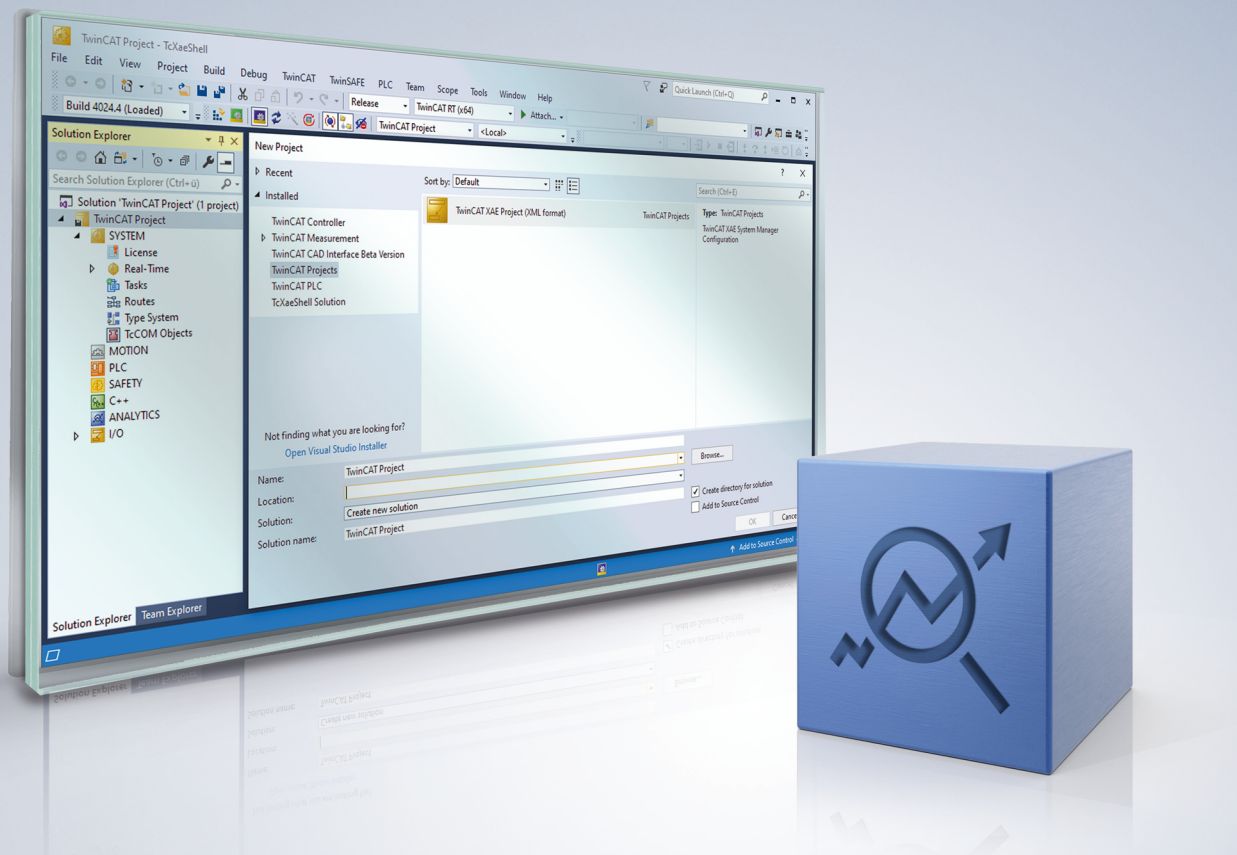
BECKHOFF New Automation Technology

取扱説明書 | JA

TE3500 | Analytics Workbench

TwinCAT 3



2025-10-09 | バージョン: 1.11.1

目次

1 序文 5
1.1 取扱説明書に関する注記 5
1.2 安全にご使用いただくために 6
1.3 情報セキュリティに関する注記 6
2 概要 8
3 インストール方法 11
3.1 システム要件 11
3.2 インストールとライセンス認証 11
3.3 ライセンス認証 15
4 解析ワークフロー - 最初のステップ 18
4.1 機械データの記録 18
4.2 通信 21
4.3 データの履歴管理 22
4.4 データの解析 29
4.5 24時間解析アプリケーション 34
5 技術概要 44
5.1 基本的なコンセプト 44
6 設定 46
6.1 データソース 46
6.1.1 ウィザード 48
6.2 Storage Provider Recorder 54
6.3 Data Scout 59
6.3.1 Data Track Editor 60
6.4 エディタの基本 72
6.4.1 ネットワーク 72
6.4.2 有効化と無効化 76
6.4.3 固定値 78
6.4.4 グローバルパラメータ 79
6.5 アルゴリズム 80
6.5.1 Analytics - Base（解析 - 基本） 82
6.5.2 Analytics - Classification（解析 - 分類） 136
6.5.3 Analytics - Clustering（解析 - クラスタリング） 162
6.5.4 Analytics - Compare（解析 - 比較） 170
6.5.5 Analytics - Energy（解析 - 電力） 194
6.5.6 Analytics - Math（解析 - 算術） 200
6.5.7 Analytics - Preprocessing（解析 - