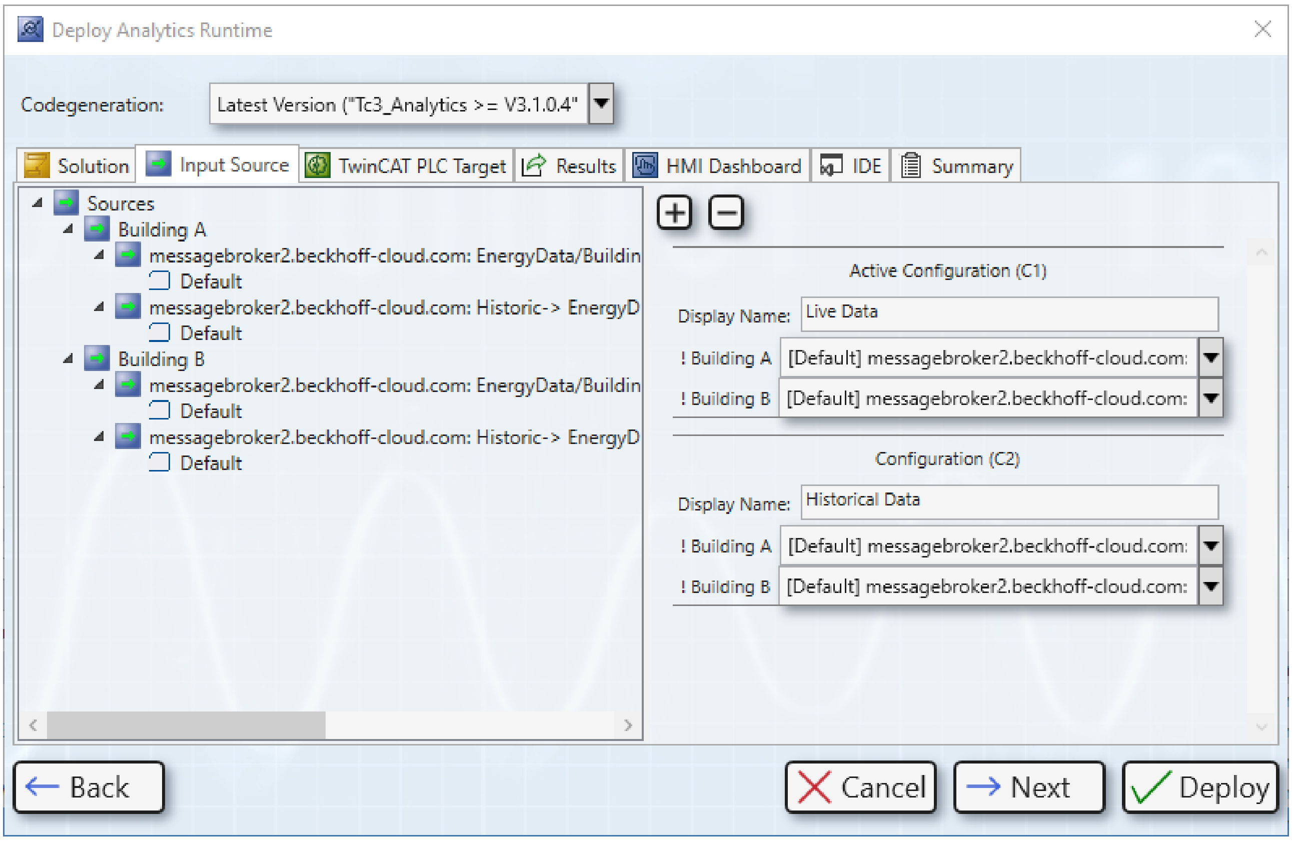
VI 1
Channel 00 @ Temperature Sensors B15x
Input @ Helper for B155

VI 2
Channel 00 @ Temperature Sensors B16x
Input @ Helper for B165
Input @ Edge Counter 1Ch

VI 3

Deploy Wizard

In Runtime deployment [▶ 353], any number of configurations can be created at **Input Source**, which can be switched in the HMI. This makes it possible in the analysis HMI to analyze and switch live as well as historical data from possibly different machines. Each of the listed configurations can be analyzed in parallel. A configuration always has as many Virtual Input Sources as are configured in the Analytics Workbench.



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4"

- Sources
  - Building A
    - messagebroker2.beckhoff-cloud.com: EnergyData/Buildin
      - Default
    - messagebroker2.beckhoff-cloud.com: Historic-> EnergyD
      - Default
  - Building B
    - messagebroker2.beckhoff-cloud.com: EnergyData/Buildin
      - Default
    - messagebroker2.beckhoff-cloud.com: Historic-> EnergyD
      - Default

Active Configuration (C1)

Display Name: Live Data

! Building A [Default] messagebroker2.beckhoff-cloud.com:

! Building B [Default] messagebroker2.beckhoff-cloud.com:

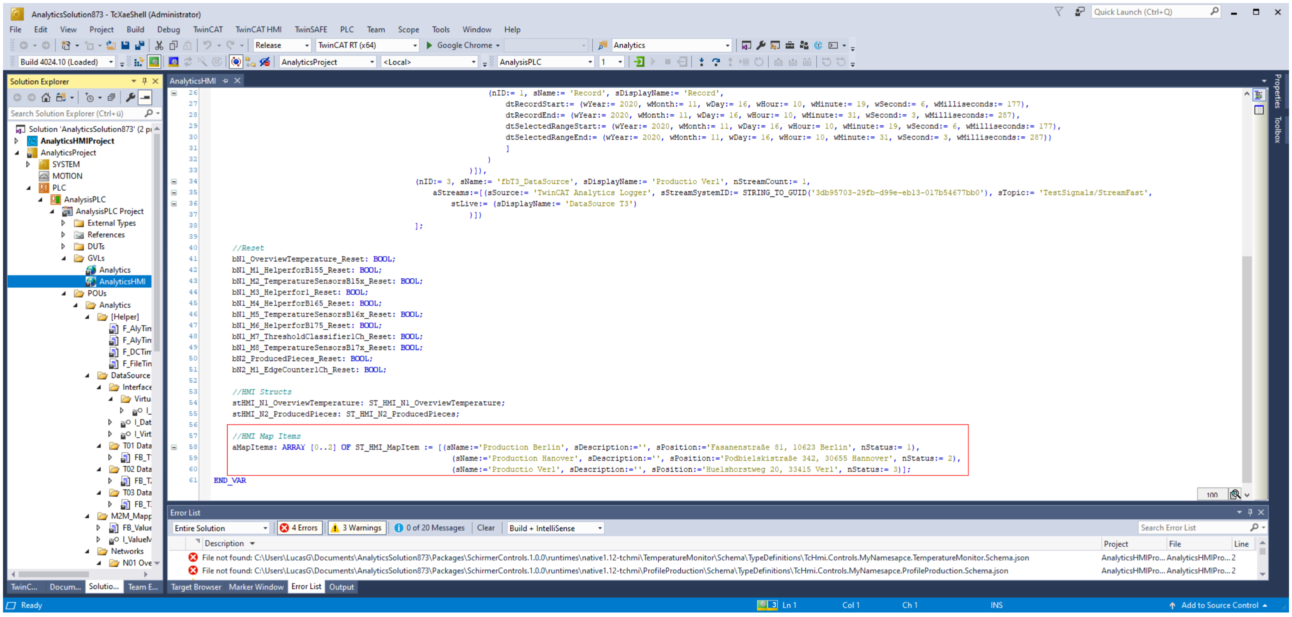
Configuration (C2)

Display Name: Historical Data

! Building A [Default] messagebroker2.beckhoff-cloud.com:

! Building B [Default] messagebroker2.beckhoff-cloud.com:

After a successful HMI and PLC generation, the names, positions, and descriptions of the input sources are entered into the global variable list of the PLC. If something is not right or you want to change a value afterwards, you can do this directly in the PLC.



```
(nID:= 1, sName:= 'Record', sDisplayName:= 'Record',
    dtRecordStart:= (wYear:= 2020, wMonth:= 11, wDay:= 16, wHour:= 10, wMinute:= 19, wSecond:= 6, wMilliseconds:= 177),
    dtRecordEnd:= (wYear:= 2020, wMonth:= 11, wDay:= 16, wHour:= 10, wMinute:= 31, wSecond:= 3, wMilliseconds:= 287),
    dtSelectedRangeStart:= (wYear:= 2020, wMonth:= 11, wDay:= 16, wHour:= 10, wMinute:= 19, wSecond:= 6, wMilliseconds:= 177),
    dtSelectedRangeEnd:= (wYear:= 2020, wMonth:= 11, wDay:= 16, wHour:= 10, wMinute:= 31, wSecond:= 3, wMilliseconds:= 287))
    ]
)
)]),
(nID:= 3, sName:= 'fbT3_DataSource', sDisplayName:= 'Productio Verl', nStreamCount:= 1,
    aStreams:=[(sSource:= 'TwinCAT Analytics Logger', sStreamSystemID:= STRING_TO_GUID('3db95703-29fb-d99e-eb13-017b54677bb0'), sTopic:= 'TestSignals/StreamFast',
        stLive:= (sDisplayName:= 'DataSource T3')
            )])
];

//Reset
bN1_OverviewTemperature_Reset: BOOL;
bN1_M1_HelperforB155_Reset: BOOL;
bN1_M2_TemperatureSensorsB15x_Reset: BOOL;
bN1_M3_Helperfor1_Reset: BOOL;
bN1_M4_HelperforB165_Reset: BOOL;
bN1_M5_TemperatureSensorsB16x_Reset: BOOL;
bN1_M6_HelperforB175_Reset: BOOL;
bN1_M7_ThresholdClassifier1Ch_Reset: BOOL;
bN1_M8_TemperatureSensorsB17x_Reset: BOOL;
bN2_ProducedPieces_Reset: BOOL;
bN2_M1_EdgeCounter1Ch_Reset: BOOL;

//HMI Structs
stHMI_N1_OverviewTemperature: ST_HMI_N1_OverviewTemperature;
stHMI_N2_ProducedPieces: ST_HMI_N2_ProducedPieces;

//HMI Map Items
aMapItems: ARRAY [0..2] OF ST_HMI_MapItem := [(sName:='Production Berlin', sDescription:='', sPosition:='Fasanenstraße 81, 10623 Berlin', nStatus:= 1),
                                              (sName:='Production Hanover', sDescription:='', sPosition:='Podbielskistraße 342, 30655 Hannover', nStatus:= 2),
                                              (sName:='Productio Verl', sDescription:='', sPosition:='Huelshorstweg 20, 33415 Verl', nStatus:= 3)];
END_VAR
```

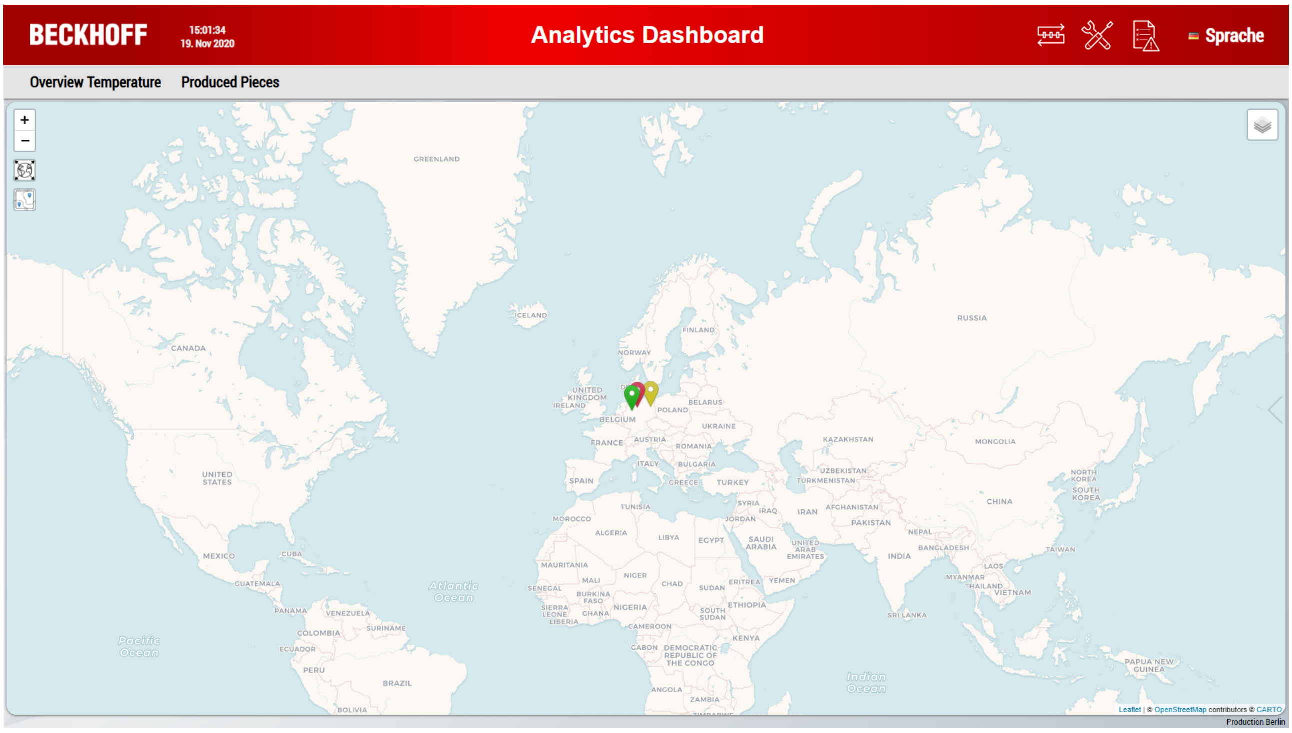
Error List

Entire Solution | 4 Errors | 3 Warnings | 0 of 20 Messages | Clear | Build + IntelliSense | Search Error List

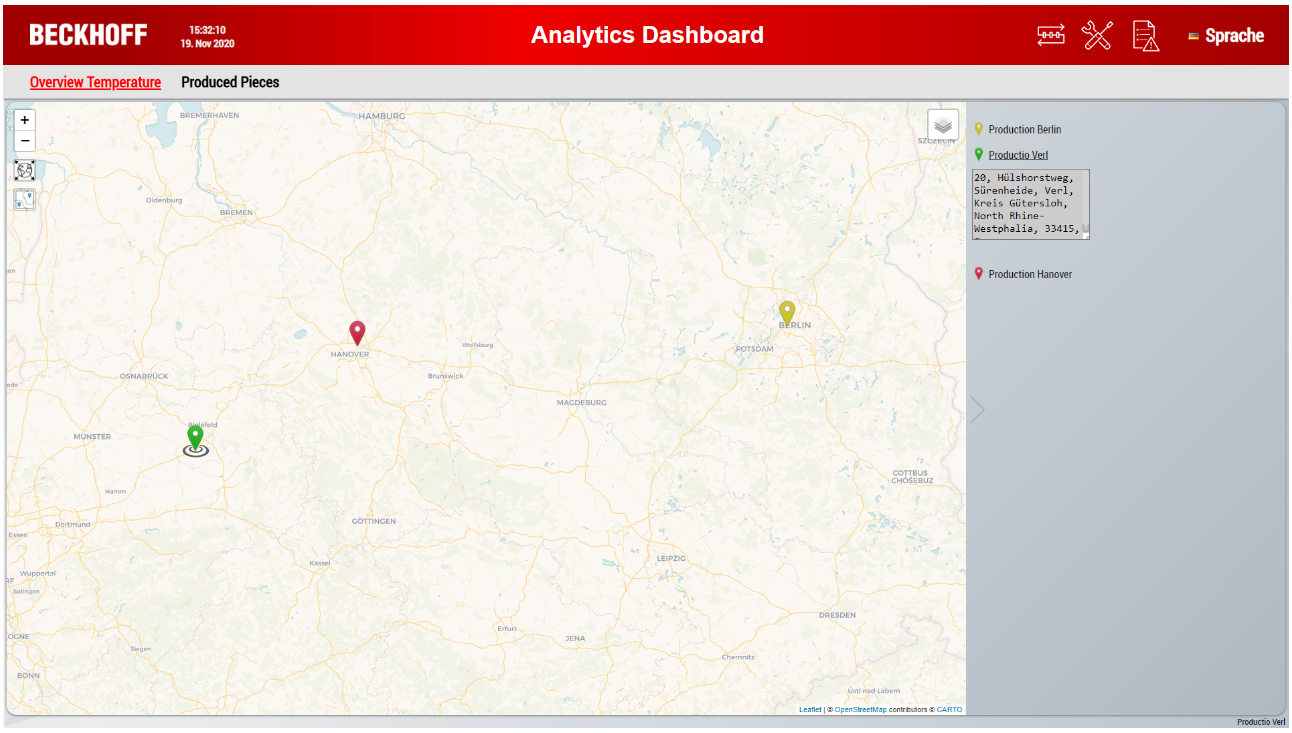
Description	Project	File	Line
File not found: C:\Users\LucasG\Documents\AnalyticsSolution873\Packages\SchirmerControls.1.0.0\runtimes\native1.12-tchmi\TemperatureMonitor\Schema\TypeDefinitions\TcHmi.Controls.MyNamesapce.TemperatureMonitor.Schema.json	AnalyticsHMIPro...	AnalyticsHMIPro...	2
File not found: C:\Users\LucasG\Documents\AnalyticsSolution873\Packages\SchirmerControls.1.0.0\runtimes\native1.12-tchmi\ProfileProduction\Schema\TypeDefinitions\TcHmi.Controls.MyNamesapce.ProfileProduction.Schema.json	AnalyticsHMIPro...	AnalyticsHMIPro...	2

Map with machines

If you selected to create a start page, the *Map Control* will be created on the homepage of your HMI Dashboard. The map shows all machine locations and lists the names of the machines (system alias) in the legend. The legend can be opened and closed using the arrow on the right. The icon color indicates the current machine status: green = OK, yellow = Warning, red = Alarm.

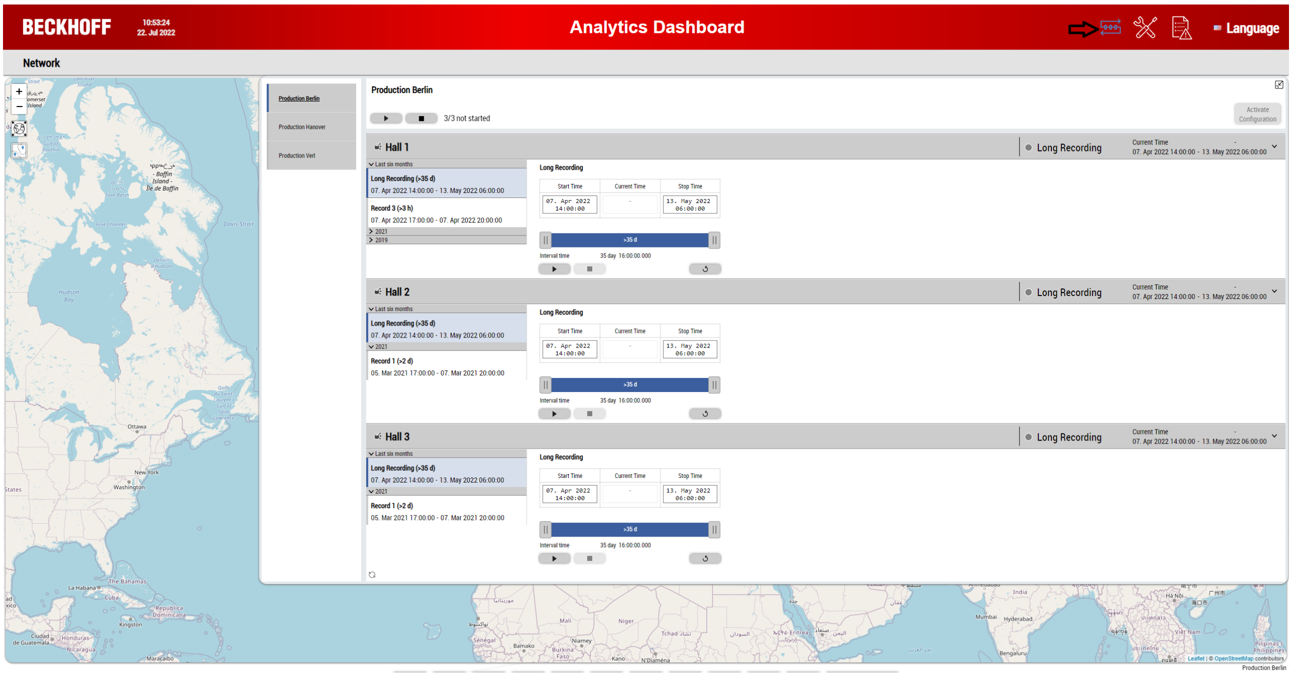


You can zoom in on a single icon by double-clicking it and zoom out to an overview of all machines using the icons on the left. It is also possible to click on an icon on the map. The corresponding legend entry is automatically highlighted. This also works the other way around, i.e. when you click on a legend entry, the corresponding icon is automatically highlighted.



Switching machine data

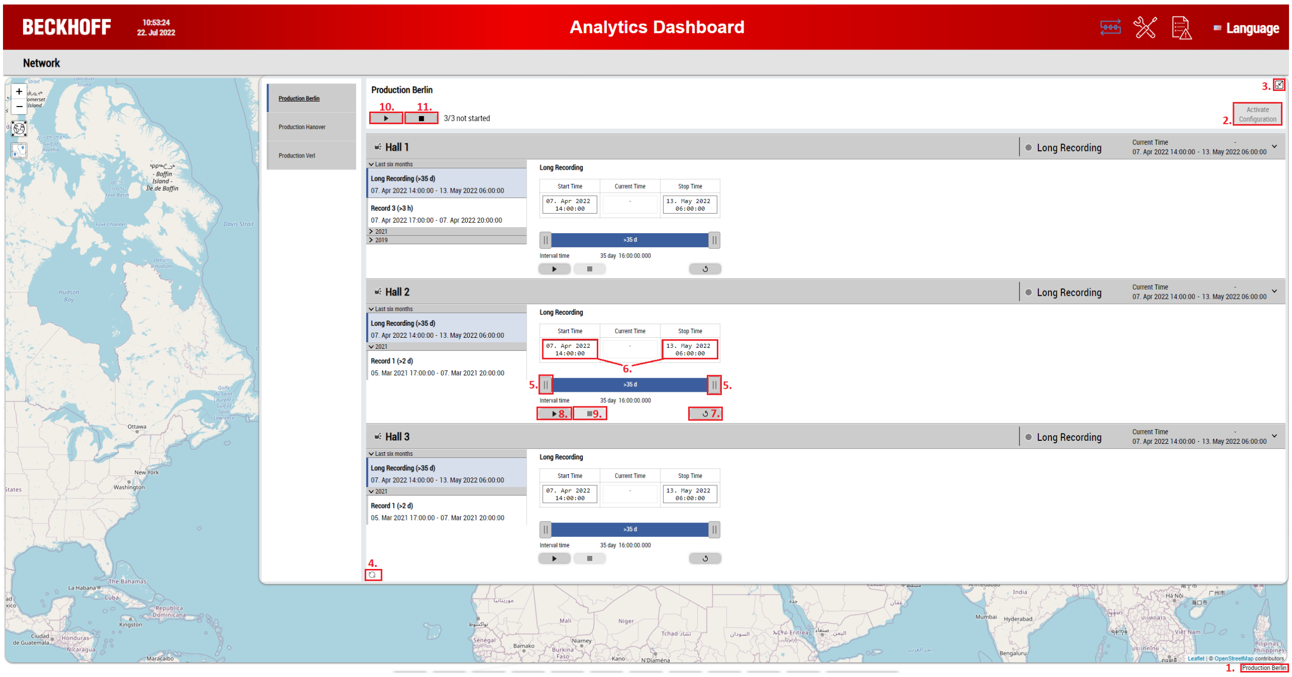
In addition to the map with the individual machine configurations, the analyses for the individual configurations can be switched. In the PLC all analyses run in separate tasks, which means that all analysis data is available at the same time. In the generated HMI, the various machines can be switched over in a specially developed control. The control can be opened and closed via the icon highlighted in the image below (black arrow).



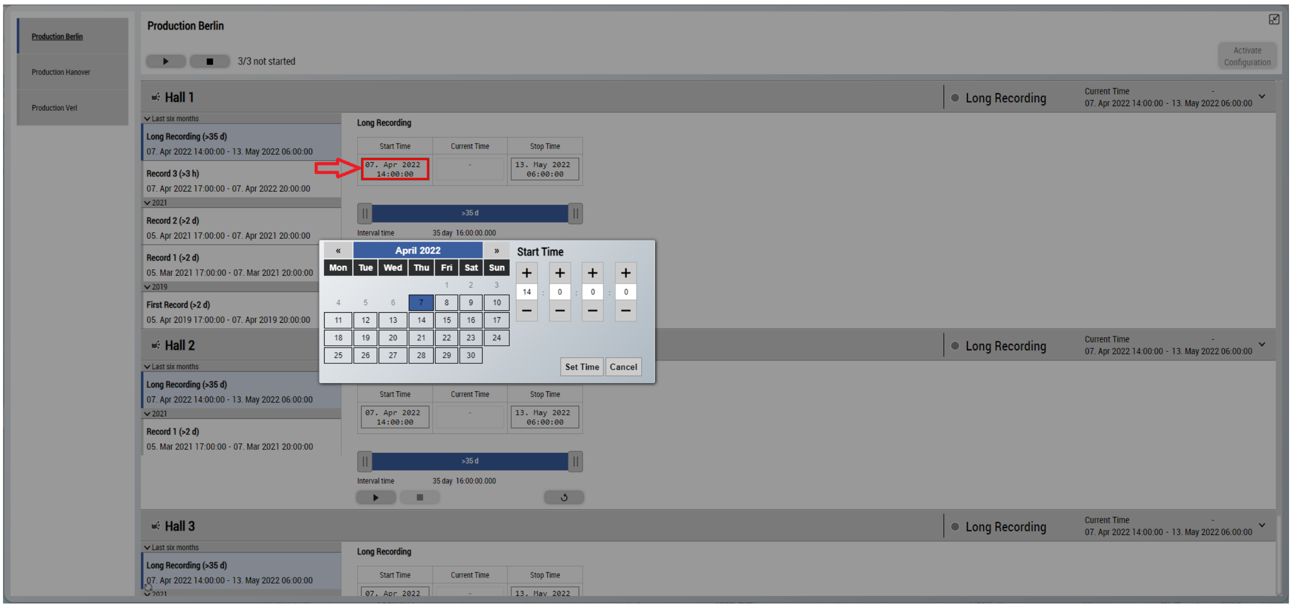
The configurations that were previously shown on the map with the locations can be selected in the control. A configuration can consist of live and historical data, whereby only historical data is configured for the **Production Berlin**. Historical data can be analyzed equivalently as in Working with Historical Data [▶ 351] in the HMI.

When clicking on another machine, this configuration is only displayed in the control. To activate, click **Activate Configuration**.
The control is explained below.

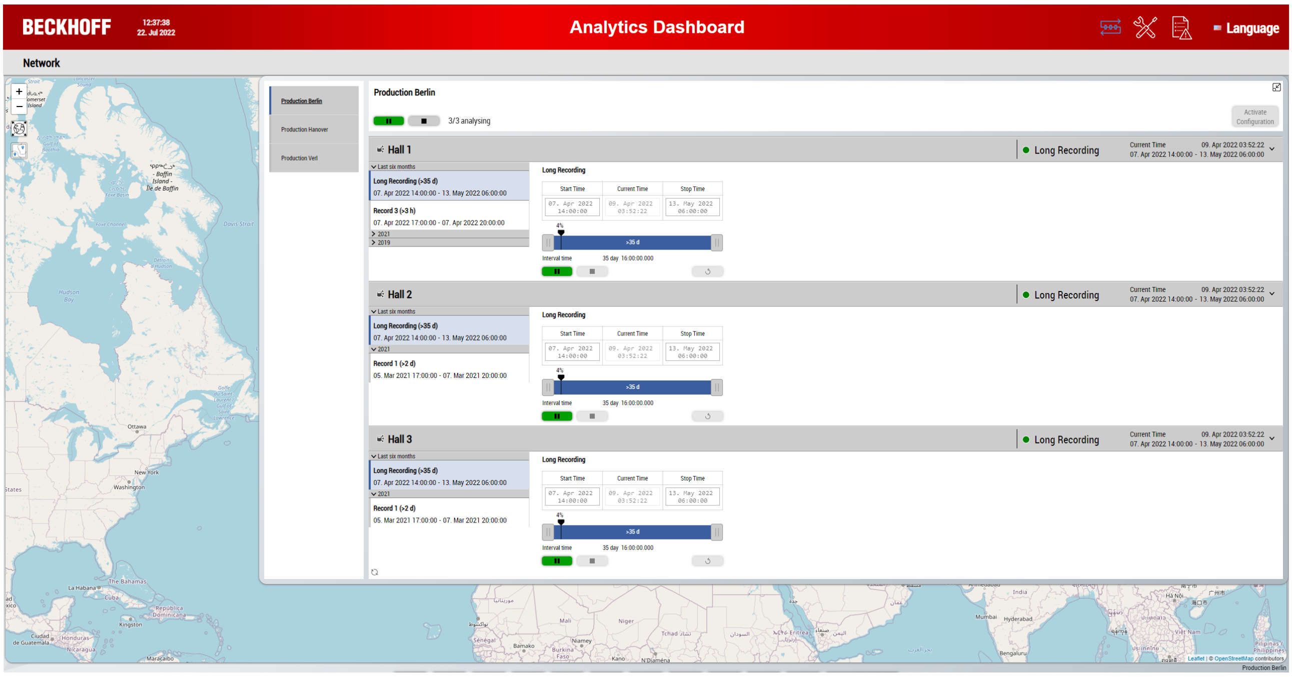
1. The currently active configuration.
2. Button to activate the selected configuration.
3. Selection window to minimize (and automatically activate the selected configuration).
4. Reloading the record list. This allows data recorded subsequently to be dynamically loaded and analyzed in the dashboard.
5. Selection of the start and end times via a slider.
6. Selection of the start and end date via a selection window with calendar.
7. Resetting the start and end times.
8. Starting the analysis of the selected **Input Source**.
9. Cancel the analysis of the selected **Input Source**.
10. Starting the entire analysis
11. Cancel the entire analysis



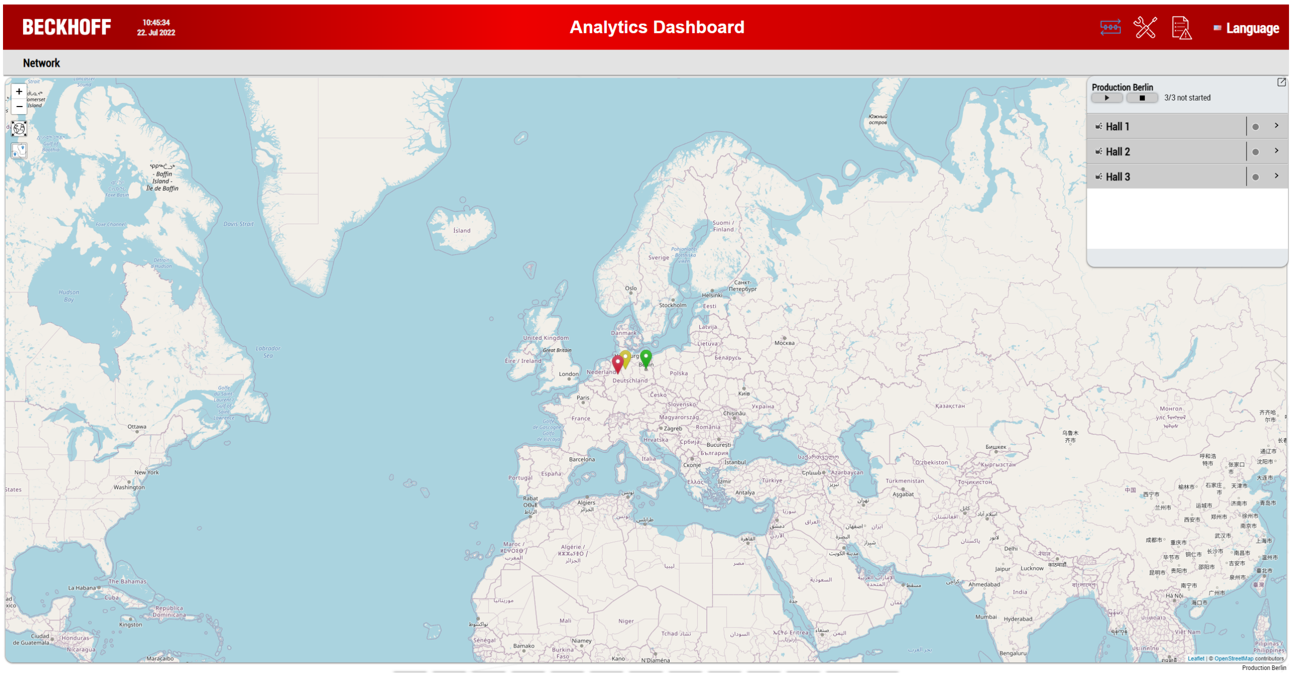
The slider is ideal for quick tests and approximate setting of the start and end times. For an exact setting down to the millisecond, click on the **text field**. This opens a selection window with calendar for selecting the date and setting the exact time.

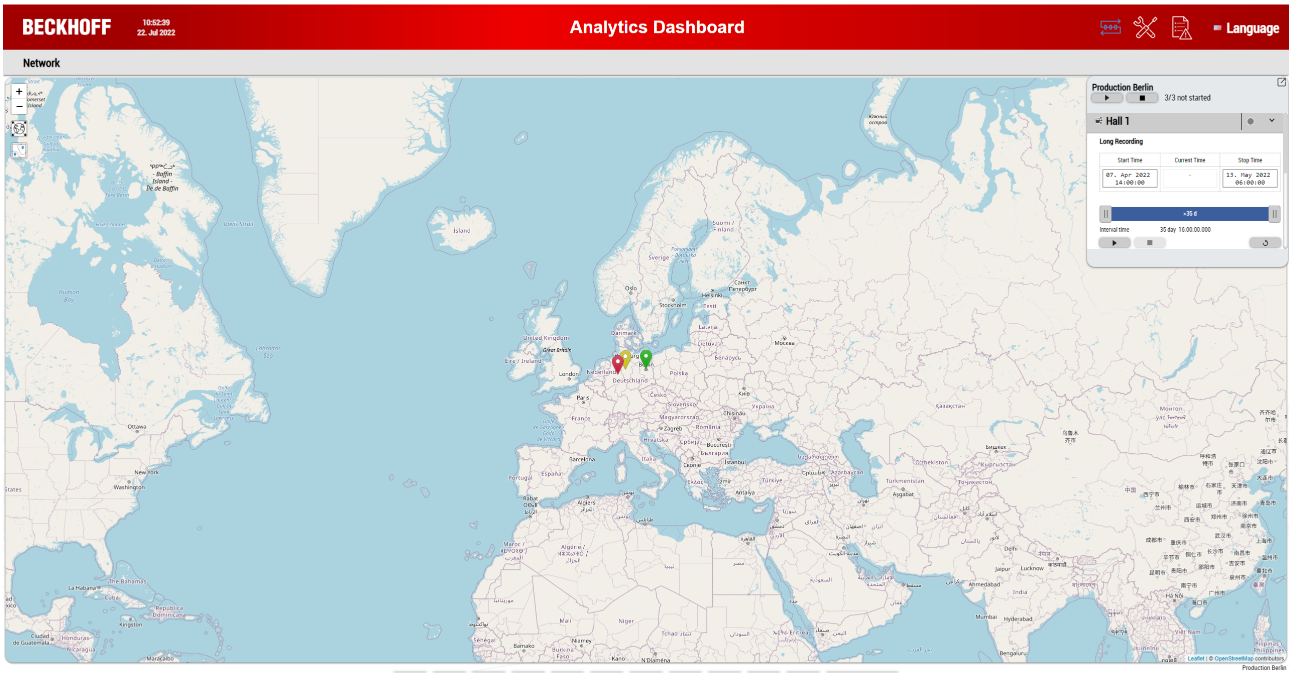


As soon as you click the button **Start**, the analysis is triggered with all input sources in the PLC. A loading symbol appears for the first time until the analysis starts. It turns green when the first data appears. Likewise, a marker shows the current time of the historical analysis in the slider. The process speed depends on the bandwidth, the number of data and the recorded cycle time. It is possible to analyze several historical recordings at the same time by simply switching machines. Internally, all analyses continue in parallel. This makes it easy to switch between analyses in the dashboard.



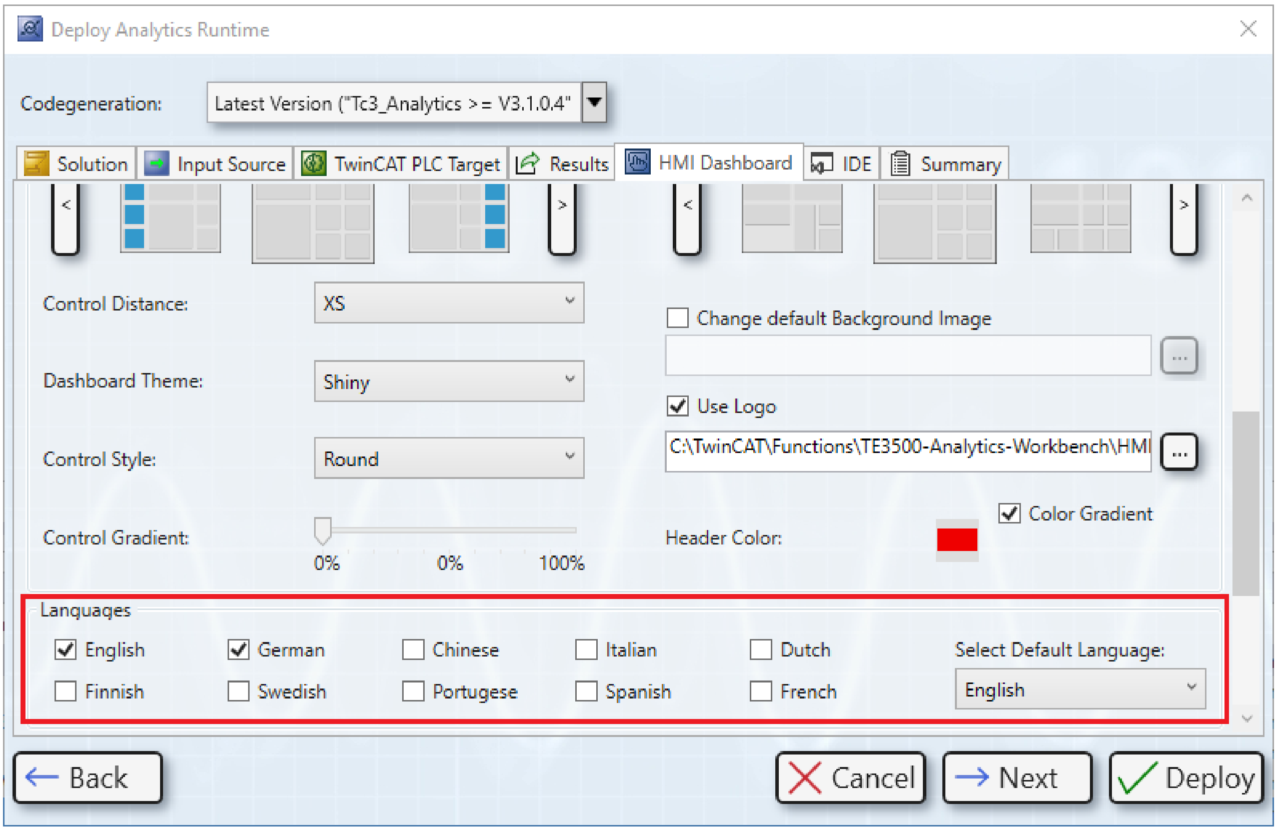
The button **Minimize** in the upper right corner hides the selection of configurations and records. This allows you to work with the records from the input sources and view the data at the same time. You can hide the window completely by clicking the blue icon at the top. Only the configuration name is always in the bottom right corner of the dashboard.





6.8.9 Integration of a language switch

In the Deploy wizard configuration window, up to 8 languages can be selected for language switching in the HMI area.



Codegeneration: Latest Version ("Tc3_Analytics >= V3.1.0.4")

Control Distance: XS

Dashboard Theme: Shiny

Control Style: Round

Control Gradient: 0% 0% 100%

□ Change default Background Image

☑ Use Logo

C:\TwinCAT\Functions\TE3500-Analytics-Workbench\HMI

Header Color: 🟥 ☑ Color Gradient

Languages

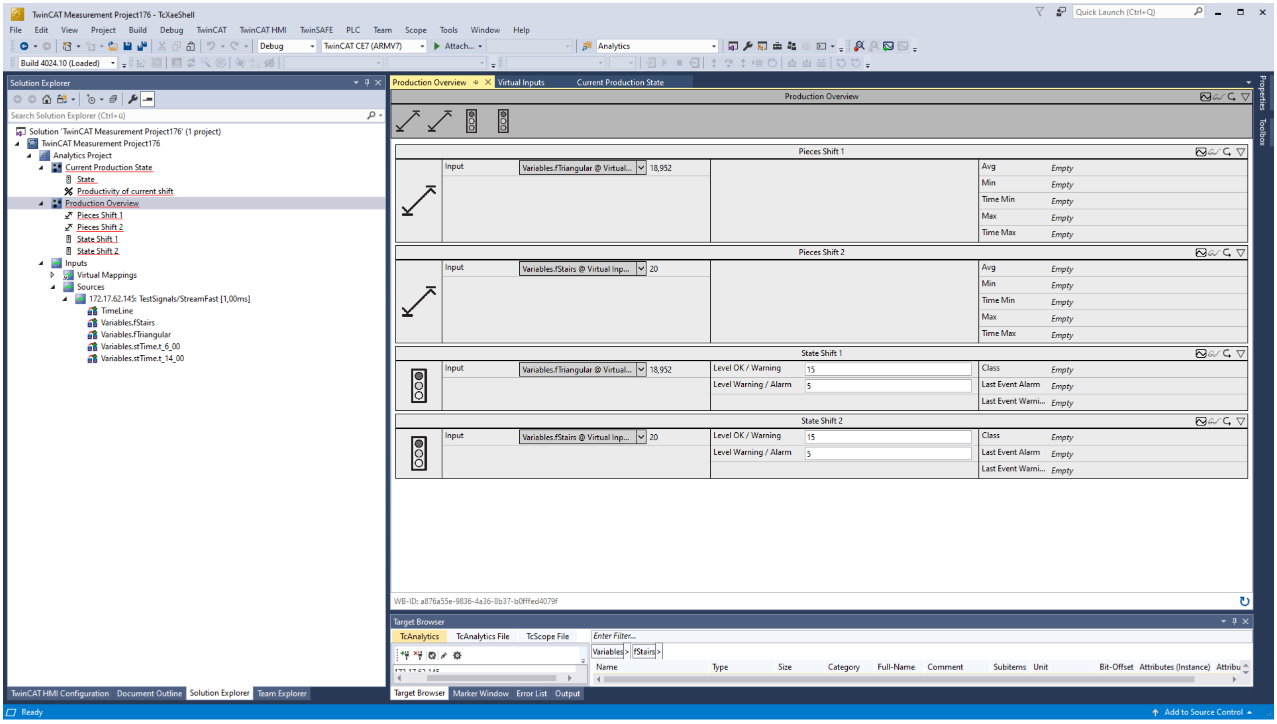
☑ English ☑ German □ Chinese □ Italian □ Dutch

□ Finnish □ Swedish □ Portugese □ Spanish □ French

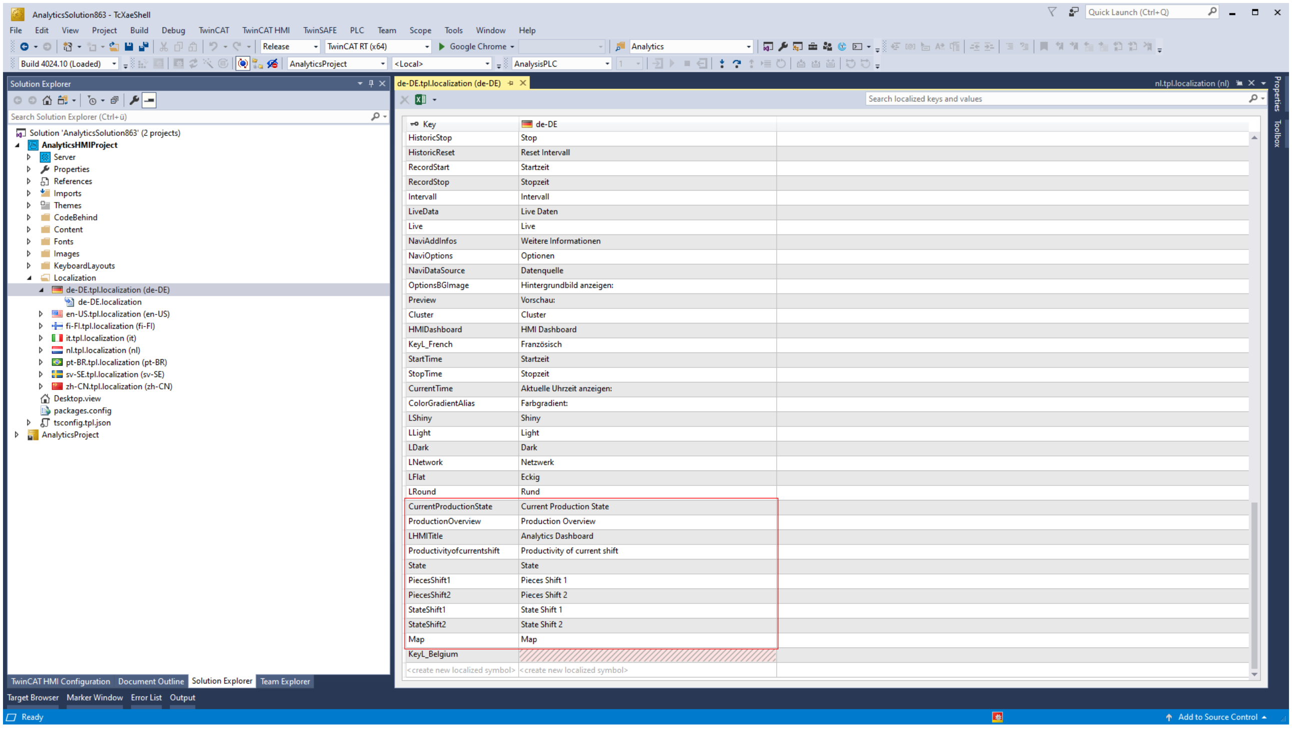
Select Default Language: English

If you use the user administration, the languages of the created users are already selected. These cannot be deselected.

The texts in the supplied Analytics Controls and all other texts can be switched automatically. Only your network and module names have to be translated, if you want them to be included in language switching. In the following screenshot, the main affected names are marked in the **Solution Explorer**.

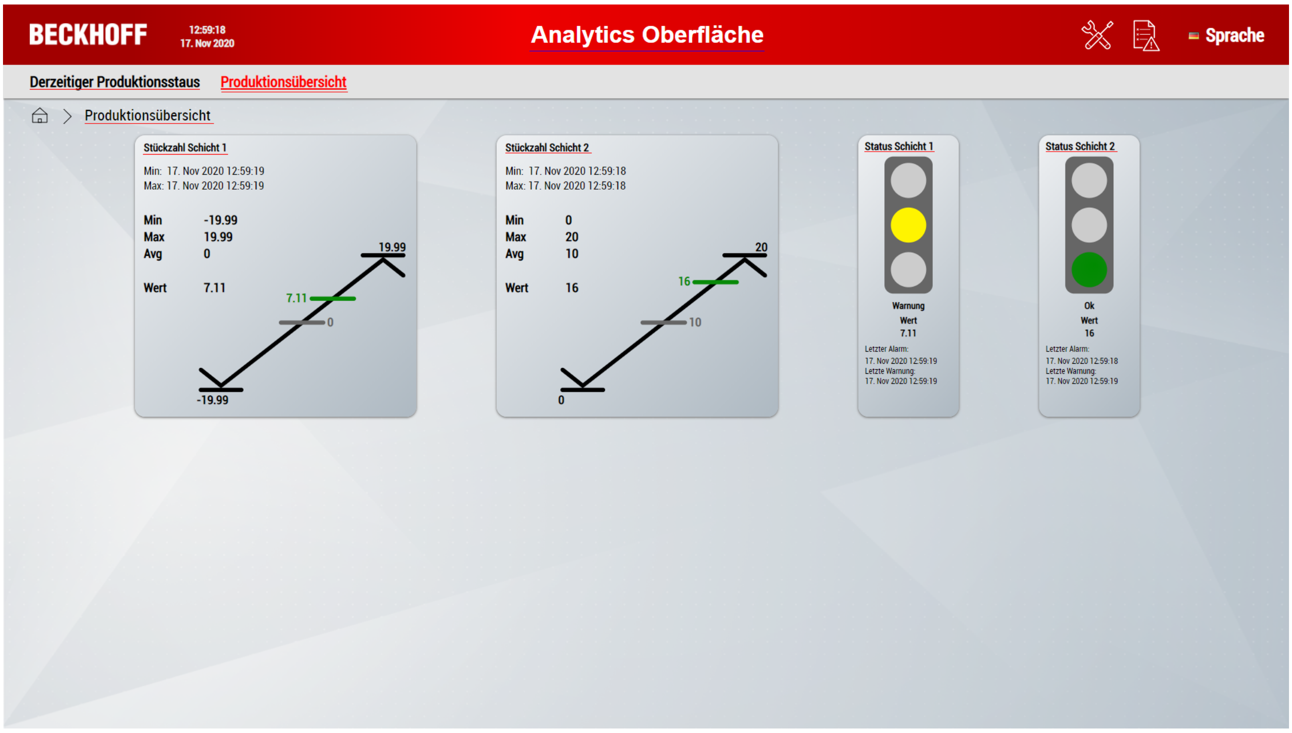


In the generated TwinCAT HMI Engineering project this is easily possible, since the translation entries are already prepared. The texts for the respective languages are stored in the **Localization** files. The names of your networks and modules are automatically entered there and only have to be translated. To do this, open all the files of the languages for which you need a translation. The following screenshot lists the entries that need to be translated for German. The number of entries to be processed varies depending on the complexity of the Analytics project.

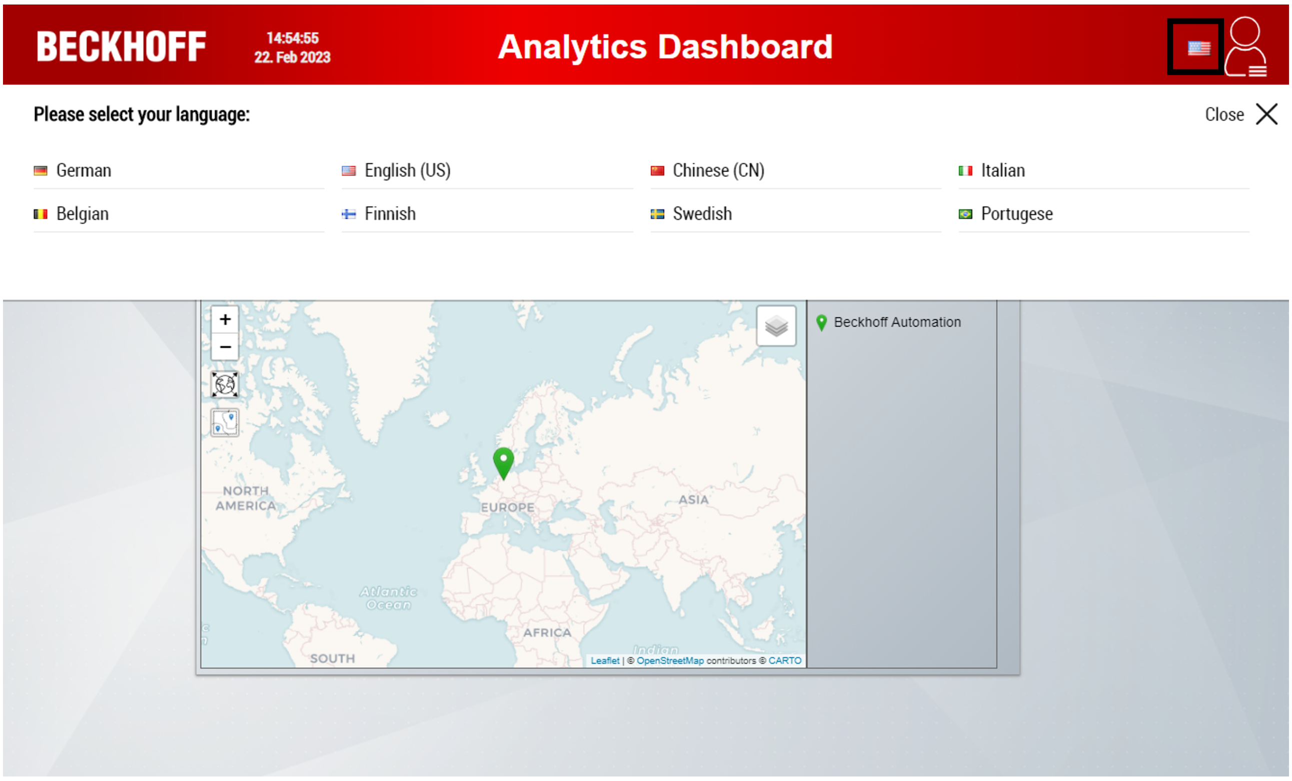

AnalyticsSolution863 - TcXaeShell
Solution Explorer
Solution 'AnalyticsSolution863' (2 projects)
AnalyticsHMIProject
Localization
de-DE.tpl.localization (de-DE)
de-DE.localization
en-US.tpl.localization (en-US)
fi-FI.tpl.localization (fi-FI)
it.tpl.localization (it)
nl.tpl.localization (nl)
pt-BR.tpl.localization (pt-BR)
sv-SE.tpl.localization (sv-SE)
zh-CN.tpl.localization (zh-CN)
AnalyticsProject
de-DE.tpl.localization (de-DE)
Google Chrome

Key	de-DE
HistoricStop	Stop
HistoricReset	Reset Intervall
RecordStart	Startzeit
RecordStop	Stopzeit
Intervall	Intervall
LiveData	Live Daten
Live	Live
NaviAddInfos	Weitere Informationen
NaviOptions	Optionen
NaviDataSource	Datenquelle
OptionsBGImage	Hintergrundbild anzeigen:
Preview	Vorschau:
Cluster	Cluster
HMIDashboard	HMI Dashboard
KeyL_French	Französisch
StartTime	Startzeit
StopTime	Stopzeit
CurrentTime	Aktuelle Uhrzeit anzeigen:
ColorGradientAlias	Farbgradient:
LShiny	Shiny
LLight	Light
LDark	Dark
LNetwork	Netzwerk
LFlat	Eckig
LRound	Rund
CurrentProductionState	Current Production State
ProductionOverview	Production Overview
LHMITitle	Analytics Dashboard
Productivityofcurrentshift	Productivity of current shift
State	State
PiecesShift1	Pieces Shift 1
PiecesShift2	Pieces Shift 2
StateShift1	State Shift 1
StateShift2	State Shift 2
Map	Map
KeyL_Belgium	▨
<create new localized symbol>	<create new localized symbol>

Once completed, you can open the dashboard by clicking the **Google Chrome** button (or the name of your default browser). Simply reloading in the browser is not sufficient, because the project has to be rebuilt internally. In the following image the texts which are now also switched with the language switch are highlighted.



The language can be changed via the flag in the upper right corner.



The language can be set and saved user-specifically from Dashboard version 2.0.

6.8.10 Using the interactive functions in standard TwinCAT HMI projects

Both the Interactive Dashboard and the Interactive Chart function can be integrated into any TwinCAT HMI project independently of the One Click Dashboard. This requires a valid TwinCAT Analytics Runtime or Runtime Base license (TF3550 or TF3551) on the target system.

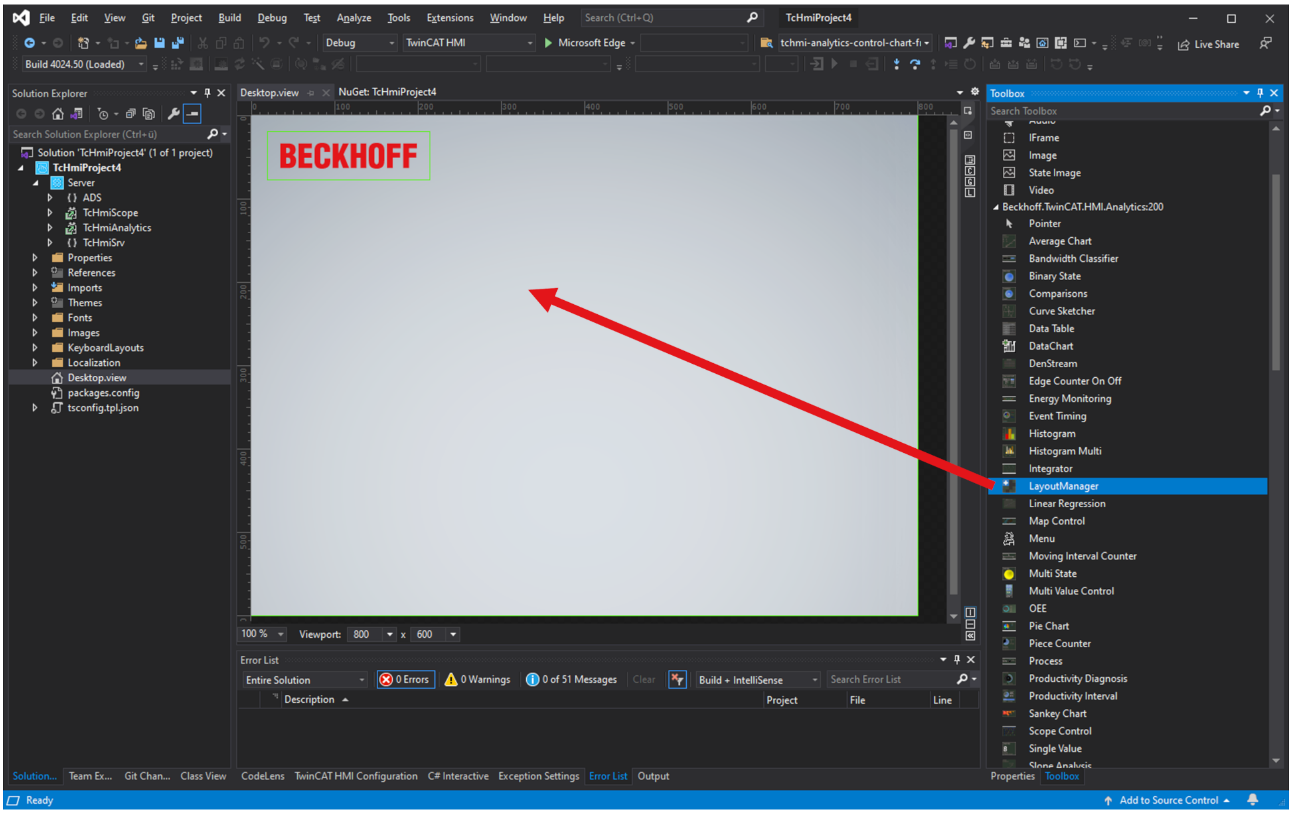
Installation of the NuGet packages

The required components (Analytics Framework Controls and the Analytics Server Extension) can be integrated into the HMI project via NuGet packages. To do this, the packages "Beckhoff.TwinCAT.HMI.Analytics.ControlsExtension" and "Beckhoff.TwinCAT.HMI.Analytics.Controls" must be installed in the NuGet Package Manager.

After installation, the "TcHmiAnalytics" server extension is located in the HMI project under the server node. If this does not start automatically, it must be started manually by right-clicking on the extension.

Adding the Layout Manager control

Dynamic and user-specific layouts can be created using the Layout Manager control. The Layout Manager control can be inserted into the HMI project via the toolbox using drag and drop. It is located in the "Beckhoff.TwinCAT.HMI.Analytics" area. The Layout Manager is similar to the Container Control. The difference is that the controls can be repositioned in the browser.



Setting the properties of the Layout Manager control

For the Layout Manager control to function optimally, the following properties should be set:

Name	Description
Identifier	Once layouts have been saved with the server extension, the identifier should no longer be changed. Otherwise, saved layouts can no longer be assigned.
WidthMode	Selection “Value” should not be changed.
HeightMode	Selection “Value” should not be changed.

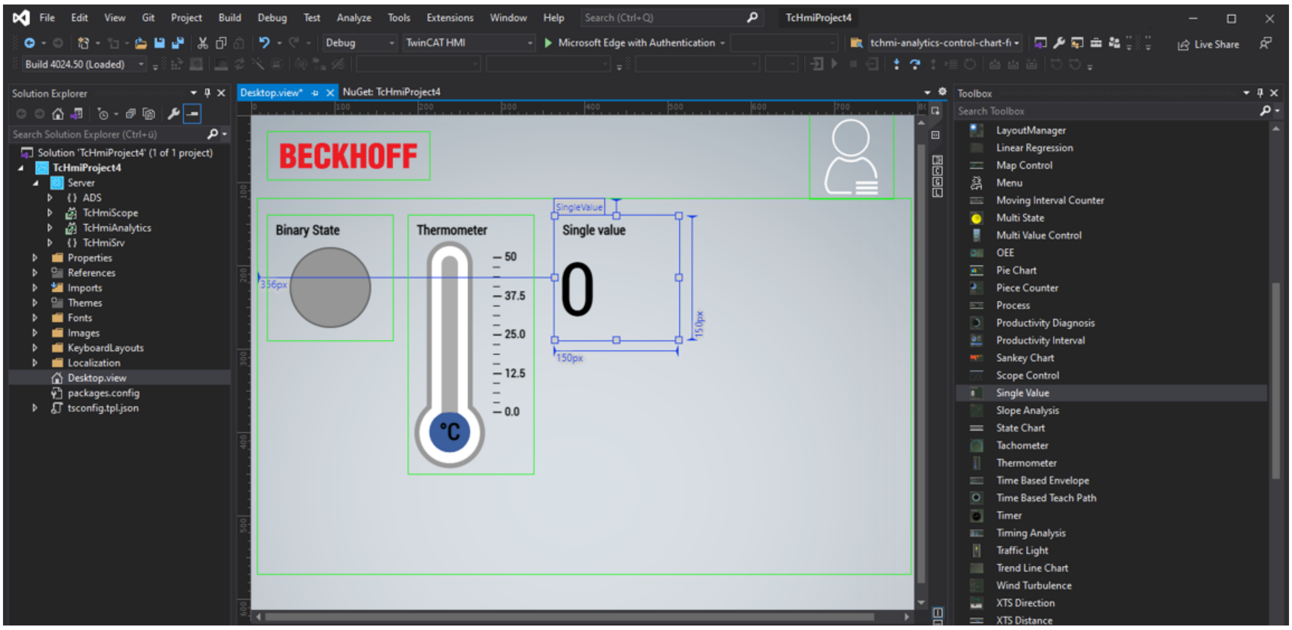
The following parameters can be set optionally:

Name	Description
DarkMode	Selects whether to display the control in DarkMode.
MobileViewWidth	Pixel width from which to switch to Mobile View.
Number of columns	Number of columns that the layout should include in the normal view.
Grid Gap	Percentage distance between the controls that are positioned on the Layout Manager.
UseCustomRowHeight	This parameter can be used to activate the fixed row height, which can be set using the RowHeight parameter. Otherwise, a dynamically scaled height is used.
RowHeight	This parameter can be used to specify the row height in pixels.
LockControls	This parameter can be used to prevent interaction with the existing controls in the layout in Edit mode.
IsAnalyticsUse	This property should only be activated if the Layout Manager control is used in an Analytics-One-Click-Dashboard. This positions the menu of the Layout Manager control appropriately on the layout.
MenuControlSymbol	A control of the "Menu" type (also from the "Beckhoff.TwinCAT.HMI.Analytics" area) can optionally be linked here. This should always be visible on the dashboard and placed on the "Desktop.view". The Edit mode can also be opened and closed via the Menu control.

Adding controls

Any controls can then be added to the Layout Manager control. These can be moved as required in HMI engineering.

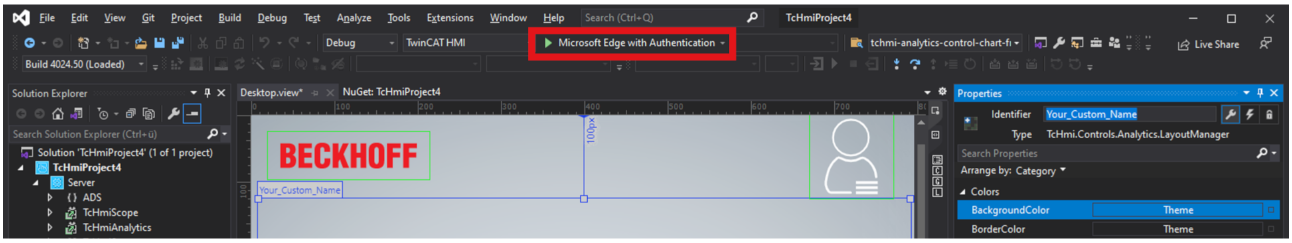
If a layout has been saved for a user using the interactive function, the items are loaded from the saved layout. If further controls are added at a later date, they are sorted into a free space within the layout. Removed controls, on the other hand, disappear directly from the layout.



Interactive chart controls can also be added in the web browser.

Building the project

To be able to use the functions of the Interactive Dashboard and Interactive Charts, the project must be built with user authentication.



Identifier Your_Custom_Name
Type TcHmi.Controls.Analytics.LayoutManager
BackgroundColor Theme
BorderColor Theme

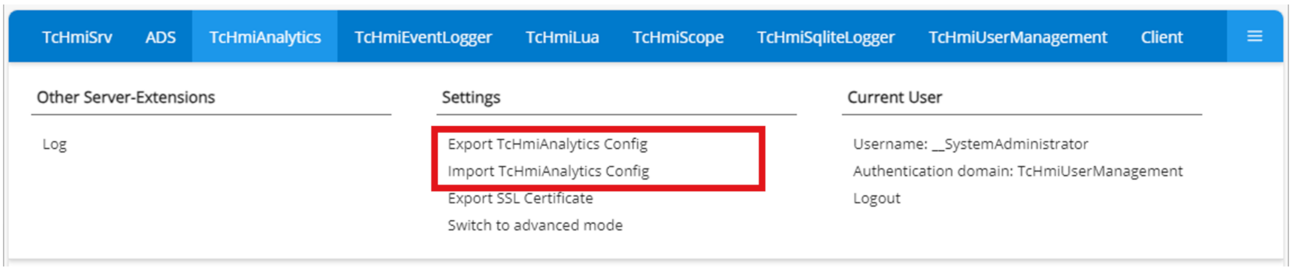
Making user-specific changes

The interactive functions can then be used in the web browser. Information on the use of the Interactive Dashboard [▶ 462] and the Interactive Chart [▶ 468] can be found on the following pages.

Exporting and importing layout and chart configurations

The configurations of layouts and charts saved in the "TcHmiAnalytics" server extension can be exported and imported. This is useful if the configurations are to be transferred from a remote project (e.g. running on an HMI server) to a local project. Alternatively, the configuration of a remote project can also be updated.

The configurations can be exported and imported into the server extension via the Config page of the respective HMI project.


TcHmiSrv ADS TcHmiAnalytics TcHmiEventLogger TcHmiLua TcHmiScope TcHmiSqliteLogger TcHmiUserManagement Client ≡

Other Server-Extensions

Log

Settings

Export TcHmiAnalytics Config
Import TcHmiAnalytics Config
Export SSL Certificate
Switch to advanced mode

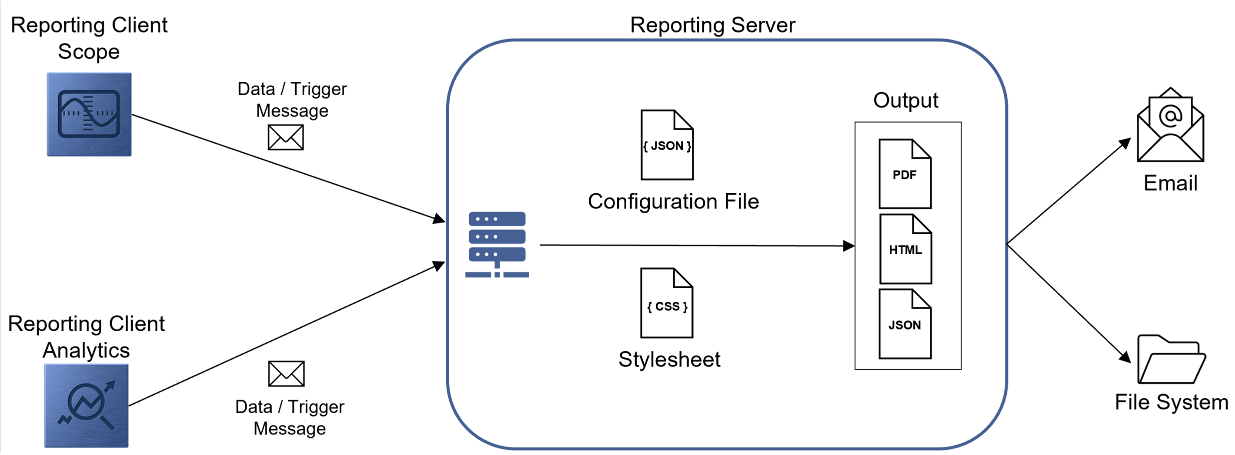
Current User

Username: __SystemAdministrator
Authentication domain: TcHmiUserManagement
Logout

6.9 Reporting Integration

6.9.1 Basic concept

The following figure shows the basic concept of TwinCAT Reporting from the Reporting Clients to the distribution of the generated Reports.



Reporting Clients

Currently, a Reporting Client is integrated in the TwinCAT Scope View as well as in the TwinCAT Analytics Workbench or in the TwinCAT Analytics Service Tool. These tools provide the ability to implement 24/7 reporting and on-demand reporting.

- Scope Reporting

- Analytics Reporting [▶ 505]

24/7 reporting is also possible via the PLC. The function blocks have the same functionality as the algorithms from TwinCAT Analytics.

- PLC Reporting

The Reporting Clients communicate with the TwinCAT Reporting Server. The Reporting Clients can send two types of messages. On the one hand, data messages can be sent with the associated data. On the other hand, so-called trigger messages can be communicated. This type of message triggers the generation of a report.

TwinCAT Reporting Server

The TwinCAT Reporting Server [▶ 513] has a variety of tasks. The server receives the messages from the Reporting Clients and stores the data in a DataStore. A report can be defined in an associated configuration file. The assignment of a data object to a report is established via a Report Name and a Data Key. The design of the report in can be customized in a stylesheet. As soon as a trigger message is received, a report with the associated data is generated in the configured formats and distributed accordingly.

Glossary

Term	Explanation
Report	A report is a document available in PDF, HTML and JSON formats. It contains the information and images.
Configuration file	The configuration file describes the structure of a report. It also offers the possibility to integrate static information into a report by default. Furthermore, settings can be made in the Reporting Server for the corresponding report.
Reporting Server	The Reporting Server manages the configuration files, receives information and creates individual reports from this information.
Reporting Client	A Reporting Client can be used to send data to the Reporting Server or to trigger the creation of a report.
Data message	A Reporting Client can send information to the Reporting Server via a data message. The contained data is cached in the Reporting Server.
Trigger message	A Reporting Client can send a trigger message to the Reporting Server, triggering the generation of a report. For this purpose, the Reporting Server uses all data received up to this point and belonging to the corresponding report. After generation, the data is deleted.
Report Name	The Report Name is used to identify a report. The Report Name is derived from the name of the configuration file without the extension (e.g.: configuration file: Beckhoff Report Template.json; Report Name: Beckhoff Report Template)
Data Key	A Data Key identifies a data object of a report. It must be unique within a report and its configuration file. This can ensure the assignment and sorting of data objects in the report.
24/7 Reporting	24/7 Reporting provides the ability to collect data and automatically generate reports.
On-demand reporting	On-demand reporting provides the ability to generate custom reports on demand.

6.9.2 Analytics Reporting

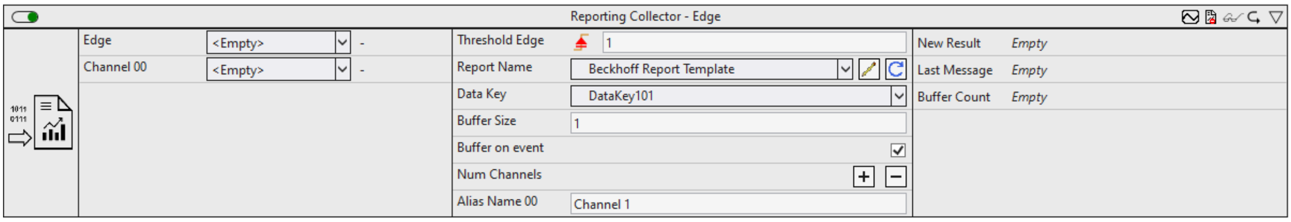
6.9.2.1 24/7 Reporting

24/7 reporting can be implemented in TwinCAT Analytics using the algorithms in the *Reporting* category. The reporting collectors collect the data and send it to the reporting server. The reporting triggers trigger the creation of a report.

6.9.2.1.1 Reporting Collector

The Reporting Collectors collect data and send it to the Reporting Server in a data message after an event.

6.9.2.1.1.1 Reporting Collector Edge



Reporting Collector - Edge

Edge	<Empty>	Threshold Edge	1	New Result	Empty
Channel 00	<Empty>	Report Name	Beckhoff Report Template	Last Message	Empty
		Data Key	DataKey101	Buffer Count	Empty
		Buffer Size	1		
		Buffer on event	☑		
		Num Channels	+ −		
		Alias Name 00	Channel 1		

The Reporting Collector Edge collects the data from the input channels and sends the data to the Reporting Server after an event or after the buffer is filled, depending on the configuration. An event is triggered when the signal of the input channel passes the configured edge at a certain threshold.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

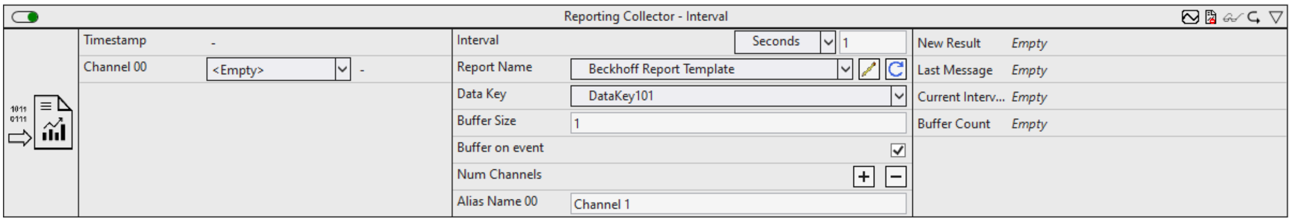
Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.
- **Data Key:** the data key should be unique within a report. The data object can be uniquely assigned to the report via the data key.
- **Buffer Size:** specifies the size of the buffered data. If the buffer size is equal to one, a key-value structure is built. If the buffer size is greater than one, the data is presented in a table.
- **Buffer on event:** specifies how the data is to be collected and buffered.
  If the parameter is True, the data of the inputs is buffered at each edge signal at the input. As soon as the Buffer Count output has the same value as the Buffer Size parameter, the data is sent to the TcReportingServer.
  If the parameter is False, the data of the inputs are buffered in the buffer at each cycle. Once the buffer size is reached, the new data replaces the previous data. If there is an edge signal at the input, the data is sent to the TcReportingServer.
- **Num Channels:** the number of input channels from which the data will be collected. And the number of parameters to allow a description of the input channels.
- **Alias Name:** serves as a description of the input channel. BufferSize = 1: the alias name serves as a key in the key-value structure. BufferSize > 1: the alias name serves as the heading of the table column.
- **Include Timestamps (optional):** inserts a column with the timestamps**.**

Output values

- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Buffer Count:** specifies the number of elements in the buffer.

6.9.2.1.1.2 Reporting Collector Interval



Reporting Collector - Interval

Timestamp	-	Interval	Seconds 1	New Result	Empty
Channel 00	<Empty>	Report Name	Beckhoff Report Template	Last Message	Empty
		Data Key	DataKey101	Current Interv...	Empty
		Buffer Size	1	Buffer Count	Empty
		Buffer on event	☑		
		Num Channels	+ −		
		Alias Name 00	Channel 1		

The Reporting Collector Interval collects the data from the input channels and sends the data to the Reporting Server after an event or after the buffer is filled, depending on the configuration. An event is triggered when the timespan of the configured interval has expired.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

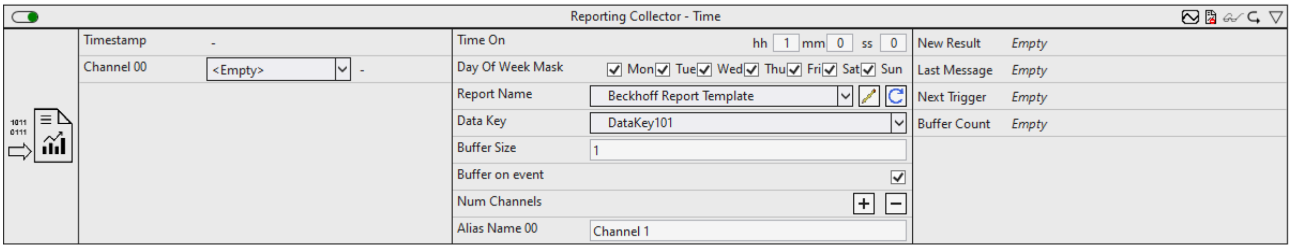
Configuration options

- **Interval:** time interval at which the data should be sent to the Reporting Server.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.
- **Data Key:** the data key should be unique within a report. The data object can be uniquely assigned to the report via the data key.
- **Buffer Size:** specifies the size of the buffered data. If the buffer size is equal to one, a key-value structure is built. If the buffer size is greater than one, the data is presented in a table.
- **Buffer on event:** specifies how the data is to be collected and buffered.
  If the parameter is True, the data of the inputs is buffered at each edge signal at the input. As soon as the Buffer Count output has the same value as the Buffer Size parameter, the data is sent to the TcReportingServer.
  If the parameter is False, the data of the inputs are buffered in the buffer at each cycle. Once the buffer size is reached, the new data replaces the previous data. If there is an edge signal at the input, the data is sent to the TcReportingServer.
- **Num Channels:** the number of input channels from which the data will be collected. And the number of parameters to allow a description of the input channels.
- **Alias Name:** serves as a description of the input channel. BufferSize = 1: the alias name serves as a key in the key-value structure. BufferSize > 1: the alias name serves as the heading of the table column.
- **Include Timestamps (optional):** inserts a column with the timestamps**.**

Output values

- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Current Interval Time:** indicates the amount of time that has already elapsed from the current interval.
- **Buffer Count:** specifies the number of elements in the buffer.

6.9.2.1.1.3 Reporting Collector Time



Reporting Collector - Time

Timestamp -
Channel 00 <Empty> -

Time On hh 1 mm 0 ss 0
Day Of Week Mask ☑ Mon ☑ Tue ☑ Wed ☑ Thu ☑ Fri ☑ Sat ☑ Sun
Report Name Beckhoff Report Template
Data Key DataKey101
Buffer Size 1
Buffer on event ☑
Num Channels + −
Alias Name 00 Channel 1

New Result *Empty*
Last Message *Empty*
Next Trigger *Empty*
Buffer Count *Empty*

The Reporting Collector Time collects the data from the input channels and sends the data to the Reporting Server after an event or after the buffer is filled, depending on the configuration. An event is triggered when the configured switch-on time is reached. The switch-on time and the days of the weeks can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Time On:** switch-on time.
- **Day of Week Mask:** weekdays on which the timer should be active.

- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.
- **Data Key:** the data key should be unique within a report. The data object can be uniquely assigned to the report via the data key.
- **Buffer Size:** specifies the size of the buffered data. If the buffer size is equal to one, a key-value structure is built. If the buffer size is greater than one, the data is presented in a table.
- **Buffer on event:** specifies how the data is to be collected and buffered.
  If the parameter is True, the data of the inputs is buffered at each edge signal at the input. As soon as the Buffer Count output has the same value as the Buffer Size parameter, the data is sent to the TcReportingServer.
  If the parameter is False, the data of the inputs are buffered in the buffer at each cycle. Once the buffer size is reached, the new data replaces the previous data. If there is an edge signal at the input, the data is sent to the TcReportingServer.
- **Num Channels:** the number of input channels from which the data will be collected. And the number of parameters to allow a description of the input channels.
- **Alias Name:** serves as a description of the input channel. BufferSize = 1: the alias name serves as a key in the key-value structure. BufferSize > 1: the alias name serves as the heading of the table column.
- **Include Timestamps (optional):** inserts a column with the timestamps**.**

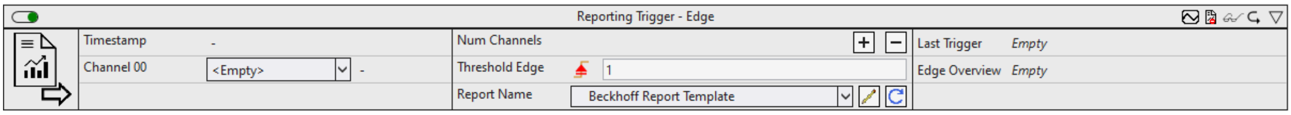
Output values

- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Next Message:** indicates the remaining time until the next message.
- **Buffer Count:** specifies the number of elements in the buffer.

6.9.2.1.2 Reporting Trigger

The Reporting Triggers send a trigger message to the Reporting Server after an event and thus trigger the creation of a report.

6.9.2.1.2.1 Reporting Trigger Edge



Reporting Trigger - Edge

Timestamp	-	Num Channels		Last Trigger	*Empty*
Channel 00	<Empty> -	Threshold Edge	1	Edge Overview	*Empty*
		Report Name	Beckhoff Report Template		

The Reporting Trigger Edge triggers the creation of a report after an event is triggered. An event is triggered when the input channel signal exceeds the configured edge at a specified threshold. Internally, the inputs that were once True remain True. The inputs are only reset to False as soon as all inputs were True at least once. This allows the output bNewResult to be used as one input by multiple Reporting Collectors and once all Reporting Collectors have sent a data message, a trigger message is sent.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

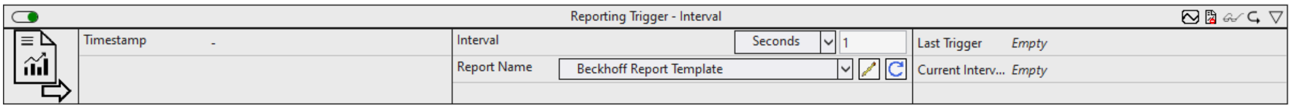
Configuration options

- **Type of the edge:** Specifies whether the edge counter should respond to a rising or falling edge.
- **Threshold:** Threshold of the signal at the respective edge. The event is triggered when the signal passes this threshold.
- **Tolerance (optional):** Tolerance value for the Equal / NotEqual comparisons.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.

Output values

- **Last Trigger:** indicates when the last message was sent to the Reporting Server.
- **Edge Overview:** indicates which input channels were True at least once.

6.9.2.1.2.2 Reporting Trigger Interval



Reporting Trigger - Interval

Timestamp -

Interval: Seconds 1

Report Name: Beckhoff Report Template

Last Trigger: *Empty*

Current Interv... *Empty*

The Reporting Trigger Interval triggers the creation of a report after an event has been triggered. An event is triggered when the timespan of the configured interval has expired.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

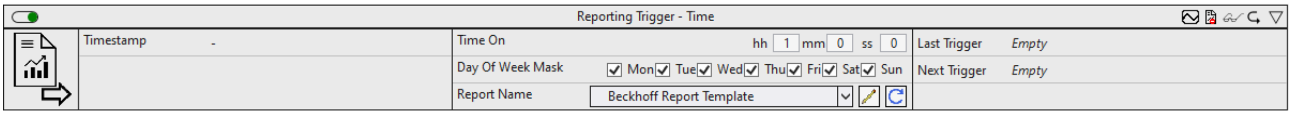
Configuration options

- **Interval:** time interval at which the data should be sent to the Reporting Server.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.

Output values

- **Last Trigger:** indicates when the last message was sent to the Reporting Server.
- **Current Interval Time:** indicates the amount of time that has already elapsed from the current interval.

6.9.2.1.2.3 Reporting Trigger Time



Reporting Trigger - Time

Timestamp -

Time On: hh 1 mm 0 ss 0

Day Of Week Mask: ☑ Mon ☑ Tue ☑ Wed ☑ Thu ☑ Fri ☑ Sat ☑ Sun

Report Name: Beckhoff Report Template

Last Trigger: *Empty*

Next Trigger: *Empty*

The Reporting Trigger Time triggers the creation of a report after an event has been triggered. An event is triggered when the configured switch-on time is reached. The switch-on time and the days of the weeks can be configured.

Optionally, a Boolean signal can be selected for the *Enable Execution* input so that the algorithm is only active if the value of the selected signal is *TRUE*.

Configuration options

- **Time On:** switch-on time.
- **Day of Week Mask:** weekdays on which the timer should be active.
- **Report Name:** specifies the name of the report. The name must correspond to the name of the configuration file at the Reporting Server. The drop-down menu displays all report names that are located in the Configuration folder of the Reporting Server. The Edit button can be used to edit the selected configuration file.

Output values

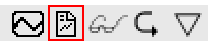
- **Last Message:** indicates when the last message was sent to the Reporting Server.
- **Next Trigger:** indicates the remaining time until the next message.

6.9.2.2 On-demand reporting

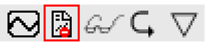
On-demand reporting provides the ability to manually integrate algorithm data and additional data into a report.

Each algorithm has the same five icons in the upper right corner: The second icon from the left is called **Include to Report** icon. This icon allows you to add the algorithm to a manual report.

This icon symbolizes that the algorithm is added to a manual report:



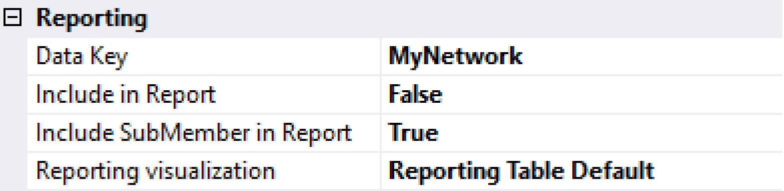
This icon symbolizes that the algorithm is not included in a manual report:



A network also contains these icons. Via these, all containing networks and algorithms are added to or ignored from the manual report.

The functionality of the **Include to Report** icon can also be mapped via the properties.

For a network, the properties look like this:



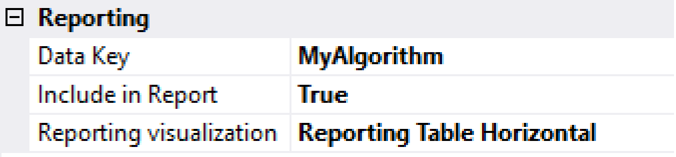
⊟ **Reporting**	
Data Key	**MyNetwork**
Include in Report	**False**
Include SubMember in Report	**True**
Reporting visualization	**Reporting Table Default**

Closed networks are treated like algorithms in on-demand reporting.

Reporting

Data Key	The Data Key uniquely identifies the data object in the report. The Data Key is preset with the network name by default.
Include in Report	This option specifies whether the network is added to a manual report. The network inputs and outputs or the network parameters are used as contents to be displayed.
Include SubMember in Report	This option specifies whether the Analytics elements contained in the network should be integrated into the report.
Reporting visualization	This option specifies how the data of the network should be presented.

For an algorithm, the properties look like this:

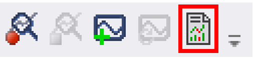


⊟ **Reporting**	
Data Key	**MyAlgorithm**
Include in Report	**True**
Reporting visualization	**Reporting Table Horizontal**

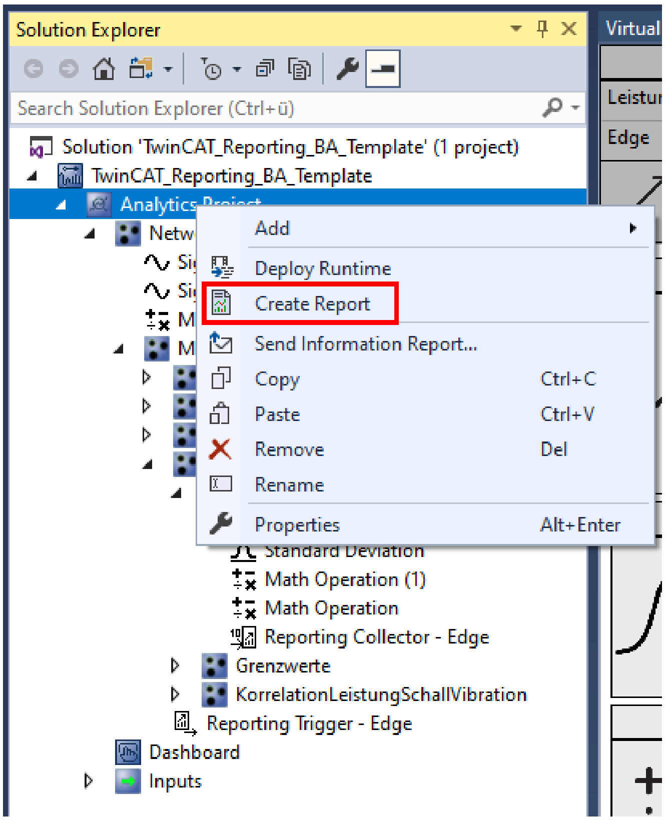
Reporting

Data Key	The Data Key uniquely identifies the data object in the report. The Data Key is preset with the network name and the name of the algorithm by default.
Include in Report	This option specifies whether the algorithm is added to a manual report.
Reporting visualization	This option specifies how the algorithm data should be presented.

On-demand reporting can be started in two ways. On the one hand, the reporting can be opened via the Reporting button in the toolbar:

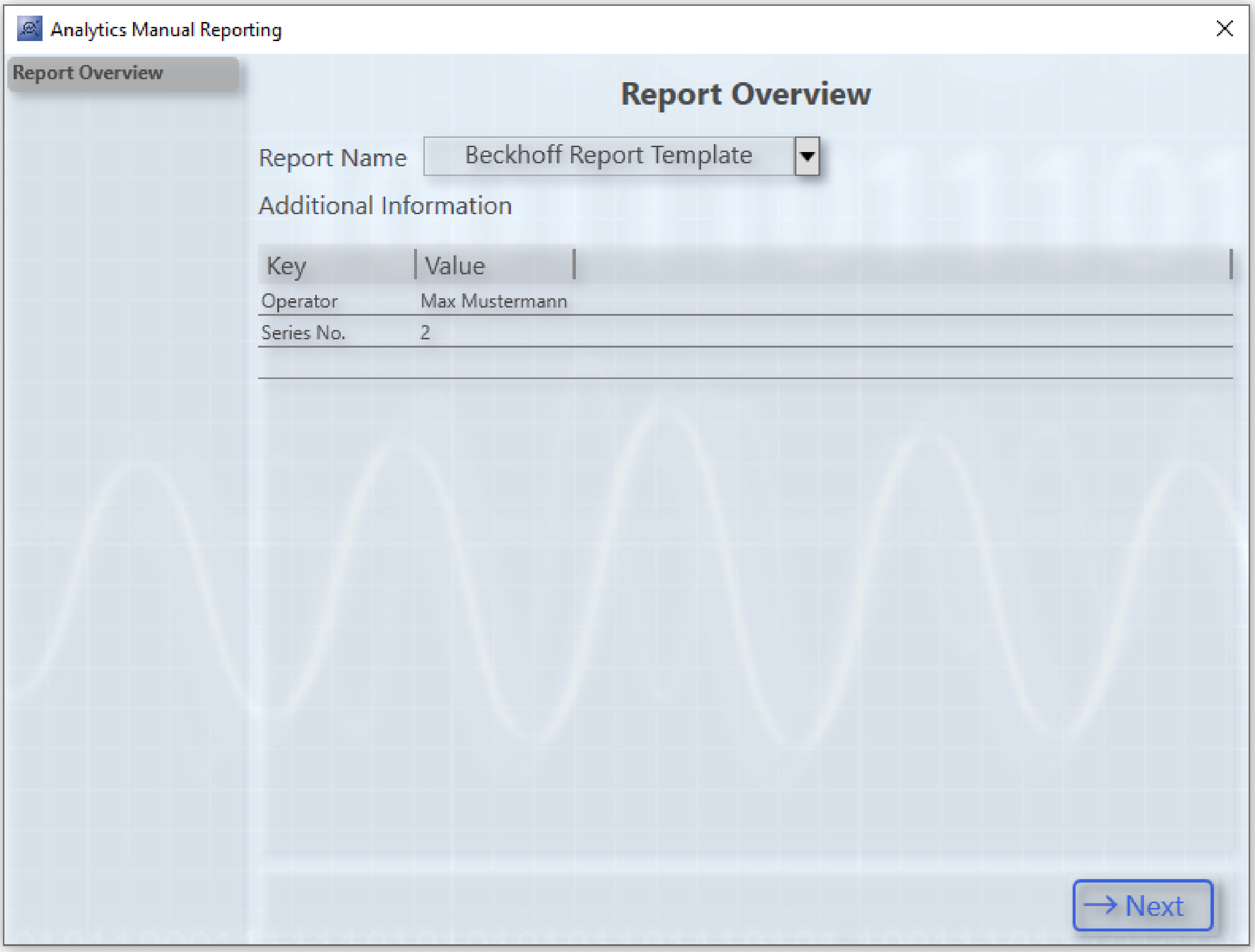


On the other hand, it can be started via the context menu on the Analytics project:



The wizard for on-demand reporting then opens.

On the first page, additional information can be added to the report.



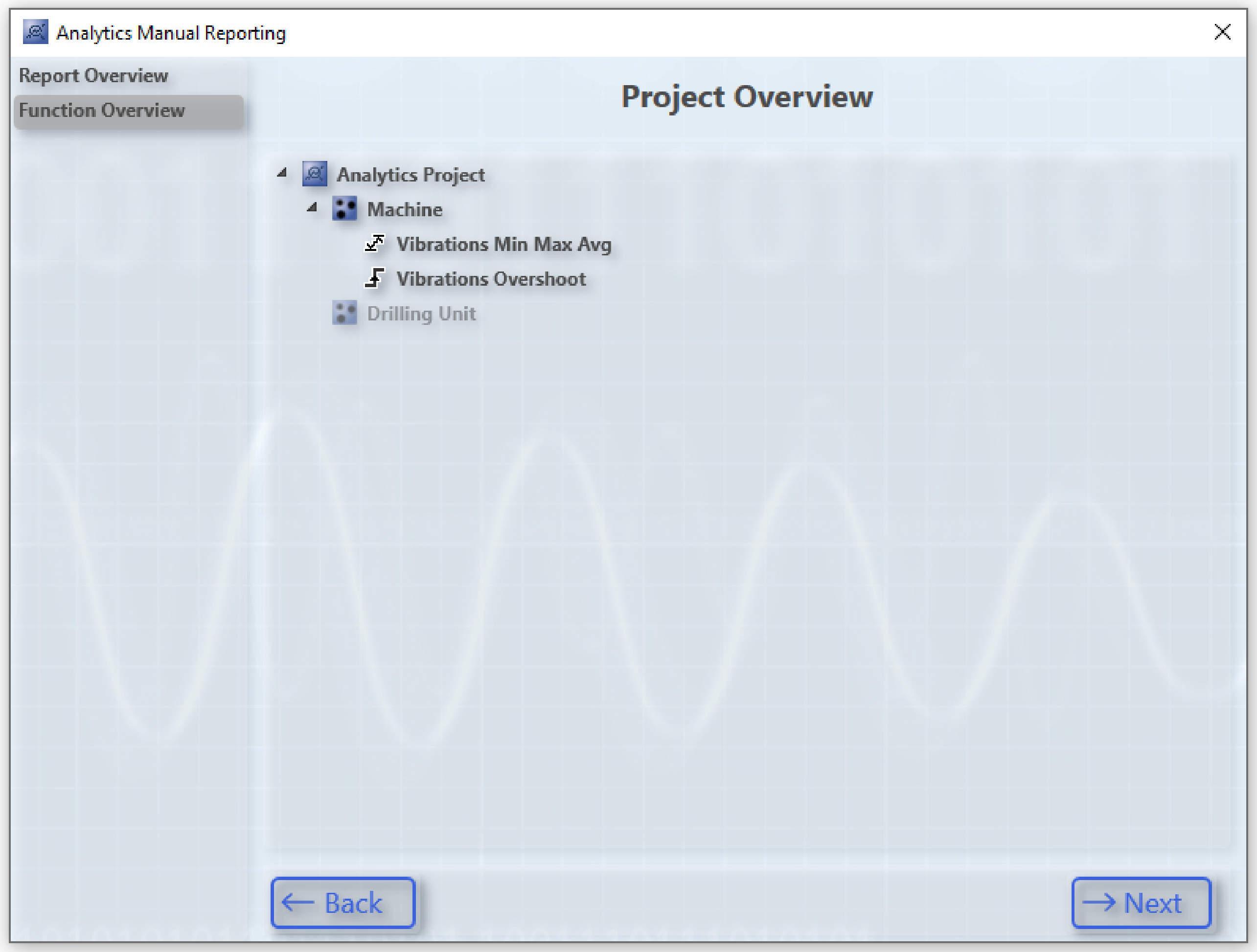
Report Name: Beckhoff Report Template

Additional Information

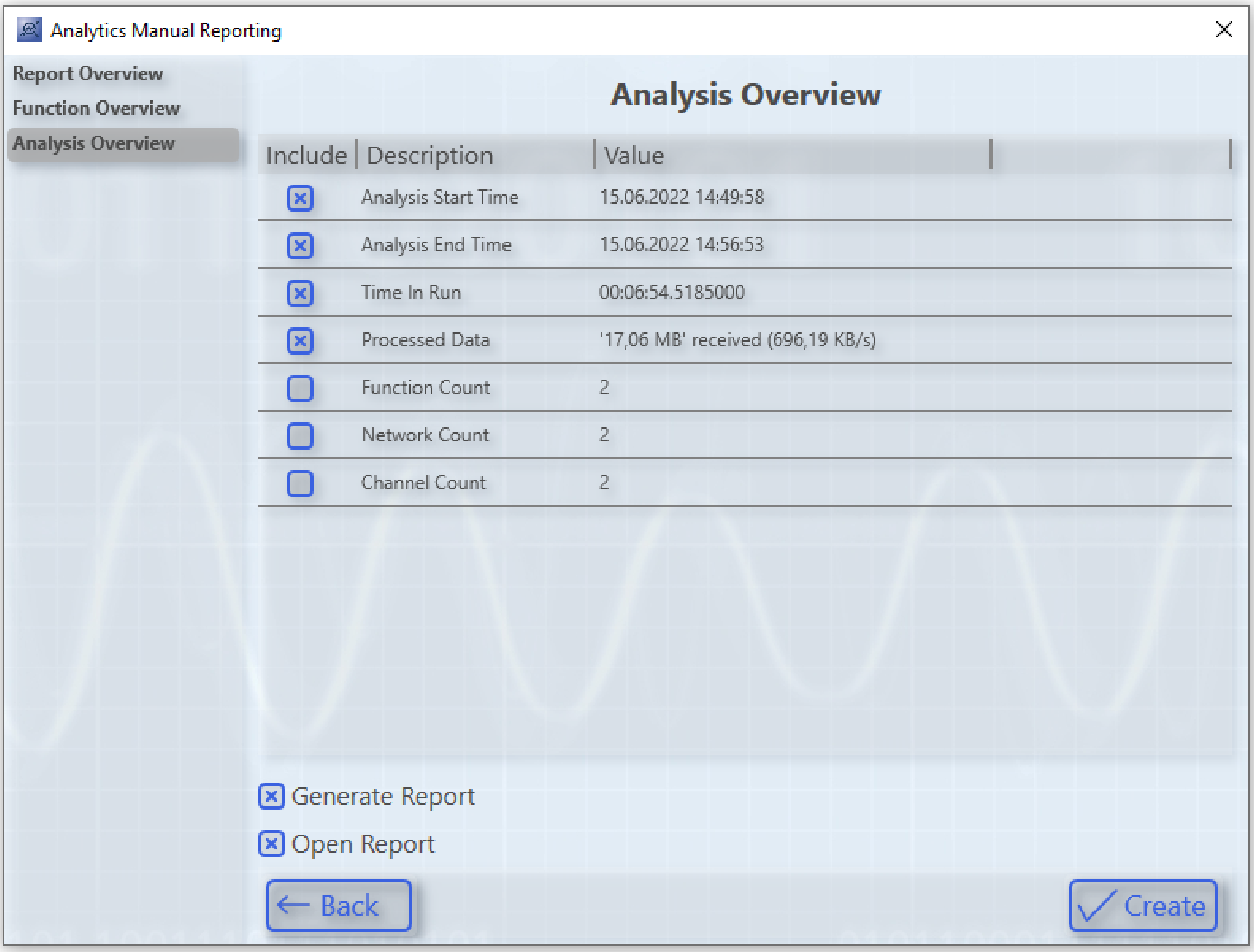
Key	Value
Operator	Max Mustermann
Series No.	2

The additional information is built in a key-value structure. Another line can be added by clicking on the empty line. A selected line can be deleted by pressing the Delete key.

An overview of the project is presented on the second page. The grayed out algorithms and networks are not included in the report.



On the third page, information about the analysis can be added to the report.



Analysis Overview

Include	Description	Value
☒	Analysis Start Time	15.06.2022 14:49:58
☒	Analysis End Time	15.06.2022 14:56:53
☒	Time In Run	00:06:54.5185000
☒	Processed Data	'17,06 MB' received (696,19 KB/s)
□	Function Count	2
□	Network Count	2
□	Channel Count	2

☒ Generate Report

☒ Open Report

In the table, a lot of information is available for integration in a report. The checkbox in the Include column can be used to select or deselect individual pieces of information.

The Generate Report option defines whether the report should be generated with the configured data. If this option is not selected, only the data will be sent to the Reporting Server, but the report will not be generated yet. This option is useful when manually adding data from multiple products to a single report.

The Open Report option defines whether the report should be opened after generation. To use this option, a FilePublishLocation with PDF or HTML formats must be defined in the configuration file. If the report is generated in PDF and HTML format, the PDF report will be opened by default.

6.9.3 TwinCAT Reporting Server

6.9.3.1 Design

6.9.3.1.1 Configuration file

The configuration file offers the possibility to organize, sort and customize the report. The configuration files must be stored in the folder *.\TwinCAT\Functions\Reporting-Server\Configuration\ConfigFiles*.

The following options are supported in the Reporting Server.

Key	Description	Sample	Leve l
General			
Store Type	Specifies which store mode to use. Currently only "Ram" is supported.	`"StoreType": "Ram"`	1

Header	Provides the possibility to display three headers arranged one below the other. If the second and / or third header is not to be used, an empty string can be specified.	`"Header": [`
		`"Header 1",`
		`"Header 2",`
		`"Header 3",`
		`]`
			1

Footer	The footer of a PDF page is divided into three areas. All three areas can be individualized with a free text. To display the date and time of creation, the placeholder "{date} {time}" must be specified. To display the page number, the placeholder "{page} of {total-pages}" must be specified. If a range is to be left blank, an empty string can be specified.	`"Footer": [`
		`"Footer Left",`
		`"Footer Center",`
		`"Footer Right",`
		`]`
			1

Logo	The logo is displayed on the top right of the first page. The logo can be specified as a relative path to the folder *Reporting-Server\Configuration\ConfigFiles*. Alternatively, the logo can be specified as a Base64String with the preceding encoding.	`"Logo": ".//<Bildname.png>"`
		`oder`
		`"Logo": "data:image/<Encoding>;base64,<Base64String>"`
			1

StyleSheet	Offers the possibility to customize the design of a report. To use a user-specific style sheet, its name must be specified (see Style Sheet [▶ 517]).	`"StyleSheet": "StyleBeckhoff.css"`	1
SignatureSettingsFile	Provides the possibility to sign a report (see Signing [▶ 520]).	`"SignatureSettingsFile": "BeckhoffSignatures.json",`	
Email text	This text will be included in every email unless a specific text is specified.	`"EmailText": "Custom of E-Mails send to all recipients with no or empty E-Mail-Text. Special. Ones"`	1
Required Keys	Is a collection from Data Key. If the data objects of these data keys are not available in the Reporting Server, the report will not be generated.	`"RequiredKeys": ["Data101"]`	1
Trigger On Required Key Received	Is another way to trigger the generation of a report. If true is selected for the option and all data objects with the required Data Keys are present, the report will be generated. This eliminates the need for a trigger message from a Reporting Client.	`"TriggerOnRequiredKeyReceived" : false`	1
Static Default Data			
Default Data	This object can be used to define data objects to be presented in each report.	`"DefaultData": {}`	1
Text Fields	This element is an array and can have multiple text field objects with the following elements.	`"TextFields": []`	2
Key	Used as a heading for the text field.	`"Key": "My Text Field"`	3
Comment	Serves as a comment for the data object. The comment is not displayed in the report.	`"Comment": "My Comment"`	3
Sort Priority	Specifies the position of the data object in the report.	`"SortPriority": 42`	3
Value	Contains an individual value and must be specified as STRING.	`"Value": "253-5553-9421"`	3
Tables	This element is an array and can have multiple table objects with the following elements.	`"Tables": []`	2

Key	Description	Sample	Leve l
Key	Used as a heading for the table.	`"Key": "My Table"`	3
Comment	Serves as a comment for the data object. The comment is not displayed in the report.	`"Comment": "My Comment"`	3
Sort Priority	Specifies the position of the data object in the report.	`"SortPriority": 42`	3
Value	This element is an object and contains the following elements.	`"Value": {}`	3
Header	This element is an array and receives the column titles of the table. The number of column titles must be equal to the columns of the table.	`"Header": ["Table Header 1", "Table Header 2"]`	4
DataTypes	This element is an array and receives the data types of the table column. The following data types are possible: BOOL, BYTE, SBYTE, CHAR, DECIMAL, DOUBLE, FLOAT, INT, UINT, LONG, ULONG, SHORT, USHORT, STRING. The number of data types must be equal to the columns of the table.	`"DataTypes": ["string", "double"]`	4
Values	This element is a 2D array and receives the value of the table. An array in the array represents a table column. The table column must match the specified data type. The number of arrays must be equal to the columns of the table. The arrays should be the same size.	`"Values": [[],[]]`	4
Images	This element is an array and can have multiple image objects with the following elements.	`"Images": []`	2
Key	Used as a caption for the image.	`"Key": "My Image"`	3
Comment	Serves as a comment for the data object. The comment is not displayed in the report.	`"Comment": "My Comment"`	3
Sort Priority	Specifies the position of the data object in the report.	`"SortPriority": 42`	3
Value	Contains the Base64String with PNG encoding.	`"Value": "<Base64String PNG-Encoding>"`	3
Value Pairs	This element is an array and can have multiple key-value pair objects with the following elements.	`"ValuePairs": []`	2
Key	Used as a heading for the collection from key-value pairs.	`"Key": "My Value Pair"`	3
Comment	Serves as a comment for the data object. The comment is not displayed in the report.	`"Comment": "My Comment"`	3
Sort Priority	Specifies the position of the data object in the report.	`"SortPriority": 42`	3
Value Pair Collection	Serves as a collection of key-value pairs. This element is an array and can have multiple objects with the following elements.	`"ValuePair_Collection": []`	3
Key	Contains an individual value and is displayed presented on the left side.	`"Key": "Serial number"`	4
Data Type	Describes the data type of the "Value" element.	`"DataType": "string"`	4
Value	Contains an individual value and is presented on the right side. The value must be specified as STRING.	`"Value": "253-5553-9421"`	4
Data Configurations			
Data Configurations	Data Configurations can be used to sort and individualize the data that is sent dynamically from the Reporting Clients. This element is an array and can have multiple objects with the following elements.	`"DataConfigurations": []`	1

Key	Description	Sample	Level
Key	Corresponds to the Data Key of the data object. This ensures that this data configuration is applied to a unique data object. (Required)	`"Key": "DataTable101"`	2
Title	Provides the option of a better name as a heading. (optional)	`"Title": "Rotations and vibrations"`	2
SortPriority	Specifies the position of the data object in the report. (Required)	`"SortPriority": 42`	2
IsRequired	Indicates whether this data object is required for generating the report. (see: TriggerOnRequiredKey)	`"IsRequired": false`	2
StoreMode	Provides the option to store individual data objects in a specific way and display them accordingly. Currently, four different store modes are supported: • **Blocking**: The first data object is saved and all subsequent data objects with the same data key are discarded. • **Overwriting**: The newly received data object replaces the last data object with the same data key. • **Appending**: The newly received data object is saved. The previously received data objects are also held and presented in the report. In the case of a table, the data is merged. • **Appending_Series**: This store mode corresponds to the store mode "Appending". However, the data objects are numbered and presented in individual tables.	`"StoreMode": "Overwriting"`	2
RoundTo	Provides the option to round floating-point numbers to a specified decimal place.	`"SortPriority": 4`	2
TimestampFormat	Provides the option to convert Timestamp to a specified format. By default, the following formatting is selected: "yyyy-MM-dd HH:mm:ss.ffff" (See also: Custom date and time format strings)	`"TimestampFormat": "HH:mm:ss.ffff"`	2
IsAsc	Provides the option to specify the sort order of a data table. If the value is true, the first data set is presented in the first line.	`"IsAsc": true`	2
StyleClass	Provides the option to use a custom CSS style class. The CSS style class must be defined in the CSS file and the CSS file must be referenced under the StyleSheet option.	`"StyleClass": "CustomClass"`	2
Publish Locations			
Publish Locations		`"PublishLocations": []`	1
Email Publish Location	Used to forward the report as an e-mail. This element is an array and can have multiple objects with the following elements.	`"EmailPublishLocations": []`	2
Type	Specifies the type of distribution.	`"Type": "File"`	3
Format	Offers the option to send only a selection of formats by email.	`"Format": ["PDF", "HTML", "JSON"]`	3
Address	Provides the ability to send the report to multiple email addresses.	`"Address": [ "m.mustermann@tester.com "]`	3
Use Bcc	Provides the option to set the email addresses in BCC.	`"UseBcc": false`	3

Key	Description	Sample	Leve l
Text	Provides the option to specify a specific email text. This text replaces the default email text.	`"Text": "Hello, \n insert email text here.\n Your Reporting Service.`	3
File Publish Location	Used to copy the report to a specified location. This element is an array and can have multiple objects with the following elements.	`"FilePublishLocations": []`	2
Type	Specifies the type of distribution.	`"Type": "File"`	3
Format	Offers the option, to send only a selection of formats by email.	`"Format": ["PDF", "HTML", "JSON"]`	3
Path	Provides the possibility to specify a destination path for the report. The destination path must not require administrator rights.	`"Path": "C:\\tmp\\CpyFolder"`	3

6.9.3.1.2 StyleSheet

The style sheets offer the possibility to customize the design of the report. The *StyleBeckhoff.css* style sheet serves as a template.This can be copied, renamed and customized. The name of the style sheet must then be specified in the corresponding configuration file. The style sheets must be stored in the folder . *\TwinCAT\Functions\Reporting-Server\Configuration\ReportStyleSheets*.
Basically, all HTML objects contained in an HTML report can be individualized in the custom style sheet. The following CSS objects are frequently used and are ideal for customization:

CSS element	Description
html	The HTML element refers to the entire report.
h1	Main heading of the header in the report
h3	Subheading of the header in the report; heading of a data object

The following CSS classes are used to design data objects:

CSS class	Description
reporting-header	Class for the header of the report.
reaporting-header img	Class for the icon on the top right of the header.
reporting-header table	The header is tabular and can be customized via this class.
reporting-body	Class for the content of the report.
reporting-body caption	The headings of the data objects are located in a Caption element.
reporting-body valuetable	Class for a data table in the report.
reporting-body valuepair	Class for a collection from key-value pairs in the report.
reporting-body infotable	Class for the additional information and analysis or recording information from the manual report.
reporting-body figure	Class for images, such as charts from the scope in the report.
reporting-body img	This class contains the actual image. It is located in the figure element.

6.9.3.2 Report format

The reports are generated in the following formats.

1. PDF

2. HTML
3. JSON

6.9.3.3 Network Configuration

A possible use case could be that the Reporting Server and the Reporting Client are running on different systems. The Reporting Server and the Reporting Client must be on the same network. To be able to map this use case, a JSON file must be extended. The file should be located under the following path and have the appropriate name: *.\TwinCAT\Functions\Reporting-Server\Configuration\Network\ReportingNetworkConfiguration.json*
A network configuration looks like this:

Name	Description	Data type
ConfigIds	Report names of the corresponding reports	List<string>
AmsNetId	AmsNetId of the target system. The Reporting Server is executed on the target system.	string

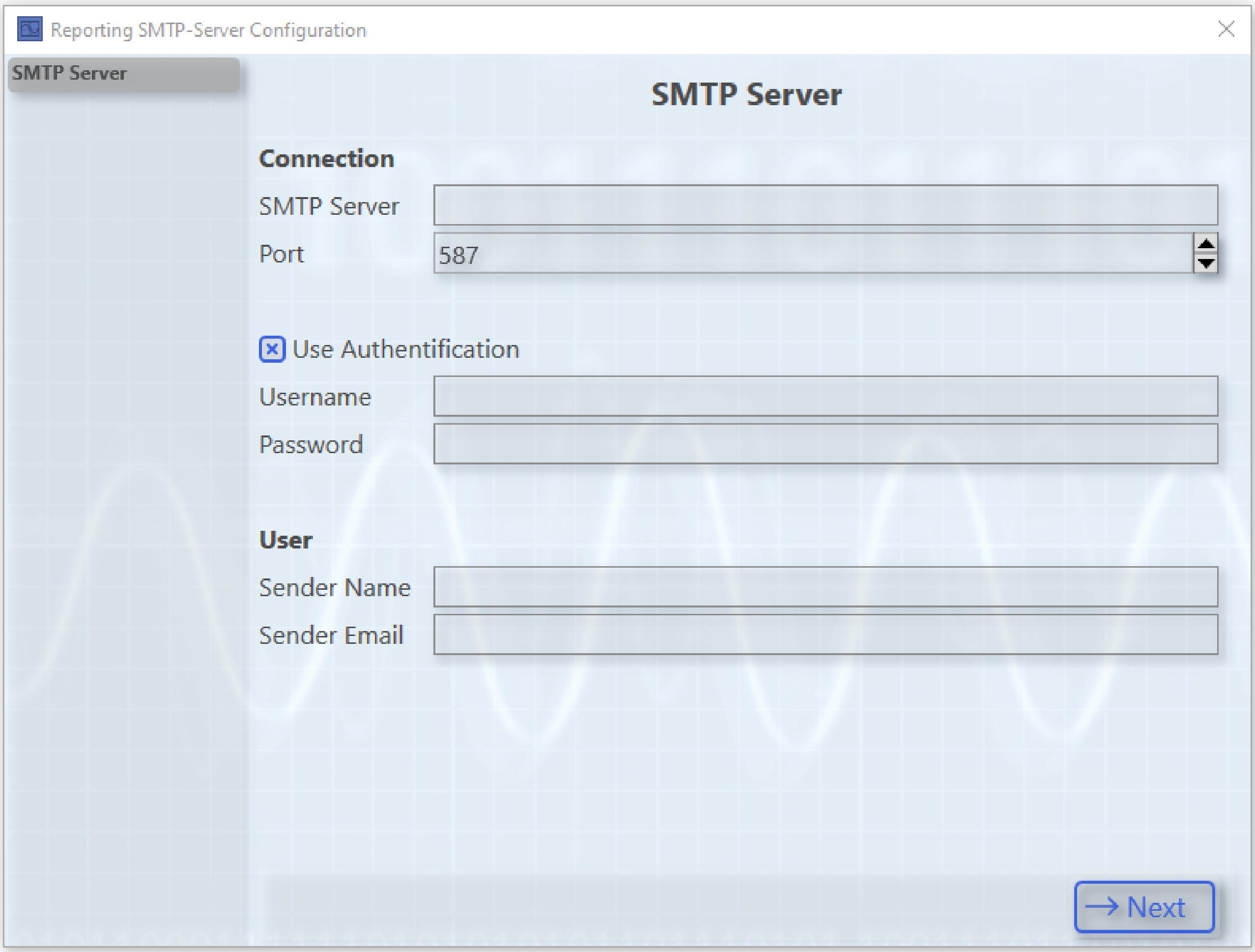
6.9.3.4 Email-Client configuration

A TwinCAT Reporting Server can be connected to an SMTP server. The SMTP server must be set up by the user and is not part of TwinCAT Reporting. The TwinCAT Reporting Server sends the configured email with the report to the SMTP server. This forwards the email to the recipient.

A wizard is available for configuring the connection to an SMTP server. The wizard can be opened in two ways.

1. **TwinCAT > Analytics > Reporting Email**
2. **Scope > Reporting Email**

In the first dialog you can configure a connection to the SMPT server.



SMTP Server

Connection

SMTP Server

Port 587

☒ Use Authentification

Username

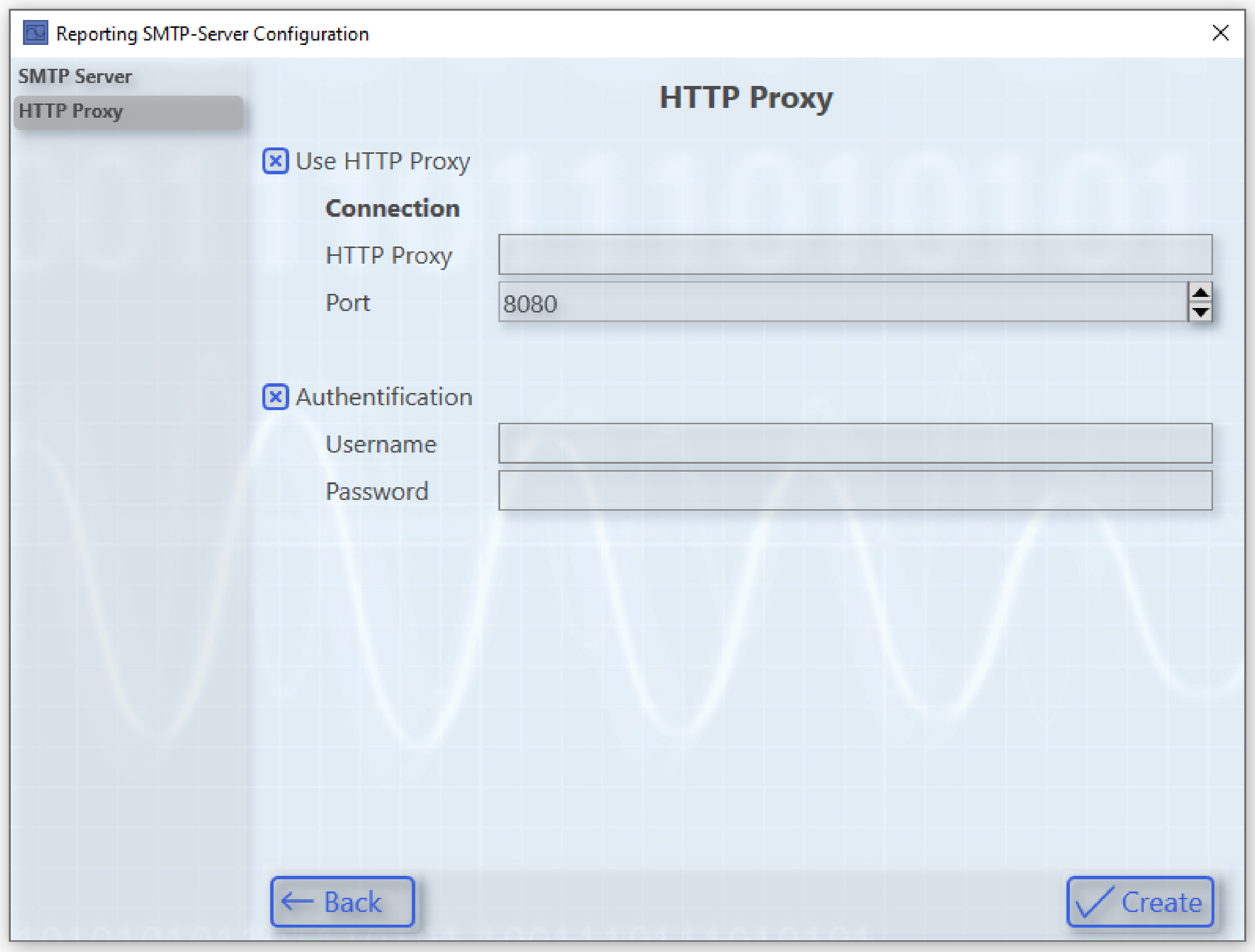
Password

User

Sender Name

Sender Email

The second dialog offers the possibility to configure a connection to an HTTP proxy. This option can be enabled if the user's architecture requires it.



Reporting SMTP-Server Configuration

HTTP Proxy

☒ Use HTTP Proxy

Connection

HTTP Proxy:

Port: 8080

☒ Authentification

Username:

Password:

After clicking the **Create** button, the configuration will be saved in a corresponding file.

6.9.3.5 Signing

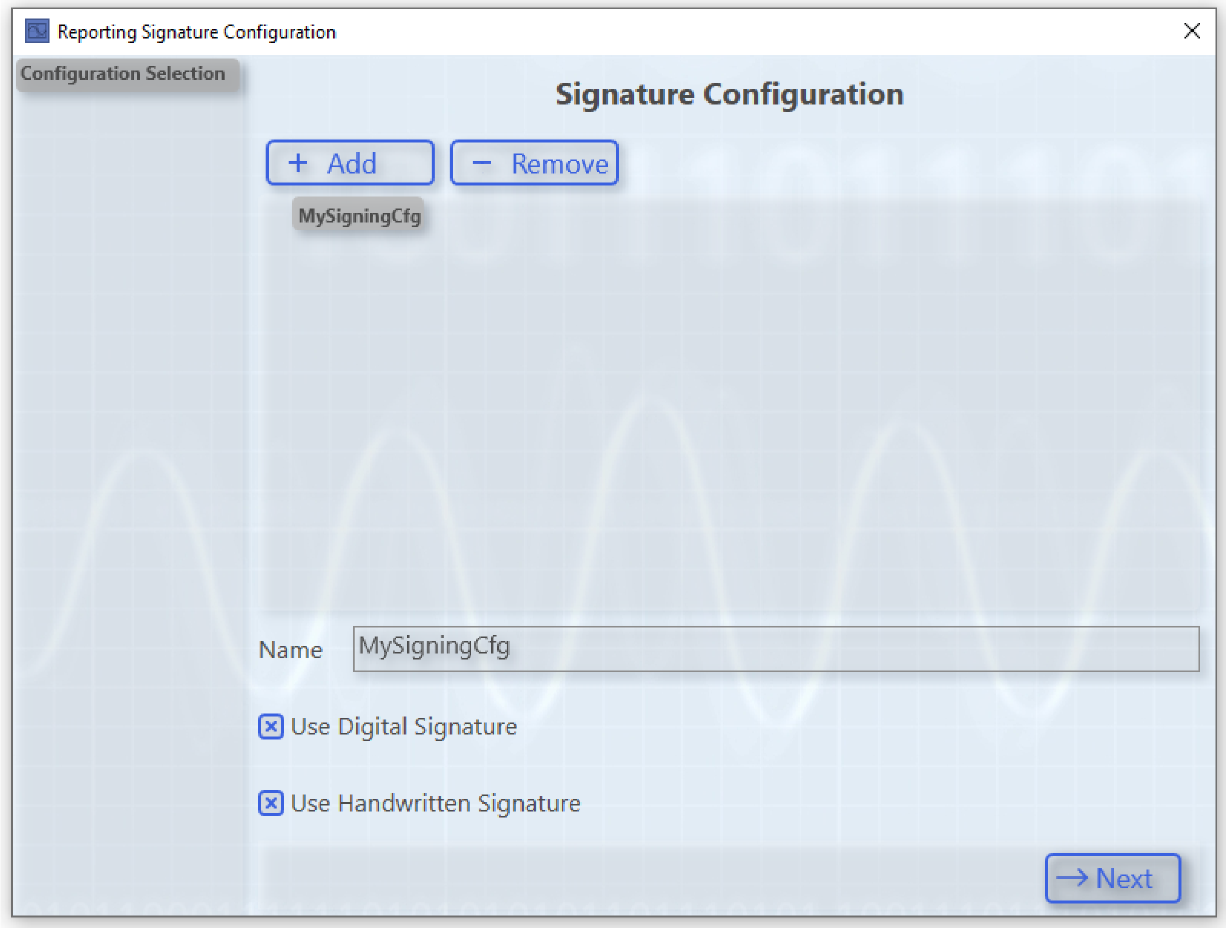
In order to verify reports, PDF reports can be signed in TwinCAT Reporting. Digital and handwritten signatures are supported. The wizard for configuring signatures can be opened in the following ways.

- **TwinCAT > Analytics > Reporting > Reporting Signatures**
- **Scope > Reporting > Reporting Signatures**

✓ The wizard offers the possibility to create multiple signature configurations, which can then be reused in multiple report configurations.

1. The name of this configuration must be referenced in the configuration file via the key "SignatureSettingsFile" (see Configuration file [▶ 513]).
   You can use the two checkboxes to select whether the configured signatures displayed on the next

pages should be used.
By clicking the button **Next** you can edit the selected configuration.



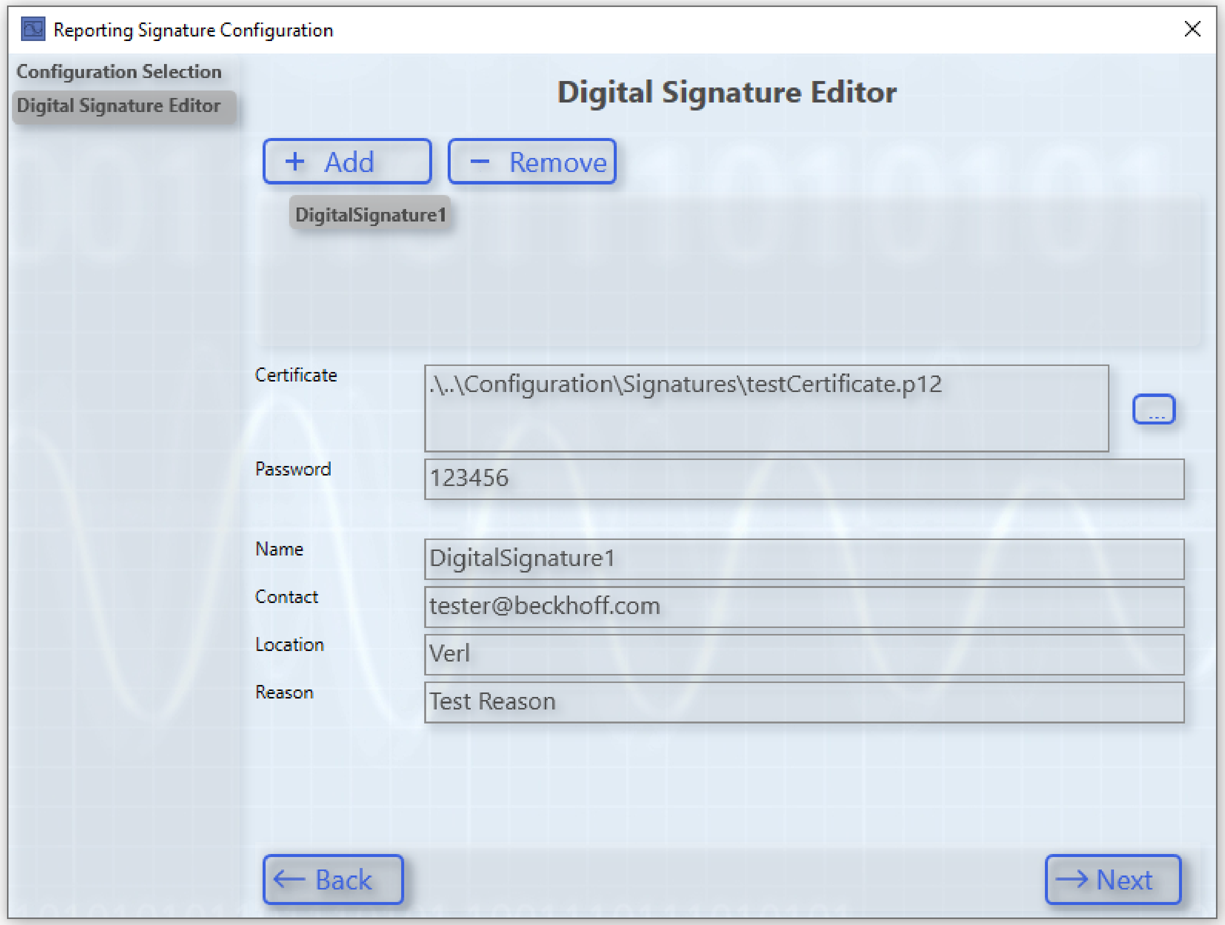
Reporting Signature Configuration

Name MySigningCfg

☒ Use Digital Signature

☒ Use Handwritten Signature

2. Here you can define a digital signature.



Certificate: .\..\Configuration\Signatures\testCertificate.p12

Password: 123456

Name: DigitalSignature1

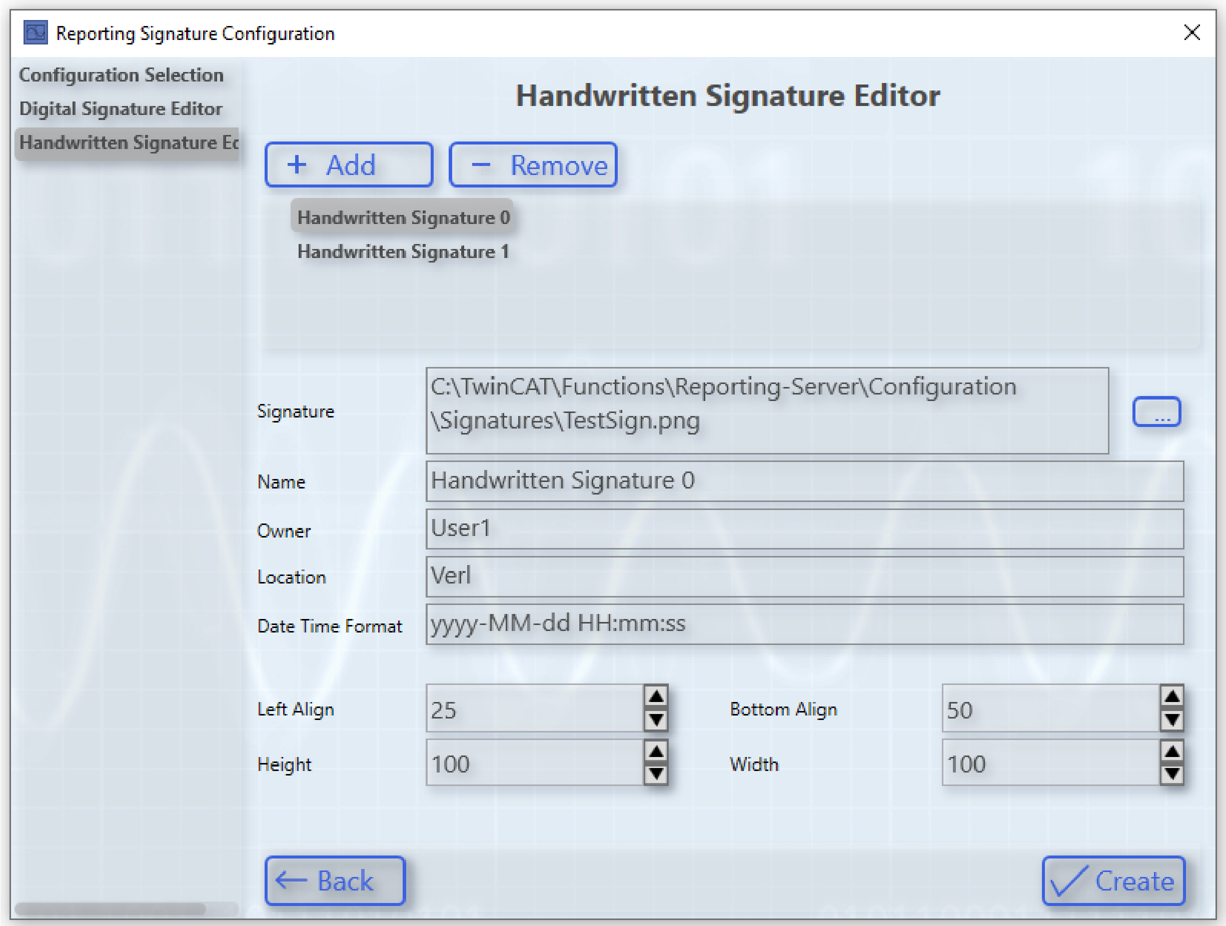
Contact: tester@beckhoff.com

Location: Verl

Reason: Test Reason

⇨ Pfx and p12 certificates are supported by TwinCAT Reporting.

3. Here you can define a handwritten signature.



Handwritten Signature 0
Handwritten Signature 1

Signature: C:\TwinCAT\Functions\Reporting-Server\Configuration\Signatures\TestSign.png
Name: Handwritten Signature 0
Owner: User1
Location: Verl
Date Time Format: yyyy-MM-dd HH:mm:ss
Left Align: 25
Bottom Align: 50
Height: 100
Width: 100

⇨ If you use handwritten signatures, an additional signature page is inserted into the report.

4. The signatures can be positioned via the left and bottom alignment. The origin is at the bottom left of the signature page.
   The following image shows an exemplary handwritten signature:

Verl, 2022-12-19 09:54:51
TestSigning
Owner
Gütersloh, 09:54:51
TestSigning
User

⇨ The signature configuration information is stored in the Reporting Server folder in plain text in a file.

6.9.3.6 Licensing

TwinCAT Reporting can be licensed as a full version and as a 7-day trial version.
The following products include the TwinCAT Reporting license.

Product number	Product name
TE1300	TC3 Scope View Professional
TF3300	TC3 Scope Server
TE3500	TC3 Analytics Workbench
TE3520	TC3 Analytics Service Tool

The following features are available with the full version or with the 7-day trial version in TwinCAT Reporting Server.

Features	**TwinCAT Reporting Server**	
	7-day trial	Full license
General:		
Reports	✓ (Max. 1)	✓ (unlimited)
Digital signatures	✗	✓
Handwritten signatures	✗	✓

Please refer to the product overview page for Reporting Client trial limitations.

6.10 Comparison of analysis results

Various analysis runs can be compared with the Analytics Diary. The Analytics Diary offers several advantages for exploratory data analysis and identifying an appropriate analysis method, including algorithm parameters.

1. By default, all analysis results including the project configuration are saved with each analysis run. You can change this setting at
Tools > Options > TwinCAT > Measurement > Analytics > Diary .

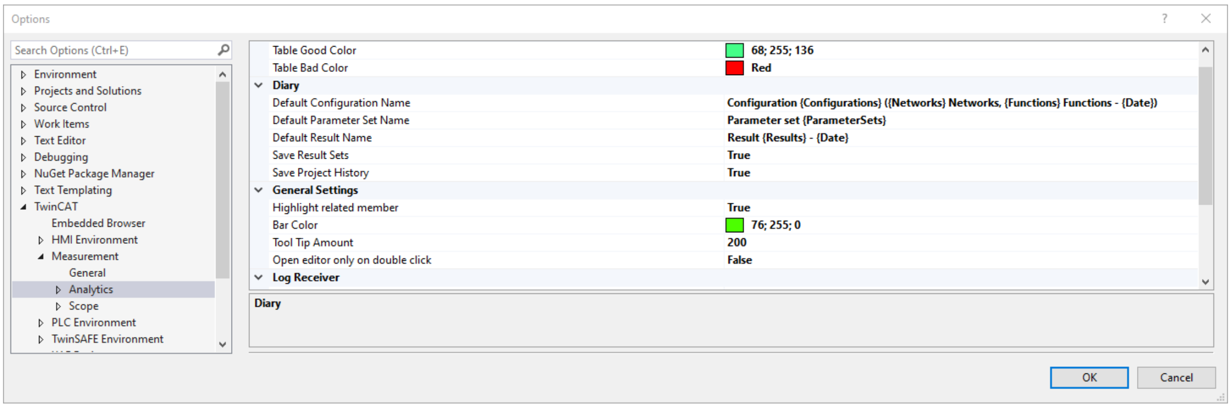
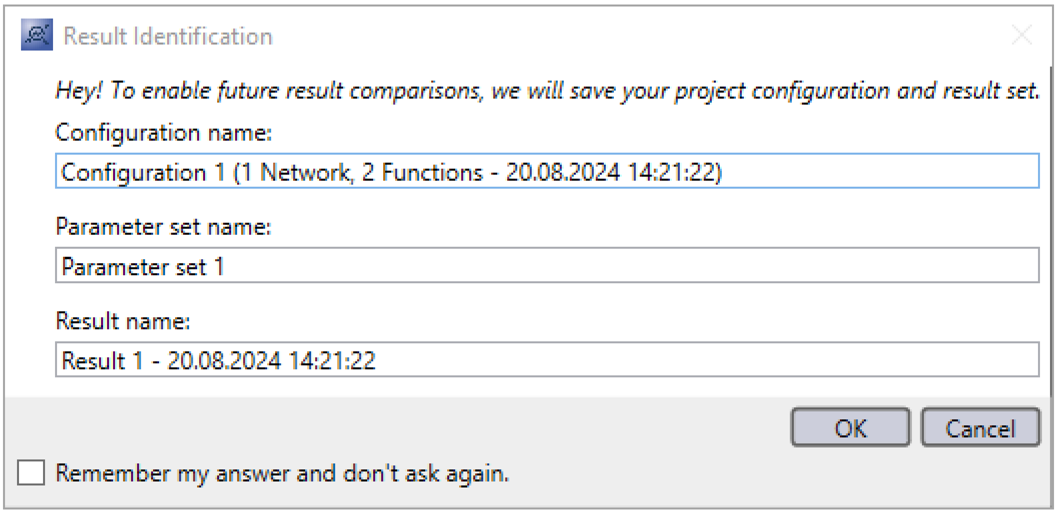


Table Good Color	🟩 68; 255; 136
Table Bad Color	🟥 Red
Diary	
Default Configuration Name	Configuration {Configurations} ({Networks} Networks, {Functions} Functions - {Date})
Default Parameter Set Name	Parameter set {ParameterSets}
Default Result Name	Result {Results} - {Date}
Save Result Sets	True
Save Project History	True
General Settings	
Highlight related member	True
Bar Color	🟩 76; 255; 0
Tool Tip Amount	200
Open editor only on double click	False
Log Receiver	

Diary

2. When you start an analysis, a Diary window appears. Here you can set the names of the configuration, the parameter set and the result. If nothing has changed in the project, only the data is saved. Likewise, no analysis configuration is generated if only the configuration of parameters changes (see also Global parameters [▶ 76]).



Hey! To enable future result comparisons, we will save your project configuration and result set.

Configuration name:

Configuration 1 (1 Network, 2 Functions - 20.08.2024 14:21:22)

Parameter set name:

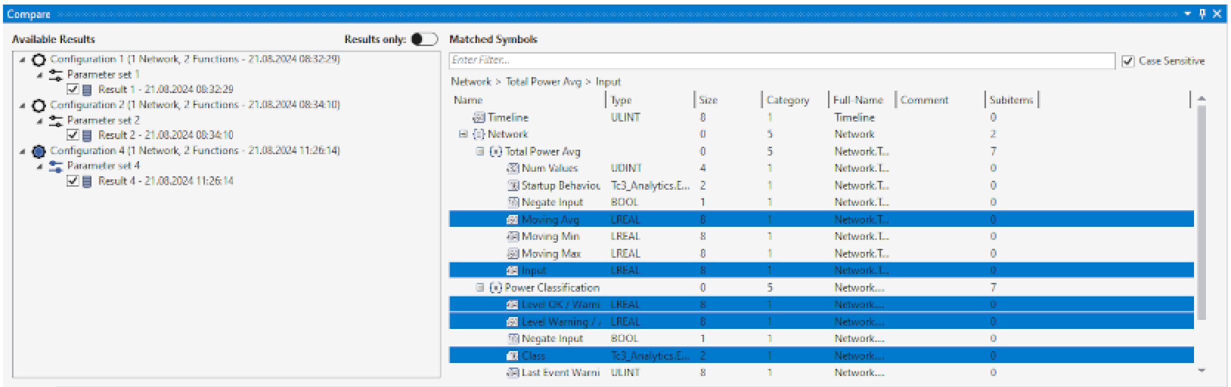
Parameter set 1

Result name:

Result 1 - 20.08.2024 14:21:22

□ Remember my answer and don't ask again.

3. These entries will then appear in a tree structure in the **Compare** window, which you can open by clicking on them in the **Solution Explorer**. This shows the history of analysis runs. You can select multiple result data sets and matching symbols, which can then be viewed in the **Table View** and **Chart View** windows. The selection of data sets and symbols is therefore synchronized with the other two windows. The last data set is always selected by default.

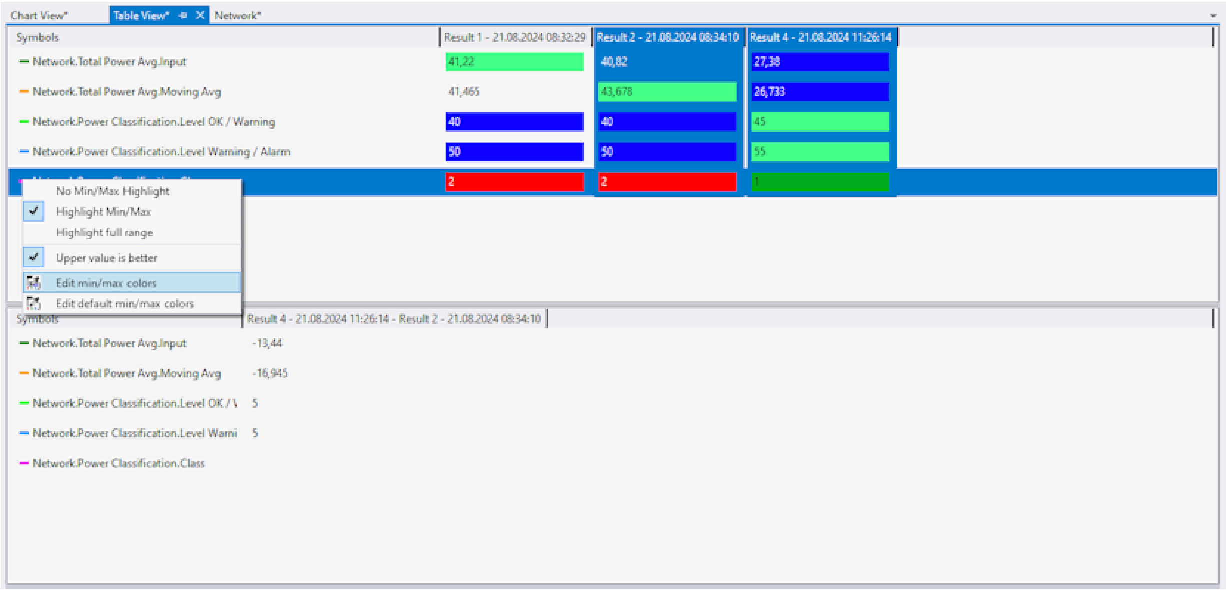


- Configuration 1 (1 Network, 2 Functions - 21.08.2024 08:32:29)
  - Parameter set 1
    - ☑ Result 1 - 21.08.2024 08:32:29
- Configuration 2 (1 Network, 2 Functions - 21.08.2024 08:34:10)
  - Parameter set 2
    - ☑ Result 2 - 21.08.2024 08:34:10
- Configuration 4 (1 Network, 2 Functions - 21.08.2024 11:26:14)
  - Parameter set 4
    - ☑ Result 4 - 21.08.2024 11:26:14

Network > Total Power Avg > Input

Name	Type	Size	Category	Full-Name	Comment	Subitems
Timeline	ULINT	8	1	Timeline		0
Network		0	5	Network		2
Total Power Avg		0	5	Network.T...		7
Num Values	UDINT	4	1	Network.T...		0
Startup Behavior	Tc3_Analytics.E...	2	1	Network.T...		0
Negate Input	BOOL	1	1	Network.T...		0
Moving Avg	LREAL	8	1	Network.T...		0
Moving Min	LREAL	8	1	Network.T...		0
Moving Max	LREAL	8	1	Network.T...		0
Input	LREAL	8	1	Network.T...		0
Power Classification		0	5	Network....		7
Level OK / Warni	LREAL	8	1	Network....		0
Level Warning /	LREAL	8	1	Network....		0
Negate Input	BOOL	1	1	Network....		0
Class	Tc3_Analytics.E...	2	1	Network....		0
Last Event Warni	ULINT	8	1	Network....		0

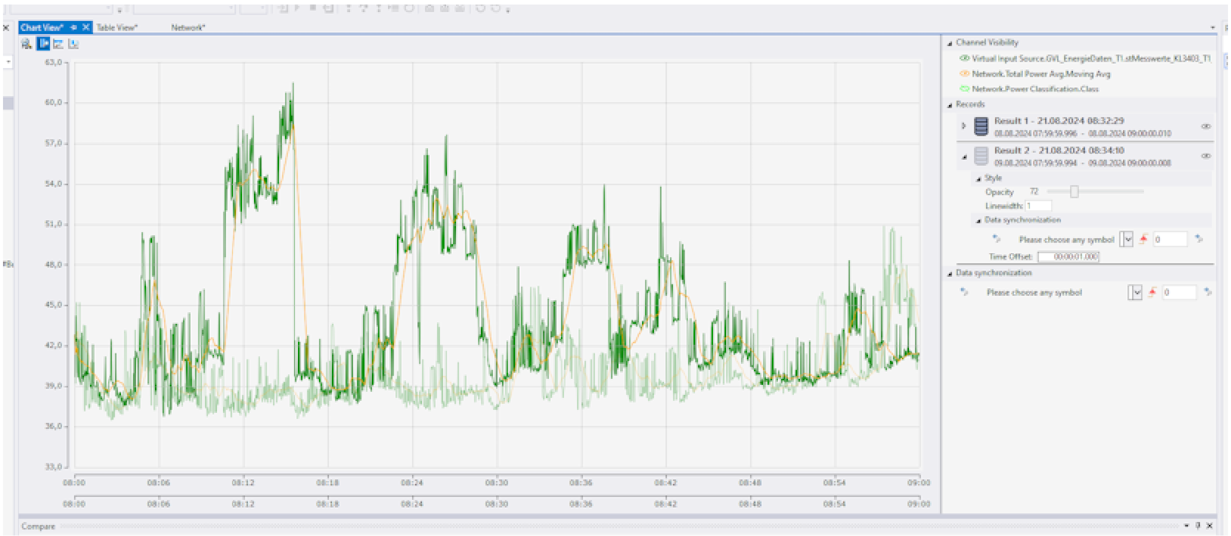
4. You can open the **Table View** and **Chart View** windows by clicking in the **Solution Explorer**. Two tables are displayed in the **Table View**. The first displays the last result of all selected data sets and symbols. The data sets are listed column by column and the symbols row by row. An optional color display can be used to visually highlight the differences between symbols. Furthermore, the differences between the symbols can be shown in a second table. At least two columns or data sets must be selected from the first table.



Symbols	Result 1 - 21.08.2024 08:32:29	Result 2 - 21.08.2024 08:34:10	Result 4 - 21.08.2024 11:26:14
Network.Total Power Avg.Input	41,22	40,82	27,38
Network.Total Power Avg.Moving Avg	41,465	43,678	26,733
Network.Power Classification.Level OK / Warning	40	40	45
Network.Power Classification.Level Warning / Alarm	50	50	55
	2	2	1

Symbols	Result 4 - 21.08.2024 11:26:14 - Result 2 - 21.08.2024 08:34:10
Network.Total Power Avg.Input	-13,44
Network.Total Power Avg.Moving Avg	-16,945
Network.Power Classification.Level OK / \	5
Network.Power Classification.Level Warni	5
Network.Power Classification.Class	

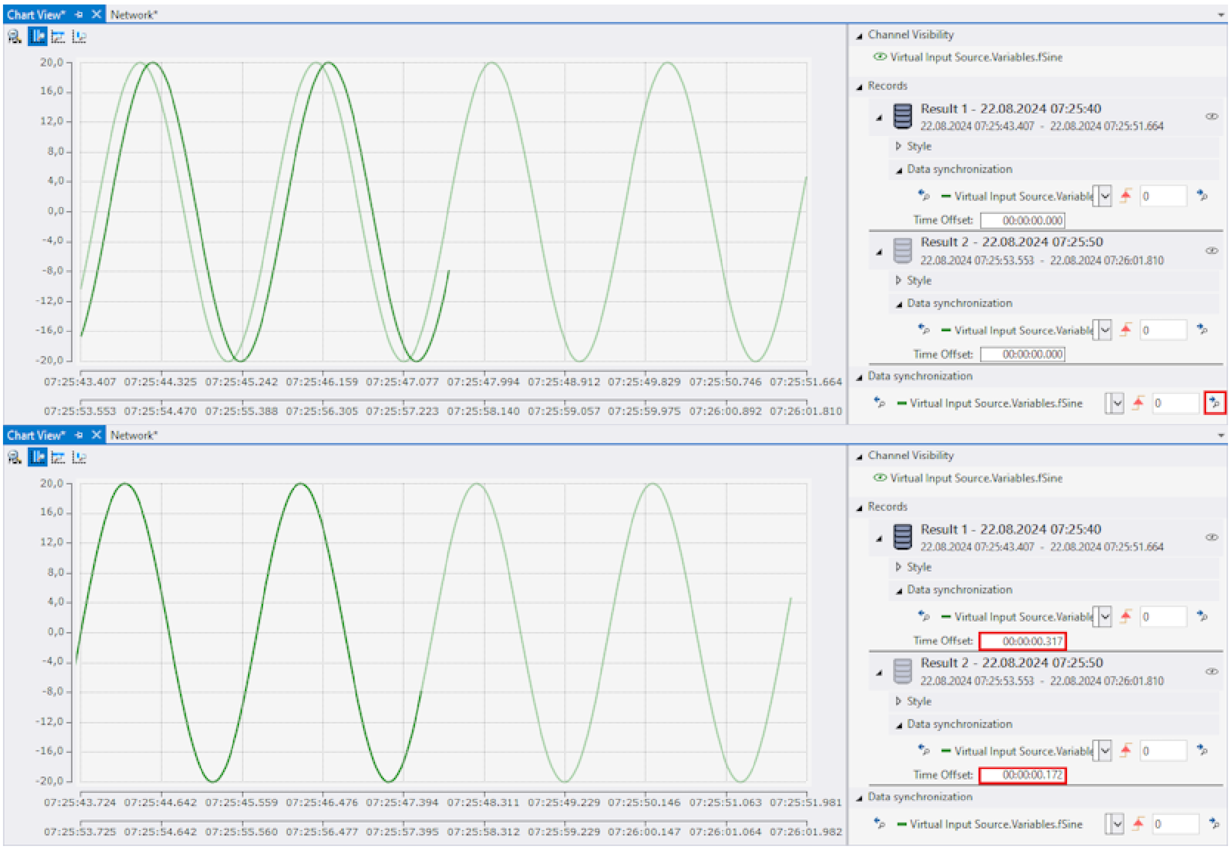
5. The *Chart View* is used to compare result data sets as time series. Different colors are assigned to the symbols. The symbols appear according to the number of selected data sets, whereby the opacity of the lines is reduced in each case. You can manually adjust the opacity and the line thickness to differentiate the data sets. To view a closer area, you can use a zoom equivalent to the scope.



Channel Visibility
Virtual Input Source.GVL_EnergieDaten_TI.stMesswerte_KL3403_TI
Network.Total Power Avg.Moving Avg
Network.Power Classification.Class

Records
Result 1 - 21.08.2024 08:32:29
08.08.2024 07:59:59.996 - 08.08.2024 09:00:00.010
Result 2 - 21.08.2024 08:34:10
09.08.2024 07:59:59.994 - 09.08.2024 09:00:00.008
Style
Opacity 72
Linewidth: 1
Data synchronization
Please choose any symbol 0
Time Offset: 00:00:01.000
Data synchronization
Please choose any symbol 0

6. If the start times of the data sets do not match, you can adjust them for a direct comparison. This can be done either by manually entering the time offset or by an automatic edge search.



Channel Visibility
Virtual Input Source.Variables.fSine

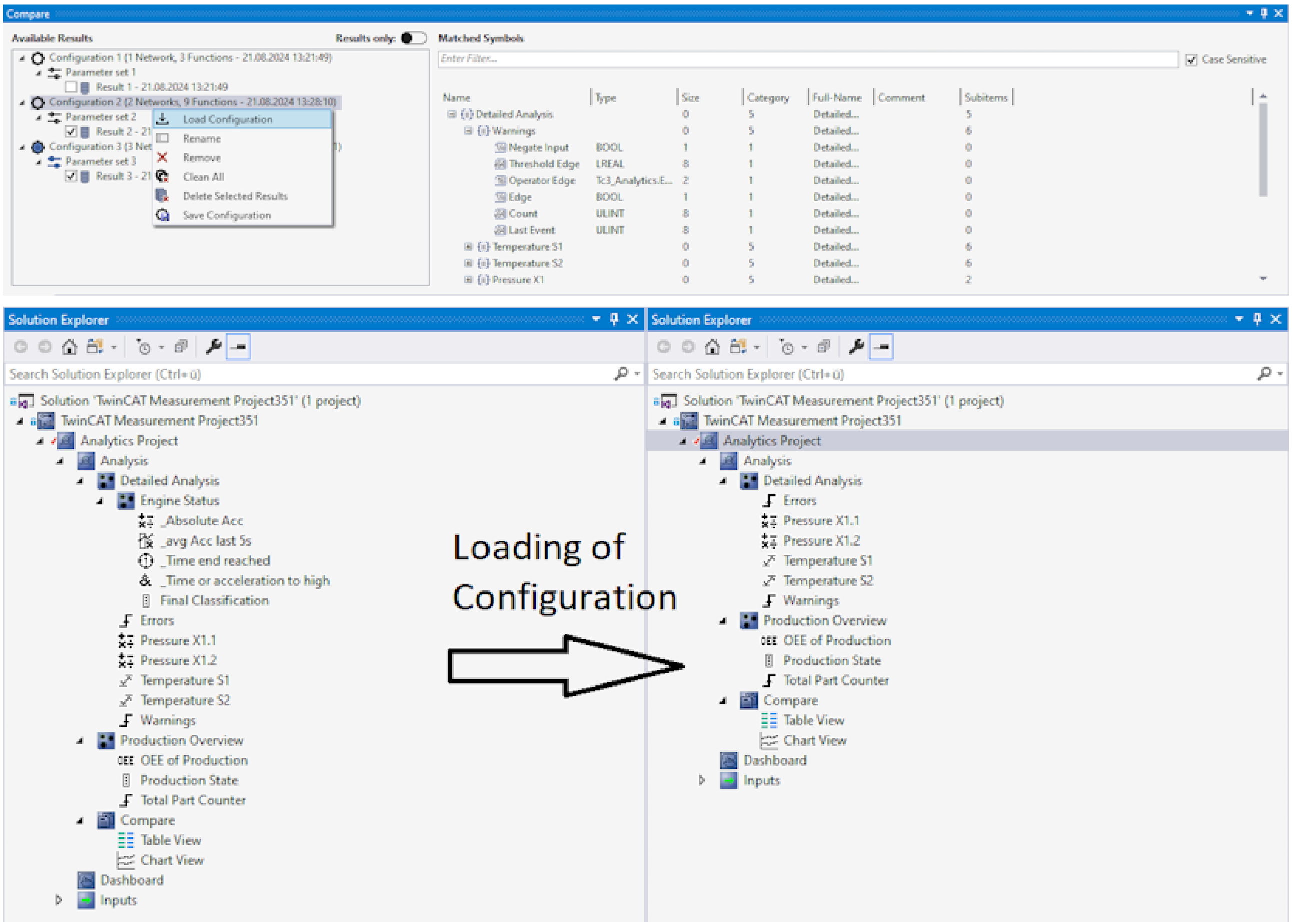
Records
Result 1 - 22.08.2024 07:25:40
22.08.2024 07:25:43.407 - 22.08.2024 07:25:51.664
Style
Data synchronization
Virtual Input Source.Variable 0
Time Offset: 00:00:00.000
Result 2 - 22.08.2024 07:25:50
22.08.2024 07:25:53.553 - 22.08.2024 07:26:01.810
Style
Data synchronization
Virtual Input Source.Variable 0
Time Offset: 00:00:00.000
Data synchronization
Virtual Input Source.Variables.fSine 0

Channel Visibility
Virtual Input Source.Variables.fSine

Records
Result 1 - 22.08.2024 07:25:40
22.08.2024 07:25:43.407 - 22.08.2024 07:25:51.664
Style
Data synchronization
Virtual Input Source.Variable 0
Time Offset: 00:00:00.317
Result 2 - 22.08.2024 07:25:50
22.08.2024 07:25:53.553 - 22.08.2024 07:26:01.810
Style
Data synchronization
Virtual Input Source.Variable 0
Time Offset: 00:00:00.172
Data synchronization
Virtual Input Source.Variables.fSine 0

6.10.1 Loading and saving project configurations

The history of the analysis runs is listed in the *Compare* window. The corresponding project status is saved for a configuration and can be loaded. To load the configuration, right-click on the desired configuration (or parameter set) and select the *Load Configuration* option in the context menu. This restores the old project status. The configuration icon then turns blue, indicating the last loaded configuration.



Name	Type	Size	Category	Full-Name	Comment	Subitems
⊟ {i} Detailed Analysis		0	5	Detailed...		5
⊟ {i} Warnings		0	5	Detailed...		6
Negate Input	BOOL	1	1	Detailed...		0
Threshold Edge	LREAL	8	1	Detailed...		0
Operator Edge	Tc3_Analytics.E...	2	1	Detailed...		0
Edge	BOOL	1	1	Detailed...		0
Count	ULINT	8	1	Detailed...		0
Last Event	ULINT	8	1	Detailed...		0
⊞ {i} Temperature S1		0	5	Detailed...		6
⊞ {i} Temperature S2		0	5	Detailed...		6
⊞ {i} Pressure X1		0	5	Detailed...		2

In addition to automatically saving a configuration at the start of an analysis, you can also save it manually. To do this, simply select *Save Configuration* from the context menu.

7 API

The TwinCAT Analytics Data Exchange API documented below can be used to feed data into or retrieve data from the Analytics Workflow. This allows you to connect external sources or, alternatively, to reuse the data from Analytics in other software components.

7.1 Data Exchange API

You can use the Analytics Data Exchange API to build your own analytics loggers or analytics runtimes in all supported programming and script languages.

A logger allows data from external sources to be stored or analyzed in Analytics. A runtime is programmed to use the Analytics data in other systems. Depending on the direction, the appropriate license is required. For the logger, it is the TF3500 Analytics Logger license. For a runtime implementation, it is TF3551 Analytics Runtime Base or alternatively TF3550 Analytics Runtime. If these products are already used on the systems, no further license is required for a separate programmatic implementation.

7.1.1 .NET

Installation

The following two components must be installed to use the Analytics Data Exchange API:

1. Microsoft Visual C++ Redistributable 2019 (x64/ x86) or higher. If this package is missing, errors occur when processing symbols.
2. TwinCAT 3 ADS

NuGet packages

The Analytics Data Exchange API is delivered as a NuGet package. Three different packages are available:

1. Beckhoff.TwinCAT.Analytics.DataExchange.Core (.Net-Core 3.1 or .Net-5 and higher)
2. Beckhoff.TwinCAT.Analytics.DataExchange.Framework (.Net-Framework 4.5.2 or higher)
3. Beckhoff.TwinCAT.Analytics.DataExchange (multi-target projects)

7.1.1.1 Mapping of the logger function

This documentation describes how to send or store data in Analytics format using the Analytics Data Exchange API.

License

A license for the TwinCAT 3 Analytics Logger is required for writing data in TwinCAT Analytics format. The Data Exchange API can be tested with a 7-day license.

Create and configure a data storage:

To send or save data in Analytics format, the first step is to create a builder to configure the data storage:

```
ISinkFactory factory = new SinkFactory();

IFileSinkBuilder fileBuilder = factory.ToFile("C:\Data\Demo");

IMqttSinkBuilder mqttBuilder = factory.ToMessageBroker("127.0.0.1", 1883);
```

There are currently two different builders available to either write to a file or send MQTT data. The builders are configured via a concatenation of method calls:

```
mqttBuilder.WithSymbol("Symbol1", TypeCode.Double).WithCompressionWidth(8)
```

The following methods are available for configuring the symbols:

General parameters	Description
WithCompression(…)	Sets the compression method for the data sink.
WithCompressionWidth(..)	Increasing the CompressionWidth will result in less compression, but faster processing. Values below 3 are not recommended. Default: 8
WithMaxSamplesPerChunk(…)	Maximum number of data in an MQTT message or in a file. Default: 32 for MQTT, 500 for file
WithMaxChunkSize(…)	Maximum data size of an MQTT message or file. Standard: not limited
WithCycleTime(…)	Cycle time of the data: 10 ms by default
Symbols:	
WithSymbol(…)	Adds a symbol. Optionally, an array length can be specified.
WithSymbols(Action)	Allows you to add several symbols at the same time.
WithSymbol<T>	Adds a symbol of the generic data type. The section Generic data types [▸ 529] contains information about the allowed data types.
MQTT only:	
ToTopic(…)	Topic over which the data is communicated.
File only:	
WithIndexing(…)	Distance between two indexed data points when using compression. Indexing increases performance when reading data points, but increases memory usage. 0 disables indexing and is default. The value should be set to at least 100 for activation.

Special parameters for MQTT communication are configured using the following methods:

WithTLS(…)	Configures the certificates to use TLS
WithCredentials(...)	User name and password for access to the message broker
WithTcpBufferSize(…)	Sets the size of the buffer for the TCP connection to the message broker. The default is 32 kB. If very large messages are sent, this value can be increased.
WithKeepAlivePeriod(…)	Set the KeepAlive time for the MQTT connection. (the default is 60 s)
WithCommunicationTimeout(…)	Timeout for TCP communication to the message broker. (the default is 3 s)
ToTopic(…) / FromTopic(…)	Topic over which the data is communicated.

The configuration of a data sink is completed by calling the build method, which returns the configured data sink. The data sink provides the following methods. If the data sink is no longer used, it is necessary to call the Dispose method to release all resources again.

InitWrite()	Initializes the MQTT connection or opens the file to be written. Also called automatically the first time values are written.
Write(…)	Adds a new value to the buffer.
Flush()	Sends the data stored in the buffer. Automatically called by Write. When MaxSamples or MaxSize is exceeded.
CompleteWrite()	Sends the data stored in the buffer and closes any open MQTT connections or file accesses.
Dispose()	Closes all open MQTT connections or file accesses without sending the buffer and releases all resources. The sink can no longer be used after that.

Generic data types:

The Analytics Data Exchange API supports only "unmanaged structs" as generic data types. This results in the following restrictions

1. Only structures or basic data types may be used within the structures.
2. Arrays within structures are only possible with "unsafe" code. (Fixed-Array-Size)
3. Only arrays of basic data types can be used. Arrays of structures are not possible.

Furthermore, only a sequential or explicit layout can be used for the structures (see https://docs.microsoft.com/de-de/dotnet/api/system.runtime.interopservices.structlayoutattribute?view=net-6.0)

1. An explicit layout should be used when writing data.
2. The layout must not contain padding (Pack = 1) when writing data
3. The padding must correspond to the data source when reading data (https://infosys.beckhoff.com/index.php?content=../content/1033/tc3_plc_intro/3539428491.html&id=)

Error messages

If errors occur while using the Data Exchange API, they are passed on to the user by exception. The following exceptions are used:

Class name	Superclass	Description
DataExchangeException	Exception	Superclass for all further exceptions within the API.
InvalidSymbolDefinitionException	DataExchangeException	The configured symbols are not correct.
SymbolNotFoundException	InvalidSymbolDefinitionException	A configured symbol could not be found in the data stream.
InvalidSymbolNameException	InvalidSymbolDefinitionException	The entered symbol name is not valid.
InvalidSymbolOffsetException	InvalidSymbolDefinitionException	The entered symbol offset is smaller than 0.
InvalidSymbolByteSizeException	InvalidSymbolDefinitionException	The entered size in bytes is less than or equal to 0.
InvalidSymbolTypeCodeException	InvalidSymbolDefinitionException	The TypeCode entered for a symbol is not valid.
InvalidSymbolRegexPatternException	InvalidSymbolDefinitionException	The entered regex pattern for a symbol is not valid.
InvalidSymbolFilterException	InvalidSymbolDefinitionException	The entered SymbolFilter for a regex pattern is not valid.
DataExchangeLicenseException	DataExchangeException	No valid license for the API could be found or TwinCAT is not in run mode.
DataExchangeTimeoutException	DataExchangeException	A time-out period has been exceeded.
QueueFullException	DataExchangeException	A queue for processing data is full because the processing was too slow.
FileSourceNotFoundException	DataExchangeExeption	No Analytics file could be found in the specified folder.
ValueExtractException	DataExchangeException	An error occurred while reading the values from the data stream. More detailed information is available in the inner exception.
DataExchangeMQTTException	DataExchangeException	An error has occurred with the MQTT connection.
BrokerConnectException	DataExchangeMQTTException	A connection to the broker could not be established.
TopicNotFoundException	DataExchangeMQTTException	The specified topic was not found at the broker.
InvalidMqttStreamFormat	DataExchangeMQTTException	The specified stream format is not correct.

Examples:

Samples of how to use the Analytics Data Exchange API can be found here:

https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12970252171.zip

7.1.1.2 Mapping of the runtime function

This documentation describes how to read data from the Analytics format using the Analytics Data Exchange API.

License

A license for the "TwinCAT 3 Analytics Runtime" or "TwinCAT 3 Analytics Runtime Base" is required to read data from the TwinCAT Analytics format. The Data Exchange API can be tested with a 7-day license.

Create and configure a data source:

The first step is to create and configure a Builder in order to read data from the Analytics format.

```
ISourceFactory factory = new SourceFactory();

IFileSourceBuilder fileBuilder = factory.FromFile("C:\Data\Demo");

IMqttLiveSourceBuilder mqttBuilder = factory.FromLiveStream("127.0.0.1", 1883);

IStorageProviderSourceBuilder FromStorageProvider("127.0.0.1", 1883);
```

There are currently three different Builders available:

1. IFileSourceBuilder: data from a file.
2. IMqttLiveSourceBuilder: data from an Analytics Logger data stream.
3. IStorageProviderSourceBuilder: recorded data of an Analytics Storage Provider.

The Builder is parameterized via a concatenation of method calls. The following options are available:

Symbols	Description
WithSymbol(…)	Adds an additional symbol for reading. Available symbols can be read via the TwinCAT Target Browser.
WithSymbols(...)	Allows you to add several symbols at the same time.
WithSymbols(pattern, …)	Adds multiple symbols using a regex expression. Additionally, RegexOptions and a SymbolFilter can be specified. SymbolFilter: Include Parent: uses the parent symbol even if child symbols apply. IncludeChilds: uses the child symbols even if the parent symbol applies. IncludeChilds or InlcudeParent must be set.
WithSymbol<T>()	Adds a symbol of the generic data type. The section Generic data types [▸ 533] contains information about the allowed data types.
MQTT only:	
FromTopic(…)	MQTT topic from which the data should be read.
WithDataTimeOut(…)	Timeout between two MQTT messages (default 500ms). -1ms means an infinite timeout.
WithSymbolTimeOut(…)	Timeout for receiving the symbol information from the message broker (default 1s). -1ms means an infinite timeout.
WithSortQueueSize(...)	Setting the size of the buffer for incoming messages. The buffer is also used to sort the messages. (Default: Min: 5 Max: 25)
WithStreamFormat(…)	Specifies the format of the received data. If nothing is specified, the format will be detected automatically. For automatic detection, the data stream must be online at startup.
Storage Provider and file:	
From(…)	Start of the data retrieval process.
To(…)	End of the data retrieval process.
StorageProvider only	
FromStorageProivder(…)	Specifies the storage provider at the broker used to retrieve the data.
FromHistoricalStream(…)	The ProviderId and the MainTopic are read from the specified topic, which otherwise must be passed via the FromStorageProvider method.
FromLayout(…)	Layout of the historical data stream.
FromOriginSystem(…)	System Id from which the data was originally sent.
FromOriginStreamTopic (…)	Topic via which the data was originally sent.
FromStreamAlias(…)	Alias under which the data stream is recorded and displayed in the TargetBrowser.

The methods FromLayout, FromOriginSystem, FromOriginStreamTopic, FromStreamAlias, are used to identify the correct data set. It is not mandatory to parameterize all of them, but it is recommended to specify as many as possible to avoid confusion of data sets. The easiest way to read this information is to use Target Browser. To do this, drag a symbol of the data set into a text editor (e.g. Notepad ++). You can find the necessary information in the Target Info section in the XML.

Special parameters for MQTT communication are configured using the following methods:

WithTLS(…)	Configures the certificates to use TLS
WithCredentials(...)	User name and password for access to the message broker
WithTcpBufferSize(…)	Sets the size of the buffer for the TCP connection to the message broker. The default is 32 kB. If very large messages are sent, this value can be increased.
WithKeepAlivePeriod(…)	Set the KeepAlive time for the MQTT connection. (the default is 60 s)
WithCommunicationTimeout(…)	Timeout for TCP communication to the message broker. (the default is 3 s)
ToTopic(…) / FromTopic(…)	Topic over which the data is communicated.

The configuration of a data source is completed by calling the build method, which returns the configured data source. The data source provides various methods. If a data source is no longer used, it is necessary to call the Dispose method to release all resources.

InitSymbols()	Initializes the ReadSymbols collection and resolves existing regex expressions.
ReadSamples(...)	Queries all data synchronously in the calling context.
ReadAsync(...)	Queries all data asynchronously. The ReadDelegate is called for each date.
Dispose()	Closes all open MQTT connections or file accesses and releases all resources. The data source can no longer be used after that.

Generic data types:

The Analytics Data Exchange API supports only "unmanaged structs" as generic data types. This results in the following restrictions

1. Only structures or basic data types may be used within the structures.
2. Arrays within structures are only possible with "unsafe" code. (Fixed-Array-Size)
3. Only arrays of basic data types can be used. Arrays of structures are not possible.

Furthermore, only a sequential or explicit layout can be used for the structures (see https://docs.microsoft.com/de-de/dotnet/api/system.runtime.interopservices.structlayoutattribute?view=net-6.0)

1. An explicit layout should be used when writing data.
2. The layout must not contain padding (Pack = 1) when writing data
3. The padding must correspond to the data source when reading data (https://infosys.beckhoff.com/index.php?content=../content/1033/tc3_plc_intro/3539428491.html&id=)

Structured data types:

The Analytics Data Exchange API automatically detects the data types of symbols if no generic data type is specified. If the symbol used is a structured data type, an IStructuredValue is returned. The values of the subsymbols can be queried from the IStructuredValue on the one hand. This can be converted directly into a JSON text on the other hand. The last option is to create an IStructedValueReader to parse through the entire symbol similar to an XMLReader.

```
void Read(ISample sample)
{
  foreach(var value in sample.Values)
  {
    if(value is IStructuredValue stValue)
    {
      //Values of structured DataTypes without generic DataType could be proccessed here
      var subValue = value["SubValues1"]; //SubValues1 is a known SubElement of the structured Value

      var jsonString = stValue.ToJsonString(); //Convert the value to JSON

      IStructuredValueReader reader stValue.GetReader(); //Use reader to parse the value iterative
    }
  }
}
```

Error messages

If errors occur while using the Data Exchange API, they are passed on to the user by exception. The following exceptions are used:

Class name	Superclass	Description
DataExchangeException	Exception	Superclass for all further exceptions within the API.
InvalidSymbolDefinitionException	DataExchangeException	The configured symbols are not correct.
SymbolNotFoundException	InvalidSymbolDefinitionException	A configured symbol could not be found in the data stream.
InvalidSymbolNameException	InvalidSymbolDefinitionException	The entered symbol name is not valid.
InvalidSymbolOffsetException	InvalidSymbolDefinitionException	The entered symbol offset is smaller than 0.
InvalidSymbolByteSizeException	InvalidSymbolDefinitionException	The entered size in bytes is less than or equal to 0.
InvalidSymbolTypeCodeException	InvalidSymbolDefinitionException	The TypeCode entered for a symbol is not valid.
InvalidSymbolRegexPatternException	InvalidSymbolDefinitionException	The entered regex pattern for a symbol is not valid.
InvalidSymbolFilterException	InvalidSymbolDefinitionException	The entered SymbolFilter for a regex pattern is not valid.
DataExchangeLicenseException	DataExchangeException	No valid license for the API could be found or TwinCAT is not in run mode.
DataExchangeTimeoutException	DataExchangeException	A time-out period has been exceeded.
QueueFullException	DataExchangeException	A queue for processing data is full because the processing was too slow.
FileSourceNotFoundException	DataExchangeExeption	No Analytics file could be found in the specified folder.
ValueExtractException	DataExchangeException	An error occurred while reading the values from the data stream. More detailed information is available in the inner exception.
DataExchangeMQTTException	DataExchangeException	An error has occurred with the MQTT connection.
BrokerConnectException	DataExchangeMQTTException	A connection to the broker could not be established.
TopicNotFoundException	DataExchangeMQTTException	The specified topic was not found at the broker.
InvalidMqttStreamFormat	DataExchangeMQTTException	The specified stream format is not correct.

Examples:

Samples of how to use the Analytics Data Exchange API can be found here:

https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12970252171.zip

7.1.2 Python

Installation

The following two components must be installed to use the Analytics Data Exchange API:

1. Microsoft Visual C++ Redistributable 2019 (x64/ x86) or higher. If this package is missing, errors occur when processing symbols.
2. TwinCAT 3 ADS

7.1.2.1 Mapping of the logger function

This documentation describes how to send or store data in Analytics format using the Analytics Data Exchange API.

License

A license for the TwinCAT 3 Analytics Logger is required for writing data in TwinCAT Analytics format. The Data Exchange API can be tested with a 7-day license.

Create and configure a data storage:

To send or save data in Analytics format, the first step is to create a builder to configure the data storage:

```
Factory = SinkFactory()

Builder = Factory.ToFile(FilePath)

Builder = Factory.ToMessageBroker(Broker, 1883)
```

The different builders provide different methods for configuration. Symbols can be added using the "WithSymbol" method. Any number of symbols can be used. Symbol implementations are already available for most basic data types. Structured data types and arrays can also be implemented.

```
class MyStruct(Struct):

def __init__(self, name : str):
self.b = Bool("b")
self.i64 = Int64("i64")
self.f64 = Real64("f64")
super().__init__(name)

TestValInt = Int32("i32")
TestValFloat = Real32("f32")
TestValArray = Array("arrf64", Real64(), [10])
TestValStruct = MyStruct("myStruct")
Builder.WithSymbol(TestValInt)
Builder.WithSymbol(TestValFloat)
Builder.WithSymbol(TestValArray)
Builder.WithSymbol(TestValStruct)
```

Once the builder is fully configured, a new sink is created using the Build method, which is initialized with InitWrite. The current value of the symbols is then written using the Write method. The values of the symbols can be updated between several calls of the Write method. Once all the data has been written, the write process is ended with CompleteWrite:

```
SampleCount = 32

TestValInt.Value = 1000
TestValFloat.Value = 0.25
Sink.Write()
for i in range(0, SampleCount):
TestValInt.Value += 1
TestValFloat.Value += 0.25
Sink.Write()
Sink.CompleteWrite()
```

Sample:

You can find the complete sample here: https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/17763893899.zip

7.1.2.2 Mapping of the runtime function

This documentation describes how to read data from the Analytics format using the Analytics Data Exchange API.

License

A license for the "TwinCAT 3 Analytics Runtime" or "TwinCAT 3 Analytics Runtime Base" is required to read data from the TwinCAT Analytics format. The Data Exchange API can be tested with a 7-day license.

Create and configure a data source:

```
Factory = SourceFactory()

Builder = Factory.FromLiveStream(Broker, 1883)

Builder = Factory.FromFile(FilePath)
```

The different builders provide different methods for configuration. Symbols can be added using the "WithSymbol", "WithName" or "WithRegex" methods:

```
TestValue = UInt64("Variables.nGrowSteady")
Builder.WithSymbol(TestValue)

Builder.WithName("TestValues.nGrowSteady")
Builder.WithRegex("Variables.f", 0, RegexSymbolFilter.IncludeChilds)
```

Once the builder is fully configured, a new sink is created using the Build method. The data query starts by calling the Read method. A Callback method is passed as a parameter. If the sample does not contain any symbols, all data has already been read out. If the Callback method returns a value other than 0, the reading is aborted.

```
def Callback(sample : ISample) -> int:
if sample.Symbols() is not None:
print(f'Stamp = {sample.Timestamp()}, {sample.Symbols()[0].SymbolName()} = {sample.Symbols()
[0].Value}')
Values.append(sample.Symbols()[0].Value)
Stamps.append(DateTimeToFileTime(sample.Timestamp()))
if len(Stamps) >= SampleCount:
return False
return True

Source.Read(Callback)
```

Sample:

The complete sample can be found here: https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/17763978891.zip

8 Samples

8.1 Pick-and-place application

The following sample shows an analysis of an XTS application together with a pick-and-place robot. It offers you two options:

1. You can use the sample data and build your own TwinCAT Analytics project.
2. You can use the ready-to-use TwinCAT Analytics project in order to understand how TwinCAT Analytics works.

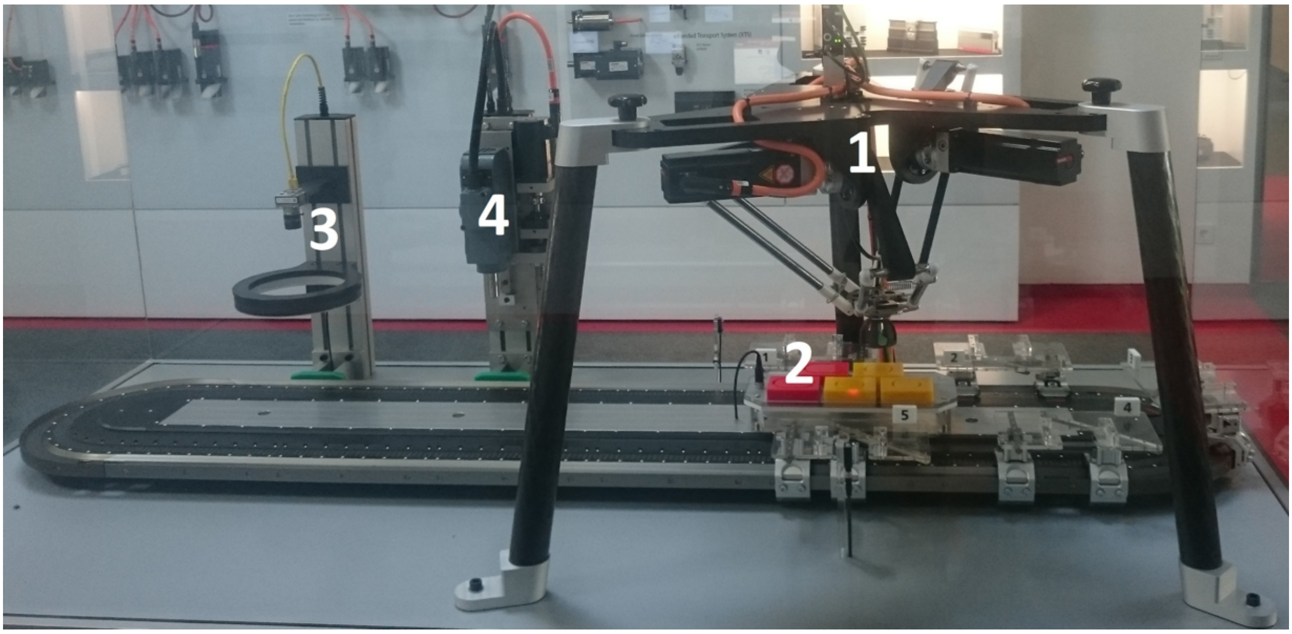
In both cases, it makes sense to understand the machine application that provides the data for the analysis. The application sequence is shown in the video:

video::https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/6815846411.mp4.

Application description

This application example describes a simulation for the processing of production goods. The application transports workpieces from a store to a drill via a Beckhoff XTS system. Depending on the color of the workpiece, a different pattern is milled into each part. The XTS system shown has a length of 4000 mm. It includes ten movers, with two movers together carrying one tray. A pick-and-place robot (1) is positioned at the workpiece store (2). Two red and three yellow parts are located in five store positions, which the picker can pick up or set down via a vacuum pump. On the opposite side of the XTS there is a vision system (3) for identifying the workpiece color and a miller (4) for the specific milling contour.

The trays drive to the workpiece store. The picker picks up the parts and loads or unloads the trays. The tray then drives to the scanner, where the color of the workpiece is scanned and communicated to the downstream miller. Depending on the color, the workpieces are processed differently: a circle is milled into yellow parts, and an oval shape is milled into red parts. The workpiece then drives back to the store and is stored there by the picker. The tray is loaded with another part and drives onwards.



Variable description

The following variables are recorded by the TwinCAT Analytics Logger in the attached Analytics File Format. You can use the data for your own Analytics configurator project or for the default configuration in the tnzip file provided. The meaning and function of each variable are shown in the following table.

Variable name	Data Type	Description
bMillerMilling	BOOL	Information about the activity of the drill: 1 if the drill is active and 0 if the drill is inactive.
bMillerMovingZ	BOOL	Information about the movement of the drill in z-direction: is 1 if the drill moves in z-direction and 0 if it does not move in z-direction.
bMillerSpindleRotation	BOOL	Information about the activity of the drill needle: is 1 when the needle is moving and the drill is active, and 0 when it is not moving.
bNewWorkpieceColorScanned	BOOL	Registers if a new color has been scanned (= 1), and then will inform the drill. If the color is changing the drill pattern has to be changed. When the color is still the same (=0) the drill pattern will not change.
bPickerHoldingWorkpiece	BOOL	Is 1, when the picker is holding a workpiece and 0, when not.
bPickerInStorage1Position	BOOL	Is 1, when the picker is in storage position 1.
bPickerInStorage2Position	BOOL	Is 1, when the picker is in storage position 2.
bPickerInStorage3Position	BOOL	Is 1, when the picker is in storage position 3.
bPickerInStorage4Position	BOOL	Is 1, when the picker is in storage position 4.
bPickerInStorage5Position	BOOL	Is 1, when the picker is in storage position 5.
bPickerInXtsPosition	BOOL	Is 1, when the picker is XTS position, which means over the XTS.
bPickerMoving	BOOL	Information, when the picker is moving (=1) or is inactive (=0).
fMillerPositionZ	LREAL	Indicates the z-position of the drill. At a value of 224 the drill is in drill position, at 245 he is inactive.
fMillerSpindleRotation	LREAL	Indicates the amount of revolutions per minute of the drill needle.
nColorOfLastScannedWorkpiece	INT	Registers the color of the last scanned workpiece. 5 → yellow workpiece 4 → red workpiece
nCurrentProductionTime	ULINT (64-bit)	Registers the current production time.
nVacuumPressure	DINT	The picker grasps the workpieces via compressed air. This variable indicates the current pressure in mbar.
stMachineVibrations aMachineVibrations[1...10]	STRUCT with ARRAY of LREAL	The structure stMachine Vibrations indicates the vibrations of the machine base in the unit m/s^2 with oversampling.
stPickerPosition aPickerPosition[1...3]	STRUCT with ARRAY of LREAL	The structure stPickerPosition contains an array with three values. The values indicate the x-/y- and z-position of the picker.
stTableInMillingPosition aTableInMillingPosition[1...5]	STRUCT with ARRAY of BOOL	The stTableInMillingPosition structure contains an array with five values, each of which represents a tray: value 1 for tray 1, value 2 for tray 2... if the value is 1, the tray is in the drill position.
stTableInStoragePosition aTableInStoragePosition[1...5]	STRUCT with ARRAY of BOOL	The stTableInStoragePosition structure contains an array with five values, each of which represents a tray: value 1 for tray 1, value 2 for tray 2... if the value is 1, the tray is in the workpiece store.

Variable name	Data Type	Description
stWorkpieceColors aWorkpieceColors[1...5]	STRUCT with ARRAY of INT	The structure stWorkpieceColors contains an array with five values, each value stands for one workpiece and indicates its color (as Int value): Workpiece 1: yellow Workpiece 2: yellow Workpiece 3: red Workpiece 4: red Workpiece 5: yellow
stWorkpiecePositions aWorkpiecePosition[1...5]	STRUCT with ARRAY of INT	The structure stWorkpiecePositions contains an array with five values for each workpiece. The Int value indicates the current position of the workpiece: 1: storage 1 2: storage 2 3: storage 3 4: storage 4 5: storage 5 6: tray 1 7: tray 2 8: tray 3 9: tray 4 10: tray 5 11: picker
stXtsMoverAcceleration aXtsMoverAcceleration[1...10]	STRUCT with ARRAY of LREAL	The structure stXtsMoverAcceleration contains an array with ten values for the movers 1-10 and indicates the acceleration of the particular mover.
stXtsMoverMovingNegative aXtsMoverMovingNegative[1...10]	STRUCT with ARRAY of BOOL	The structure stXtsMoverMovingNegative contains an array with ten values for the movers 1-10 and is 1, when the particular mover is moving backward.
stXtsMoverMovingPositive aXtsMoverMovingPositive [1...10]	STRUCT with ARRAY of BOOL	The structure stXtsMoverMovingPositive contains an array with ten values for the movers 1-10 and is 1, when the particular mover is moving forward.
stXtsMoverPositionsX aXtsMoverPositionsX[1...10]	STRUCT with ARRAY of LREAL	The structure stXtsMoverPositionsX contains an array with ten values for movers 1-10. The value indicates the x-position of the respective mover on the XTS.
stXtsMoverVelocity aXtsMoverVelocity[1...10]	STRUCT with ARRAY of LREAL	The structure stXtsMoverAcceleration contains an array with ten values for the movers 1-10 and indicates the velocity of the particular mover.

How to get the data

In the download area you will find a 30-minute recording of the XTS system that can be used for further analysis. To access these data, you must make the data available in a historical stream using the Analytics Storage Provider. A message broker should be made available for this. For the samples prepared, we assume that all components are running on the local system.

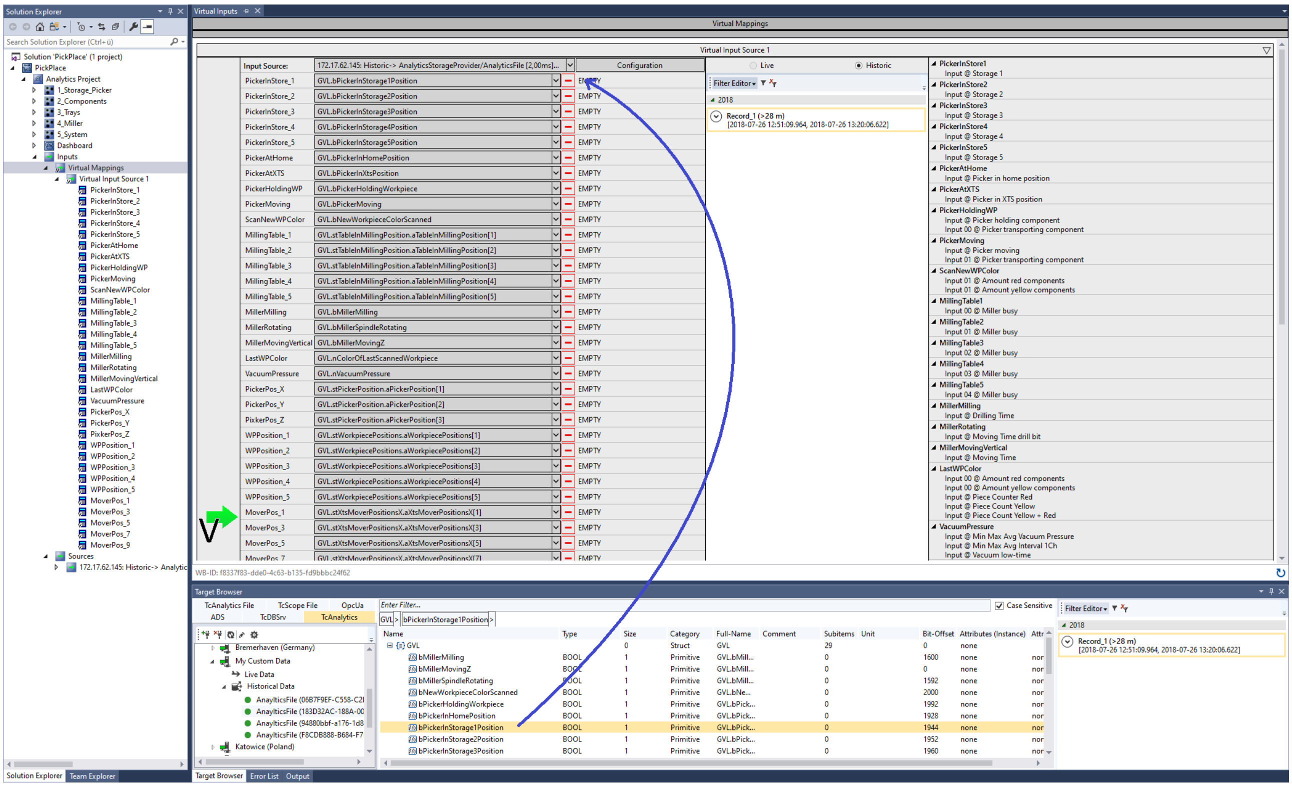
Install the Analytics Storage Provider. Follow the instructions given.

1. Import your Analytics file data into the Storage Provider (see: Importing Analytics Files).
2. Use the Analytics Storage Provider Configurator to start the Storage Provider Windows service.
3. Open the TwinCAT Target Browser in the TcXaeShell or in Visual Studio®. Select the TcAnalytics tab and your message broker. In the tree on the left-hand side you will now find an entry for the historical data, which will be displayed in the symbol list on the right-hand side when selected.
4. Connect individual symbols to the inputs of their Analytics algorithms.
5. Start the analysis.

Analysis with the Analytics Workbench

The analysis that is the subject of the TwinCAT Analytics project from the download area is explained below. Download and open the finished project.

Select the Virtual Mappings node and also browse for the topic specified by your storage provider in the Target Browser. Select the data and link them to the appropriate virtual inputs in the editor.



Input Source:	172.17.62.145: Historic-> AnalyticsStorageProvider/AnalyticsFile [2,00ms]...	Configuration
PickerInStore_1	GVL.bPickerInStorage1Position	EMPTY
PickerInStore_2	GVL.bPickerInStorage2Position	EMPTY
PickerInStore_3	GVL.bPickerInStorage3Position	EMPTY
PickerInStore_4	GVL.bPickerInStorage4Position	EMPTY
PickerInStore_5	GVL.bPickerInStorage5Position	EMPTY
PickerAtHome	GVL.bPickerInHomePosition	EMPTY
PickerAtXTS	GVL.bPickerInXtsPosition	EMPTY
PickerHoldingWP	GVL.bPickerHoldingWorkpiece	EMPTY
PickerMoving	GVL.bPickerMoving	EMPTY
ScanNewWPColor	GVL.bNewWorkpieceColorScanned	EMPTY
MillingTable_1	GVL.stTableInMillingPosition.aTableInMillingPosition[1]	EMPTY
MillingTable_2	GVL.stTableInMillingPosition.aTableInMillingPosition[2]	EMPTY
MillingTable_3	GVL.stTableInMillingPosition.aTableInMillingPosition[3]	EMPTY
MillingTable_4	GVL.stTableInMillingPosition.aTableInMillingPosition[4]	EMPTY
MillingTable_5	GVL.stTableInMillingPosition.aTableInMillingPosition[5]	EMPTY
MillerMilling	GVL.bMillerMilling	EMPTY
MillerRotating	GVL.bMillerSpindleRotating	EMPTY
MillerMovingVertical	GVL.bMillerMovingZ	EMPTY
LastWPColor	GVL.nColorOfLastScannedWorkpiece	EMPTY
VacuumPressure	GVL.nVacuumPressure	EMPTY
PickerPos_X	GVL.stPickerPosition.aPickerPosition[1]	EMPTY
PickerPos_Y	GVL.stPickerPosition.aPickerPosition[2]	EMPTY
PixkerPos_Z	GVL.stPickerPosition.aPickerPosition[3]	EMPTY
WPPosition_1	GVL.stWorkpiecePositions.aWorkpiecePositions[1]	EMPTY
WPPosition_2	GVL.stWorkpiecePositions.aWorkpiecePositions[2]	EMPTY
WPPosition_3	GVL.stWorkpiecePositions.aWorkpiecePositions[3]	EMPTY
WPPosition_4	GVL.stWorkpiecePositions.aWorkpiecePositions[4]	EMPTY
WPPosition_5	GVL.stWorkpiecePositions.aWorkpiecePositions[5]	EMPTY
MoverPos_1	GVL.stXtsMoverPositionsX.aXtsMoverPositionsX[1]	EMPTY
MoverPos_3	GVL.stXtsMoverPositionsX.aXtsMoverPositionsX[3]	EMPTY
MoverPos_5	GVL.stXtsMoverPositionsX.aXtsMoverPositionsX[5]	EMPTY
MoverPos_7	GVL.stXtsMoverPositionsX.aXtsMoverPositionsX[7]	EMPTY

2018
Record_1 (>28 m)
[2018-07-26 12:51:09.964, 2018-07-26 13:20:06.622]

- PickerInStore1
  - Input @ Storage 1
- PickerInStore2
  - Input @ Storage 2
- PickerInStore3
  - Input @ Storage 3
- PickerInStore4
  - Input @ Storage 4
- PickerInStore5
  - Input @ Storage 5
- PickerAtHome
  - Input @ Picker in home position
- PickerAtXTS
  - Input @ Picker in XTS position
- PickerHoldingWP
  - Input @ Picker holding component
  - Input 00 @ Picker transporting component
- PickerMoving
  - Input @ Picker moving
  - Input 01 @ Picker transporting component
- ScanNewWPColor
  - Input 01 @ Amount red components
  - Input 01 @ Amount yellow components
- MillingTable1
  - Input 00 @ Miller busy
- MillingTable2
  - Input 01 @ Miller busy
- MillingTable3
  - Input 02 @ Miller busy
- MillingTable4
  - Input 03 @ Miller busy
- MillingTable5
  - Input 04 @ Miller busy
- MillerMilling
  - Input @ Drilling Time
- MillerRotating
  - Input @ Moving Time drill bit
- MillerMovingVertical
  - Input @ Moving Time
- LastWPColor
  - Input 00 @ Amount red components
  - Input 00 @ Amount yellow components
  - Input @ Piece Counter Red
  - Input @ Piece Count Yellow
  - Input @ Piece Count Yellow + Red
- VacuumPressure
  - Input @ Min Max Avg Vacuum Pressure
  - Input @ Min Max Avg Interval 1Ch
  - Input @ Vacuum low-time

WB-ID: f8337f83-dde0-4c63-b135-fd9bbbc24f62

Target Browser: TcAnalytics File | TcScope File | OpcUa | ADS | TcDBSrv | TcAnalytics

- Bremerhaven (Germany)
- My Custom Data
  - Live Data
  - Historical Data
    - AnalyticsFile (06B7F9EF-C558-C2I
    - AnalyticsFile (183D32AC-188A-00
    - AnalyticsFile (94880bbf-a176-1d8
    - AnalyticsFile (F8CDB888-B684-F7
- Katowice (Poland)

Enter Filter... ☑ Case Sensitive

GVL > bPickerInStorage1Position >

Name	Type	Size	Category	Full-Name	Comment	Subitems	Unit	Bit-Offset	Attributes (Instance)	Attr
GVL		0	Struct	GVL		29		0	none	
bMillerMilling	BOOL	1	Primitive	GVL.bMill...		0		1600	none	nor
bMillerMovingZ	BOOL	1	Primitive	GVL.bMill...		0		0	none	nor
bMillerSpindleRotating	BOOL	1	Primitive	GVL.bMill...		0		1592	none	nor
bNewWorkpieceColorScanned	BOOL	1	Primitive	GVL.bNe...		0		2000	none	nor
bPickerHoldingWorkpiece	BOOL	1	Primitive	GVL.bPick...		0		1992	none	nor
bPickerInHomePosition	BOOL	1	Primitive	GVL.bPick...		0		1928	none	nor
bPickerInStorage1Position	BOOL	1	Primitive	GVL.bPick...		0		1944	none	nor
bPickerInStorage2Position	BOOL	1	Primitive	GVL.bPick...		0		1952	none	nor
bPickerInStorage3Position	BOOL	1	Primitive	GVL.bPick...		0		1960	none	nor

2018
Record_1 (>28 m)
[2018-07-26 12:51:09.964, 2018-07-26 13:20:06.622]

Description of the Analytics project

Network name	Used variables	Used Algorithms	Description
1_Storage_Picker	bPickerinStoragePosition, bPickerHoldingWorkpiece, bPickerInXtsPosition, bPickerInHomePosition, bPickerMoving, stPickerPosition	Edge Counter, Math Operation, Edge Counter OnOff, Logic Operation Counter, Numerical Compare, Min Max Avg	Picker: This network analyses the picker actions. It counts the quantities if the picker holds a workpiece, is in starting position or in XTS position. It is also analyzed how often the picker is transporting a workpiece, how often the picker is loading a mover as well as its xyz-position. How often does the picker load a workpiece? Storage: This network analyses the storage use of the five storage positions next to the picker. The quantities and proportions of storage uses are counted. Is each storage used equally?
2_Components	bNewWorkpieceColorScanned, bMillerMilling, nColorOfLastScannedWorkpiece, stWorkpiecePosition	Logic Operation Counter, Threshold String Classificator, Histogram, Trend Line	This network analyzes the workpieces themselves. How many workpieces of each color are processed at the scanner and the miller and the current position of each workpiece can be detected. The histogram algorithms show the distribution of the workpiece positions. How many red workpieces were processed?
3_Mover	stTableInMillingPosition, sXtsMoverPosition	XTS Velocity Analysis, XTS Distance Integrator, Min Max Avg Interval, Math Operation, Threshold Classificator, Trend Line, Edge Counter, Lifecycle Analysis, Logic Operation Counter	Distance: This network analyses the distance travelled of each tray and calculates the overall distance of all movers. As the length of the XTS system is known (4000 mm), the average amount of rounds for each tray can be derived. The Threshold Analysis gives a warning after 40 rounds and an alarm after 45 rounds. Which distance did all trays travel in total? Tray position: This network analyses how often the trays are stopping at the storage or at the miller and stops how long it takes a tray to move one round. How often did all trays stop at the miller? Velocity: This network analyses the current velocity of the trays as well as the minimum and maximum velocity in the last 20 seconds. The trays can move backwards. Therefore, the minimum velocity can be negative. What is the maximum velocity you can read?

Network name	Used variables	Used Algorithms	Description
4_Miller	bMillerMilling, bMillerSpindleRotating, bPickerMoving	Event Timing Analysis, Lifetime Analysis	This network performs the analysis for the miller. Drilling time: time the miller is active. Drill bit movement time: time during which the drill bit is active. Drill movement time: time during which the drill is active. For which of the three actions did the miller spend the most time?
5_System	bNewWorkpieceColorScanned, bMillerMilling, nVacuumPressure, stMachineVibrations	Min Max Avg, Min Max Avg Interval, Timing Analysis, Math Operation Lifecycle Analysis, Threshold Classificator, Continuous Piece Counter	Productivity: The productivity is analyzed by counting the processed workpieces per 20 seconds and comparing it to previous intervals. Based on the maximum count the productivity of the following intervals is derived. What is the maximum number of workpieces in an interval? Technical components: This network analyzes the vacuum pressure of the picker and issues a warning or alarm message when the life cycle of the vacuum suction unit falls below defined threshold values. How long does it take to get a warning?

PLC code generation and HMI

After completion of the analysis in the Analytics Workbench, you can generate PLC code and an HMI dashboard (see also: 24h Analytics application [▶ 33]):

1. Double-clicking the project node opens the overview page. Select **deploy Runtime** there

2. The **Deploy Wizard** opens, which guides you step by step. If no individual settings are desired, you can simply proceed.

3. Optional: on the PLC tab, select a CPU core on which the analysis should be executed. An isolated core must be selected for execution in a virtual machine.

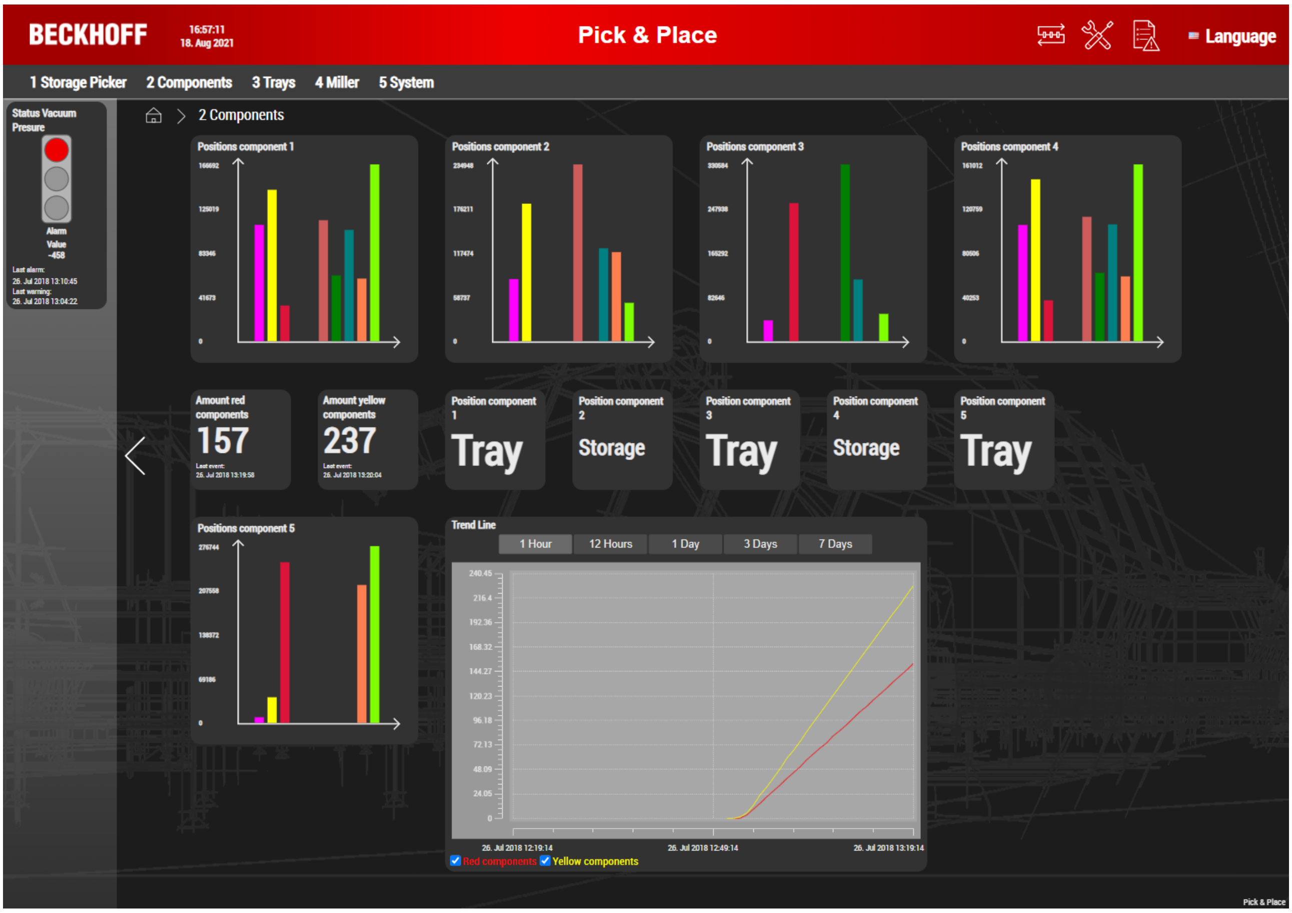
4. On the HMI Dashboard tab, enable generation. Adapt the design to suit your taste.

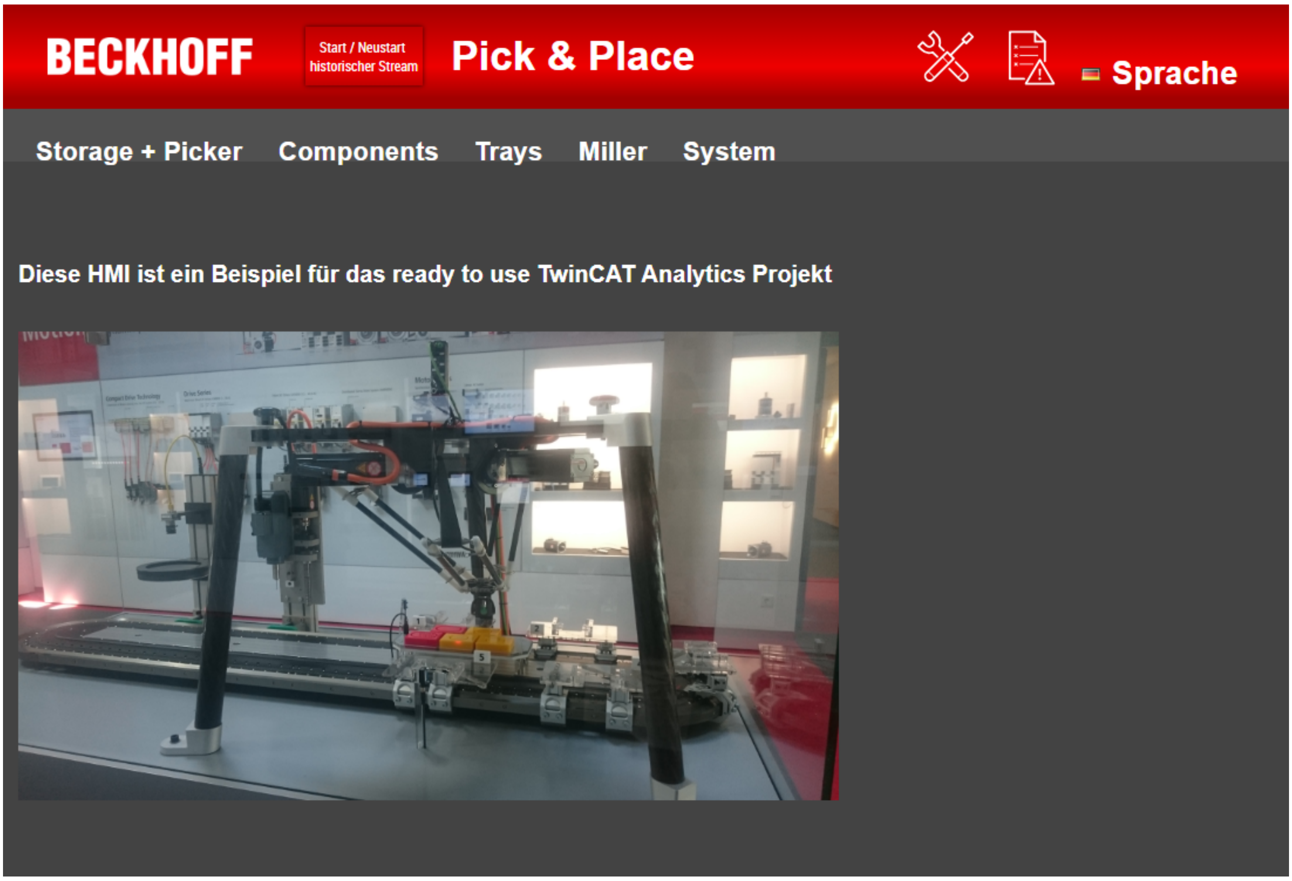
5. After subsequent confirmation via the **Deploy** button, a new development environment opens. A TwinCAT XAE project and a TwinCAT HMI project are automatically generated.

6. At the end of the generation, a dashboard opens in the browser. You can observe your original networks via the navigation.

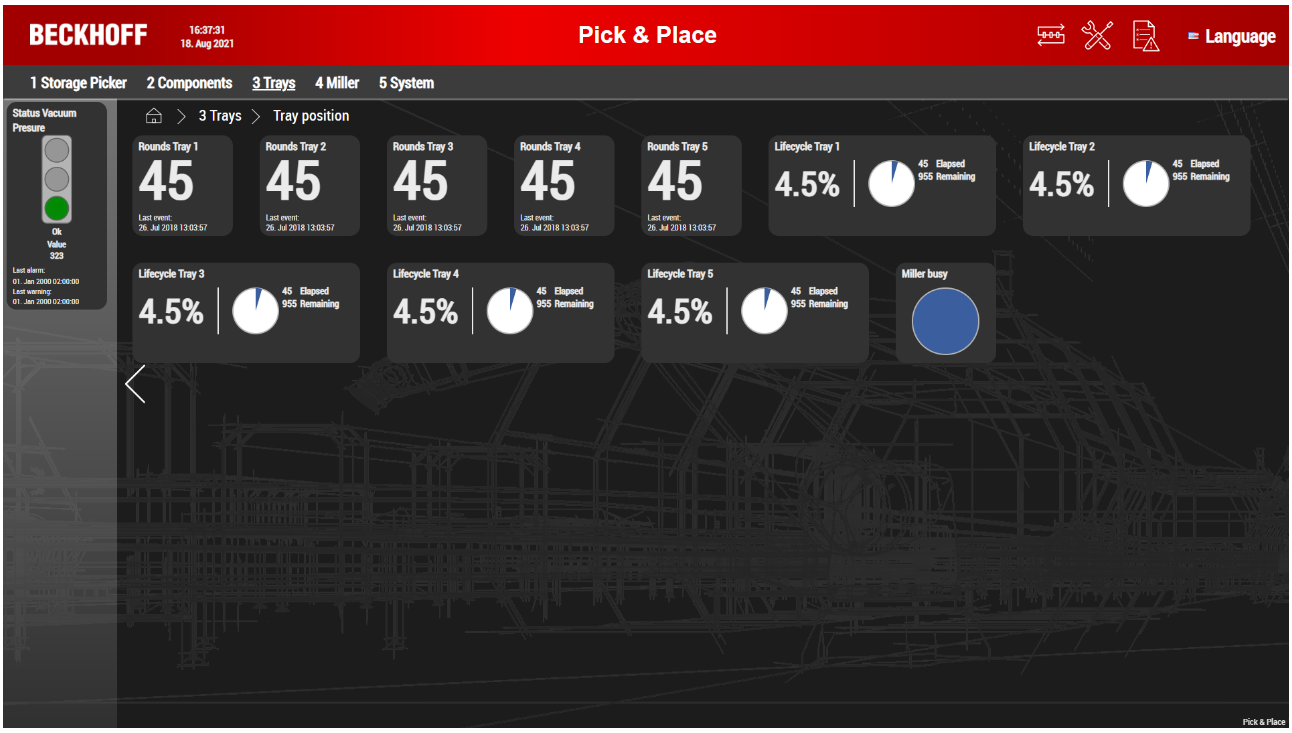
7. You have to start the analysis in order to analyze your historical data. To do this, open the corresponding option element via the third icon from the right at the top.

8. Select the time range of your recorded data that is to be analyzed and start the recording in order to track the analysis results.





Diese HMI ist ein Beispiel für das ready to use TwinCAT Analytics Projekt



Download the finished PLC code generation

If you only want to have a look at a ready generated Analytics PLC project with the corresponding HMI, there is also a download available for this (Sample_Analysis_PickNPlace_PLC). In order for this project to be executable, the main topic and the identification GUID of the storage provider have to be adapted in the code. Appropriate warnings are output, allowing you to navigate conveniently to the appropriate location within TwinCAT, where you will also find indications where the information is available.

Downloads

- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/6925371147.zip
- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/8817652491.zip
- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/8817654667.zip

8.2 Pick-and-place application: DenStream

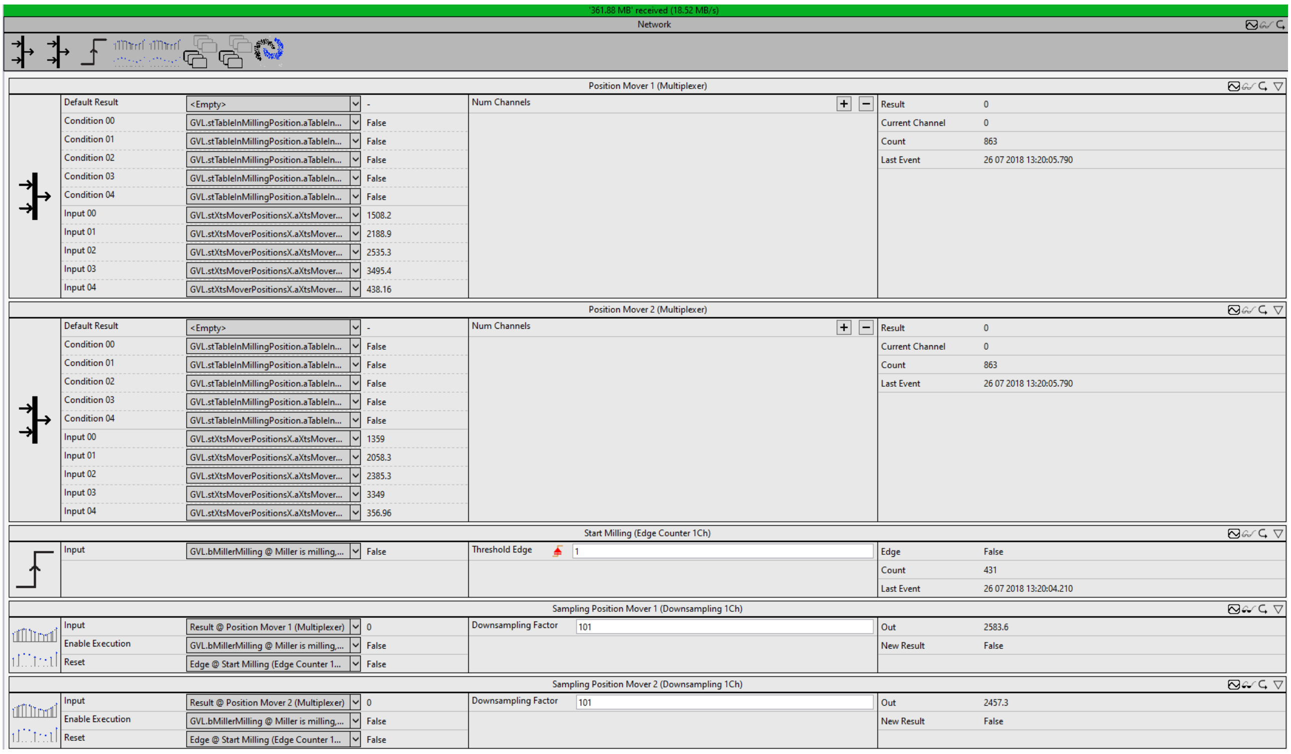
The application described below is an exemplary application of the DenStream function block.

The sample is based on data from the pick-and-place application. For this reason, a good understanding of the pick-and-place sample is a good basis for understanding this application. Data access is performed in the same way.

The aim of the application is to divide the milled shapes into two groups, oval (red) and round (yellow), based on the movement performed during the milling process.

The figures below show the individual function blocks of the application.

The trays for the workpieces are fixed with two movers each on the XTS track. Therefore, there are two position values per tray, which can be used to map the milled shape in a Scope project. In this case, one of the positions is sufficient to determine the cluster that is sampled at seven different times during the milling process. These seven values are then processed at the same time as individual inputs in the DenStream function block and form the seven-dimensional feature space.



'361.88 MB' received (18.52 MB/s)

Network

Position Mover 1 (Multiplexer)

Default Result	<Empty>	-
Condition 00	GVL.stTableInMillingPosition.aTableIn...	False
Condition 01	GVL.stTableInMillingPosition.aTableIn...	False
Condition 02	GVL.stTableInMillingPosition.aTableIn...	False
Condition 03	GVL.stTableInMillingPosition.aTableIn...	False
Condition 04	GVL.stTableInMillingPosition.aTableIn...	False
Input 00	GVL.stXtsMoverPositionsX.aXtsMover...	1508.2
Input 01	GVL.stXtsMoverPositionsX.aXtsMover...	2188.9
Input 02	GVL.stXtsMoverPositionsX.aXtsMover...	2535.3
Input 03	GVL.stXtsMoverPositionsX.aXtsMover...	3495.4
Input 04	GVL.stXtsMoverPositionsX.aXtsMover...	438.16

Num Channels

Result	0
Current Channel	0
Count	863
Last Event	26 07 2018 13:20:05.790

Position Mover 2 (Multiplexer)

Default Result	<Empty>	-
Condition 00	GVL.stTableInMillingPosition.aTableIn...	False
Condition 01	GVL.stTableInMillingPosition.aTableIn...	False
Condition 02	GVL.stTableInMillingPosition.aTableIn...	False
Condition 03	GVL.stTableInMillingPosition.aTableIn...	False
Condition 04	GVL.stTableInMillingPosition.aTableIn...	False
Input 00	GVL.stXtsMoverPositionsX.aXtsMover...	1359
Input 01	GVL.stXtsMoverPositionsX.aXtsMover...	2058.3
Input 02	GVL.stXtsMoverPositionsX.aXtsMover...	2385.3
Input 03	GVL.stXtsMoverPositionsX.aXtsMover...	3349
Input 04	GVL.stXtsMoverPositionsX.aXtsMover...	356.96

Num Channels

Result	0
Current Channel	0
Count	863
Last Event	26 07 2018 13:20:05.790

Start Milling (Edge Counter 1Ch)

Input	GVL.bMillerMilling @ Miller is milling,...	False

Threshold Edge: 1

Edge	False
Count	431
Last Event	26 07 2018 13:20:04.210

Sampling Position Mover 1 (Downsampling 1Ch)

Input	Result @ Position Mover 1 (Multiplexer)	0
Enable Execution	GVL.bMillerMilling @ Miller is milling,...	False
Reset	Edge @ Start Milling (Edge Counter 1...	False

Downsampling Factor: 101

Out	2583.6
New Result	False

Sampling Position Mover 2 (Downsampling 1Ch)

Input	Result @ Position Mover 2 (Multiplexer)	0
Enable Execution	GVL.bMillerMilling @ Miller is milling,...	False
Reset	Edge @ Start Milling (Edge Counter 1...	False

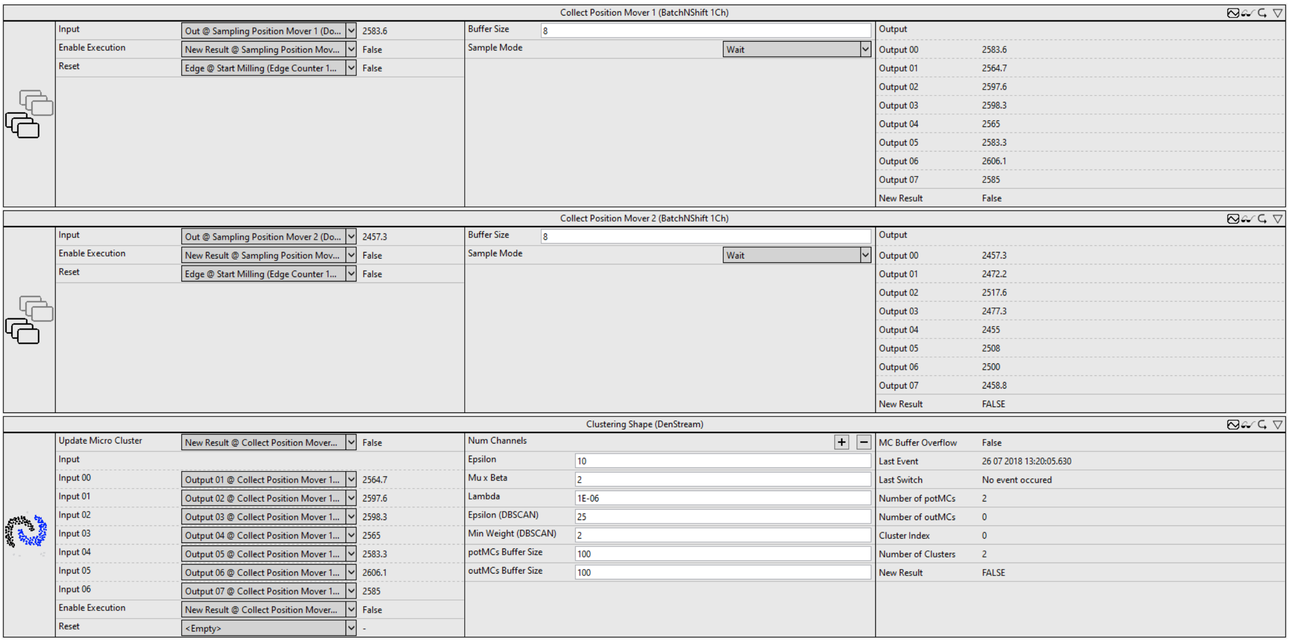
Downsampling Factor: 101

Out	2457.3
New Result	False

With the help of the function blocks **Position Mover 1 (Multiplexer)** and **Position Mover 2 (Multiplexer)** the position of the two movers of the tray whose workpiece is currently being milled is extracted.

The function block **Edge Counter (Start Milling)** outputs a rising edge when the milling process begins. This is required as input of subsequent function blocks.

The two downsampling function blocks (**Sampling Position Mover 1** & **Sampling Position Mover 2**) store a position value every 100 cycles (downsampling factor) over the milling process, starting with the rising edge of *bMillerMilling*. The function blocks are only active when milling is taking place (*bMillerMilling* == TRUE).



Collect Position Mover 1 (BatchNShift 1Ch)

Input	Out @ Sampling Position Mover 1 (Do...	2583.6
Enable Execution	New Result @ Sampling Position Mov...	False
Reset	Edge @ Start Milling (Edge Counter 1...	False

Buffer Size: 8

Sample Mode: Wait

Output	
Output 00	2583.6
Output 01	2564.7
Output 02	2597.6
Output 03	2598.3
Output 04	2565
Output 05	2583.3
Output 06	2606.1
Output 07	2585
New Result	False

Collect Position Mover 2 (BatchNShift 1Ch)

Input	Out @ Sampling Position Mover 2 (Do...	2457.3
Enable Execution	New Result @ Sampling Position Mov...	False
Reset	Edge @ Start Milling (Edge Counter 1...	False

Buffer Size: 8

Sample Mode: Wait

Output	
Output 00	2457.3
Output 01	2472.2
Output 02	2517.6
Output 03	2477.3
Output 04	2455
Output 05	2508
Output 06	2500
Output 07	2458.8
New Result	FALSE

Clustering Shape (DenStream)

Update Micro Cluster	New Result @ Collect Position Mover...	False
Input		
Input 00	Output 01 @ Collect Position Mover 1...	2564.7
Input 01	Output 02 @ Collect Position Mover 1...	2597.6
Input 02	Output 03 @ Collect Position Mover 1...	2598.3
Input 03	Output 04 @ Collect Position Mover 1...	2565
Input 04	Output 05 @ Collect Position Mover 1...	2583.3
Input 05	Output 06 @ Collect Position Mover 1...	2606.1
Input 06	Output 07 @ Collect Position Mover 1...	2585
Enable Execution	New Result @ Collect Position Mover...	False
Reset	<Empty>	-

Num Channels	
Epsilon	10
Mu x Beta	2
Lambda	1E-06
Epsilon (DBSCAN)	25
Min Weight (DBSCAN)	2
potMCs Buffer Size	100
outMCs Buffer Size	100

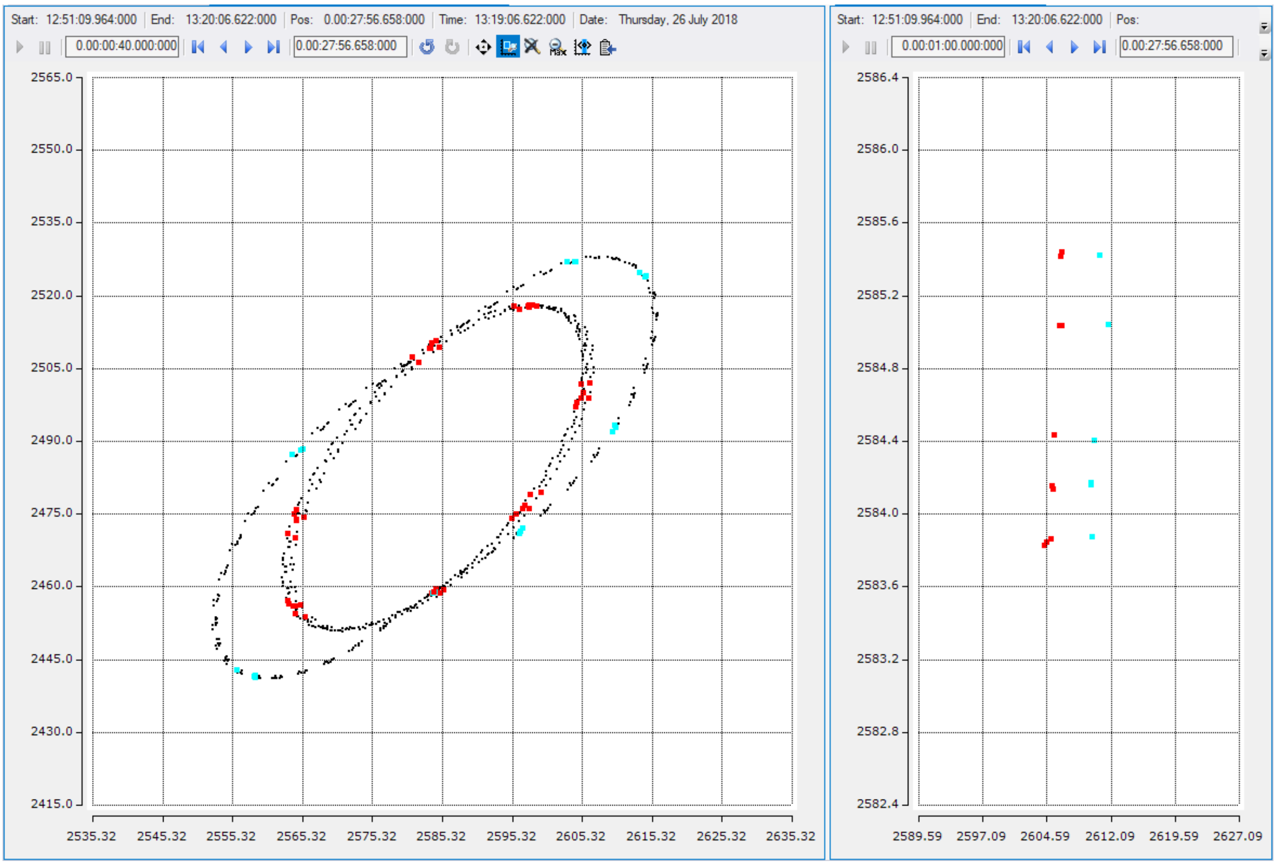
MC Buffer Overflow	False
Last Event	26 07 2018 13:20:05.630
Last Switch	No event occured
Number of potMCs	2
Number of outMCs	0
Cluster Index	0
Number of Clusters	2
New Result	FALSE

The outputs of the downsampling blocks are each linked to an input of a BatchNShift function block (**Collect Position Mover 1** & **Collect Position Mover 2**). These summarize the values in a batch. The batch is output only when the number of eight values (*Buffer Size*) is reached.

Finally, the outputs of the first BatchNShift are linked to the inputs of the DenStream function block. The first value is discarded because the position at the beginning of the milling process is independent of the milling shape and therefore does not contain any information for the classification.

The micro clusters are updated only when the DownsamplingBuffer function block updates its outputs. In order to estimate the parameter values for *Epsilon* and *Epsilon (DBSCAN)* correctly, you can look at the distances between the individual input variables for the different milling shapes in a plot. Since the milling shapes do not change over time, the parameter value *Lambda* is set very low and could also be set to 0.

The scope charts show the results from the analysis.



If the position of the two movers of a tray is presented in an XY chart, these show the movement curve that is traversed by the trays during milling. The motion curve is shown in black in the left plot. In the time period shown here, both milling shapes were driven several times. The points used as inputs for the DenStream function block are shown in color. Dynamic styles display the points in different colors depending on the associated cluster index. It can be seen that the clusters have been correctly assigned. The points representing the oval milling shape are turquoise, while those for the others are red, therefore different cluster indices have been assigned to them.

The feature space of the clusters is not displayed. Since this is seven-dimensional, it cannot be directly visualized. Two features (Input 05 and Input 06) are shown in the right-hand plot. It can be seen that the points of the two clusters are spatially separated from each other

Downloads

- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12177933195.zip

8.3 Pick-and-place application: correlation

The application described below is an example application of the CorrelationFunctionReference function block.

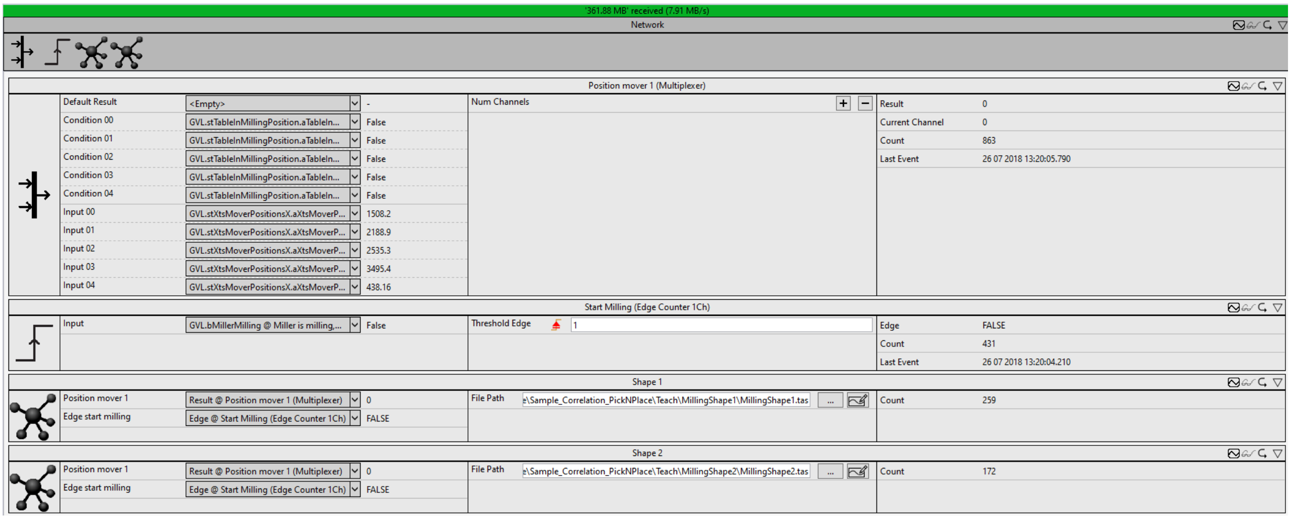
The sample is based on data from the pick-and-place application. For this reason, a good understanding of the pick-and-place sample is a good basis for understanding this application. Data access is performed in the same way.

The aim of the application is to divide the milled shapes into two groups, oval (red) and round (yellow), based on the movement performed during the milling process.

The movement of one of the two movers on which the tray is mounted was previously recorded for the two shapes and is available in the files *MillingShape1.tas* and *MillingShape2.tas* for the two different milling shapes. These sequences are each correlated with the current motion pattern. The maximum coefficient output by the CorrelationFunctionReference function block is used to determine whether the shapes match or not.

In addition, the *Maximizing Lag* output value of the CorrelationFunctionReference function block determines the delay in number of cycles after the miller is turned on before the shape is followed.

The figures below show the individual function blocks of the application.


'361.88 MB' received (7.91 MB/s)

Network

Position mover 1 (Multiplexer)

Default Result	<Empty>	-
Condition 00	GVL.stTableInMillingPosition.aTableIn...	False
Condition 01	GVL.stTableInMillingPosition.aTableIn...	False
Condition 02	GVL.stTableInMillingPosition.aTableIn...	False
Condition 03	GVL.stTableInMillingPosition.aTableIn...	False
Condition 04	GVL.stTableInMillingPosition.aTableIn...	False
Input 00	GVL.stXtsMoverPositionsX.aXtsMoverP...	1508.2
Input 01	GVL.stXtsMoverPositionsX.aXtsMoverP...	2188.9
Input 02	GVL.stXtsMoverPositionsX.aXtsMoverP...	2535.3
Input 03	GVL.stXtsMoverPositionsX.aXtsMoverP...	3495.4
Input 04	GVL.stXtsMoverPositionsX.aXtsMoverP...	438.16

Num Channels

Result 0
Current Channel 0
Count 863
Last Event 26 07 2018 13:20:05.790

Start Milling (Edge Counter 1Ch)

Input GVL.bMillerMilling @ Miller is milling,... False
Threshold Edge 1
Edge FALSE
Count 431
Last Event 26 07 2018 13:20:04.210

Shape 1

Position mover 1 Result @ Position mover 1 (Multiplexer) 0
Edge start milling Edge @ Start Milling (Edge Counter 1Ch) FALSE
File Path e\Sample_Correlation_PickNPlace\Teach\MillingShape1\MillingShape1.tas
Count 259

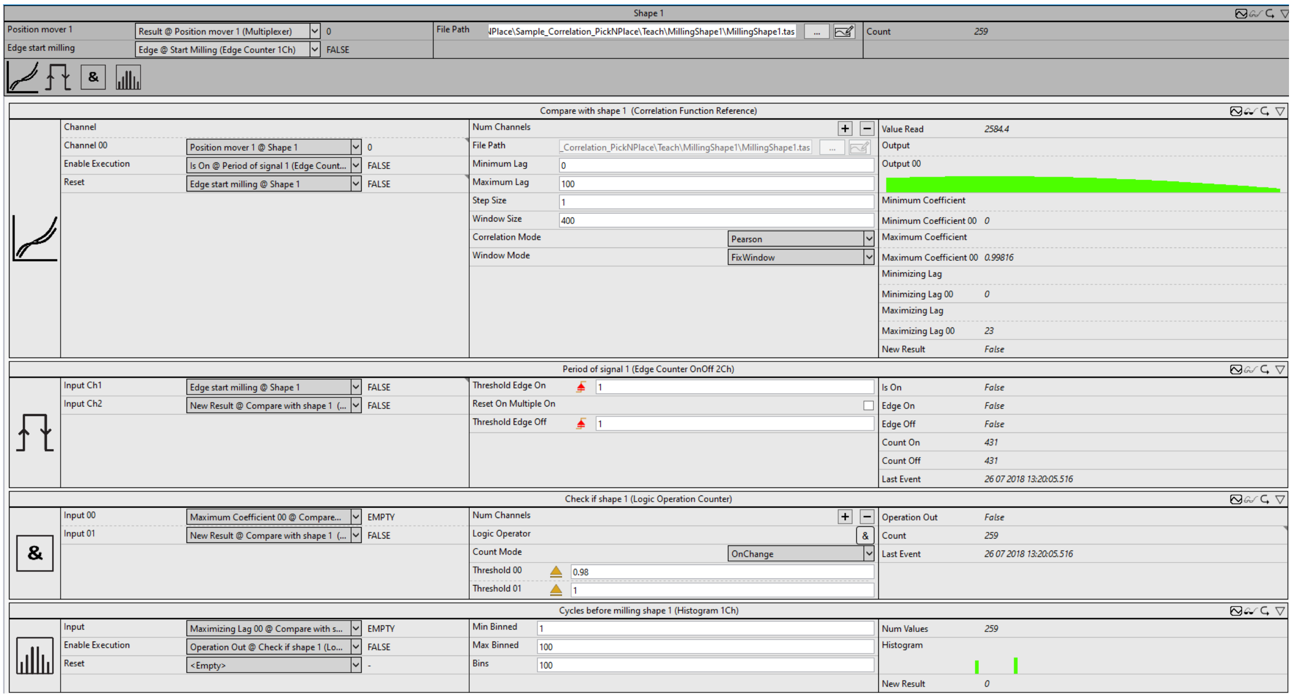
Shape 2

Position mover 1 Result @ Position mover 1 (Multiplexer) 0
Edge start milling Edge @ Start Milling (Edge Counter 1Ch) FALSE
File Path e\Sample_Correlation_PickNPlace\Teach\MillingShape2\MillingShape2.tas
Count 172

The **Position Mover 1 (Multiplexer)** function block is used to extract the position of the first mover belonging to the tray whose workpiece is currently being milled. This is also the mover for which the motion profile is available over the course of the milling process for the two shapes. For the analysis, it is not necessary to consider both movers of a tray, since the movers are mounted at a constant distance from each other and thus have the same motion profile.

The function block **Edge Counter (Start Milling)** outputs a rising edge when the milling process begins. This edge is required as input for subsequent function blocks.

The calculations of the correlations are performed individually for each milling shape in a separate network.


Shape 1

Position mover 1 Result @ Position mover 1 (Multiplexer) 0
Edge start milling Edge @ Start Milling (Edge Counter 1Ch) FALSE
File Path NPlace\Sample_Correlation_PickNPlace\Teach\MillingShape1\MillingShape1.tas
Count 259

Compare with shape 1 (Correlation Function Reference)

Channel
Channel 00 Position mover 1 @ Shape 1 0
Enable Execution Is On @ Period of signal 1 (Edge Count... FALSE
Reset Edge start milling @ Shape 1 FALSE

Num Channels
File Path _Correlation_PickNPlace\Teach\MillingShape1\MillingShape1.tas
Minimum Lag 0
Maximum Lag 100
Step Size 1
Window Size 400
Correlation Mode Pearson
Window Mode FixWindow

Value Read 2584.4
Output
Output 00
Minimum Coefficient
Minimum Coefficient 00 0
Maximum Coefficient
Maximum Coefficient 00 0.99816
Minimizing Lag
Minimizing Lag 00 0
Maximizing Lag
Maximizing Lag 00 23
New Result False

Period of signal 1 (Edge Counter OnOff 2Ch)

Input Ch1 Edge start milling @ Shape 1 FALSE
Input Ch2 New Result @ Compare with shape 1 (... FALSE

Threshold Edge On 1
Reset On Multiple On □
Threshold Edge Off 1

Is On False
Edge On False
Edge Off False
Count On 431
Count Off 431
Last Event 26 07 2018 13:20:05.516

Check if shape 1 (Logic Operation Counter)

Input 00 Maximum Coefficient 00 @ Compare... EMPTY
Input 01 New Result @ Compare with shape 1 (... FALSE

Num Channels
Logic Operator &
Count Mode OnChange
Threshold 00 0.98
Threshold 01 1

Operation Out False
Count 259
Last Event 26 07 2018 13:20:05.516

Cycles before milling shape 1 (Histogram 1Ch)

Input Maximizing Lag 00 @ Compare with s... EMPTY
Enable Execution Operation Out @ Check if shape 1 (Lo... FALSE
Reset <Empty> -

Min Binned 1
Max Binned 100
Bins 100

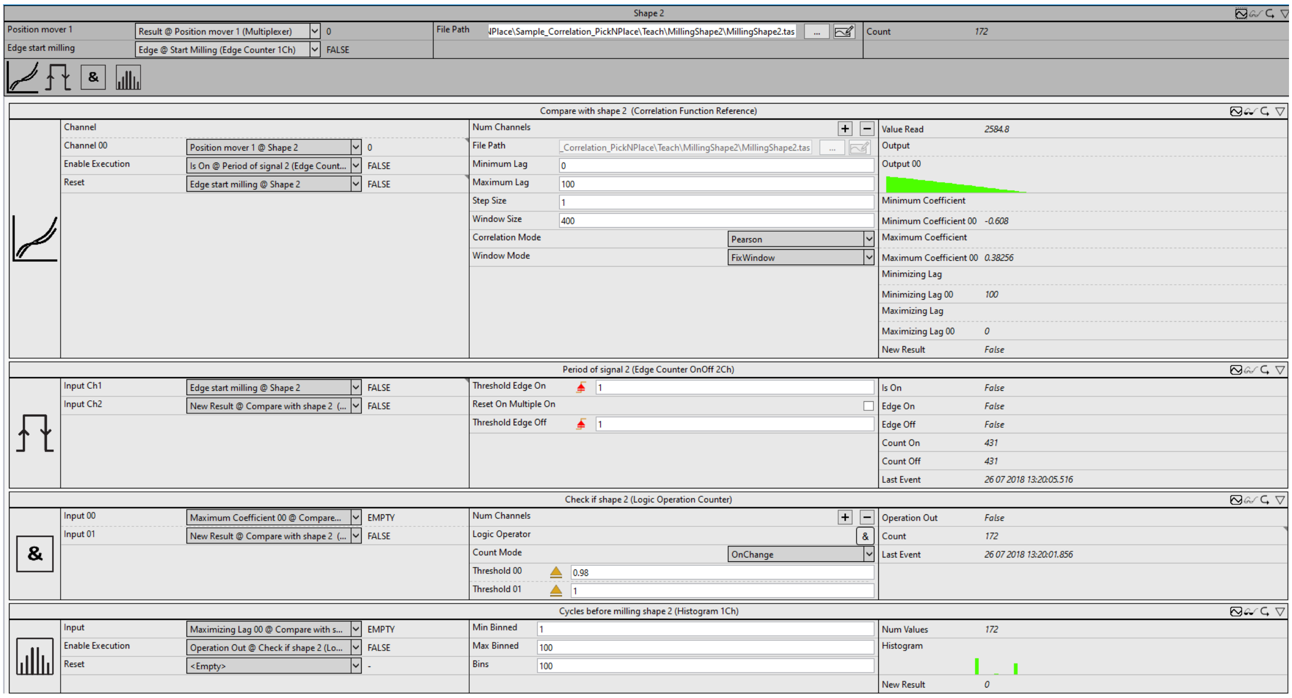
Num Values 259
Histogram
New Result 0

In the function block **Compare with shape 1 (Correlation Function Reference)**, the position of the mover (*Result* output of the multiplexer function block) is correlated with the recorded signal of the first shape. The correlation calculation starts with the switching on of the miller. This is achieved by linking the *Reset* input to the rising edge to turn on the miller (*edge output* from **Start Milling (Edge Counter 1Ch)**). This is because the correlation calculation starts again with each rising edge of this function block.

Before starting the application, the path to the reference file must be set. The reference file is located in the project directory in the folder *"Teach\MillingShape1\MillingShape1.tas".*

To get new results only when milling is in progress, the *Enable Execution* input of the correlation function block is linked. Here, the *Is On* output of the subsequent function block **Period of Signal 1 (Edge Counter OnOff 2Ch)** is linked. This becomes TRUE when the miller is turned on and FALSE again, if all values from the file have been used once for the correlation calculation and the correlation function block has updated its outputs. The **Check if shape 1 (Logic Operation Counter)** function block returns TRUE if the maximum Pearson correlation coefficient is greater than 0.98 after comparing the sequences. This way, the number of milled parts with shape 1 is also counted.

For the milling procedures, which are thus assigned to the first shape, the function block **Cycles before Milling Shape 1 (Histogram 1Ch)** determines the distribution of the number of cycles that have elapsed between switching on the miller and driving along the contour.


Shape 2
Position mover 1 | Result @ Position mover 1 (Multiplexer) | 0
Edge start milling | Edge @ Start Milling (Edge Counter 1Ch) | FALSE
File Path | NPlace\Sample_Correlation_PickNPlace\Teach\MillingShape2\MillingShape2.tas
Count | 172

Compare with shape 2 (Correlation Function Reference)
Channel
Channel 00 | Position mover 1 @ Shape 2 | 0
Enable Execution | Is On @ Period of signal 2 (Edge Count... | FALSE
Reset | Edge start milling @ Shape 2 | FALSE
Num Channels
File Path | _Correlation_PickNPlace\Teach\MillingShape2\MillingShape2.tas
Minimum Lag | 0
Maximum Lag | 100
Step Size | 1
Window Size | 400
Correlation Mode | Pearson
Window Mode | FixWindow
Value Read | 2584.8
Output
Output 00
Minimum Coefficient
Minimum Coefficient 00 | -0.608
Maximum Coefficient
Maximum Coefficient 00 | 0.38256
Minimizing Lag
Minimizing Lag 00 | 100
Maximizing Lag
Maximizing Lag 00 | 0
New Result | False

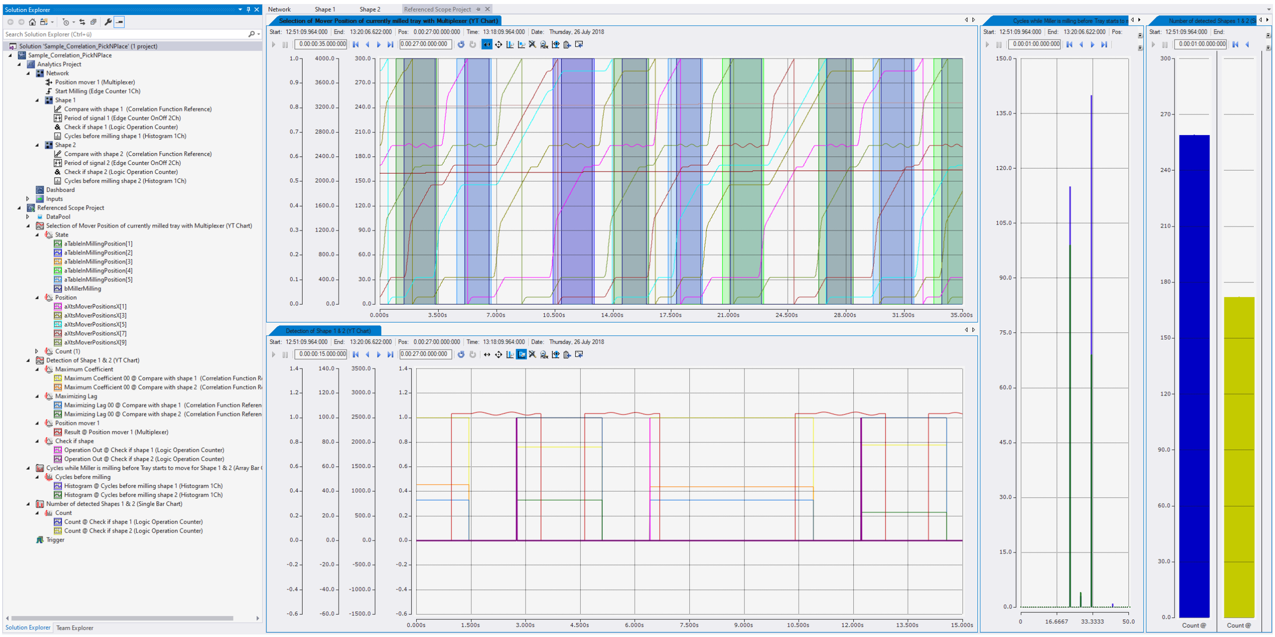
Period of signal 2 (Edge Counter OnOff 2Ch)
Input Ch1 | Edge start milling @ Shape 2 | FALSE
Input Ch2 | New Result @ Compare with shape 2 (... | FALSE
Threshold Edge On | 1
Reset On Multiple On | □
Threshold Edge Off | 1
Is On | False
Edge On | False
Edge Off | False
Count On | 431
Count Off | 431
Last Event | 26 07 2018 13:20:05.516

Check if shape 2 (Logic Operation Counter)
Input 00 | Maximum Coefficient 00 @ Compare... | EMPTY
Input 01 | New Result @ Compare with shape 2 (... | FALSE
Num Channels
Logic Operator | &
Count Mode | OnChange
Threshold 00 | 0.98
Threshold 01 | 1
Operation Out | False
Count | 172
Last Event | 26 07 2018 13:20:01.856

Cycles before milling shape 2 (Histogram 1Ch)
Input | Maximizing Lag 00 @ Compare with s... | EMPTY
Enable Execution | Operation Out @ Check if shape 2 (Lo... | FALSE
Reset | <Empty> | -
Min Binned | 1
Max Binned | 100
Bins | 100
Num Values | 172
Histogram
New Result | 0

The correlation calculation of the second milling shape is performed analogously in a separate network.

The scope plots below visualize the results from the analysis.


Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'Sample_Correlation_PickNPlace' (1 project)
Sample_Correlation_PickNPlace
Analytics Project
Network
Position mover 1 (Multiplexer)
Start Milling (Edge Counter 1Ch)
Shape 1
Compare with shape 1 (Correlation Function Reference)
Period of signal 1 (Edge Counter OnOff 2Ch)
Check if shape 1 (Logic Operation Counter)
Cycles before milling shape 1 (Histogram 1Ch)
Shape 2
Compare with shape 2 (Correlation Function Reference)
Period of signal 2 (Edge Counter OnOff 2Ch)
Check if shape 2 (Logic Operation Counter)
Cycles before milling shape 2 (Histogram 1Ch)
Dashboard
Inputs
Referenced Scope Project
DataPool
Selection of Mover Position of currently milled tray with Multiplexer (YT Chart)
State
aTableInMillingPosition[1]
aTableInMillingPosition[2]
aTableInMillingPosition[3]
aTableInMillingPosition[4]
aTableInMillingPosition[5]
bMillerMilling
Position
aXtsMoverPositionsX[1]
aXtsMoverPositionsX[3]
aXtsMoverPositionsX[5]
aXtsMoverPositionsX[7]
aXtsMoverPositionsX[9]
Count (1)
Detection of Shape 1 & 2 (YT Chart)
Maximum Coefficient
Maximum Coefficient 00 @ Compare with shape 1 (Correlation Function R
Maximum Coefficient 00 @ Compare with shape 2 (Correlation Function R
Maximizing Lag
Maximizing Lag 00 @ Compare with shape 1 (Correlation Function Referen
Maximizing Lag 00 @ Compare with shape 2 (Correlation Function Referen
Position mover 1
Result @ Position mover 1 (Multiplexer)
Check if shape
Operation Out @ Check if shape 1 (Logic Operation Counter)
Operation Out @ Check if shape 2 (Logic Operation Counter)
Cycles while Miller is milling before Tray starts to move for Shape 1 & 2 (Array Bar (
Cycles before milling
Histogram @ Cycles before milling shape 1 (Histogram 1Ch)
Histogram @ Cycles before milling shape 2 (Histogram 1Ch)
Number of detected Shapes 1 & 2 (Single Bar Chart)
Count
Count @ Check if shape 1 (Logic Operation Counter)
Count @ Check if shape 2 (Logic Operation Counter)
Trigger
Solution Explorer
Team Explorer
Network
Shape 1
Shape 2
Referenced Scope Project
Selection of Mover Position of currently milled tray with Multiplexer (YT Chart)
Start: 12:51:09.964 000
End: 13:20:06.622 000
Pos: 0.00:27:00.000 000
Time: 13:18:09.964 000
Date: Thursday, 26 July 2018
0.000s
3.500s
7.000s
10.500s
14.000s
17.500s
21.000s
24.500s
28.000s
31.500s
35.000s
Detection of Shape 1 & 2 (YT Chart)
0.000s
1.500s
3.000s
4.500s
6.000s
7.500s
9.000s
10.500s
12.000s
13.500s
15.000s
Cycles while Miller is milling before Tray starts to
0
16.6667
33.3333
50.0
Number of detected Shapes 1 & 2
Count @
Count @

The upper left chart shows the inputs and the *Result* output of the multiplexer. It can be seen that the mover follows a sinusoidal path when the corresponding tray is in the position of the miller.

In the lower left chart, the *Result* output of the multiplexer is shown in red. The two function blocks **Check if shape 1 (Logic Operation Counter)** and **Check if shape 2 (Logic Operation Counter)** each have a peak (pink and purple) if the correlated sequence has achieved a maximum coefficient (yellow and orange) greater than 0.98. In the section shown, you can check that these results match the different curves of the *Result* output of the multiplexer. Also shown in blue and green are the corresponding *Maximizing Lag* outputs of the correlation function blocks.

The left of the two bar charts shows the *Maximizing Lag* outputs of the correlation function blocks. They result from the results of the histogram function blocks. Both shapes produce large deflections at a lag of 23 as well as a lag of 33 and a small deflection at a lag of lag 28. The first shape generates a unique lag of 43 cycles.

The right-hand bar chart shows the number of parts that have been assigned to shape 1 and 2. 259 parts were assigned to the first shape and 172 parts to the second.

Downloads

- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12177935499.zip

9 Appendix

9.1 FAQ - frequently asked questions and answers

In this section frequently asked questions are answered, in order to facilitate your work with the TwinCAT Analytics Workbench. If you have any further questions, please contact our support team at support@beckhoff.com.

1. Are open source software components used in TwinCAT Measurement products? [▶ 551]

Are open source software components used in TwinCAT Measurement products?

Yes, various open source components are used. You can find a list of them including license conditions in the directory *...\TwinCAT\\Functions\TwinCAT Measurement\Legal*.

Trademark statements

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.

Third-party trademark statements

Chrome, Chromium and Google are trademarks of Google LLC.

Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

More Information:
www.beckhoff.com/TE3500

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com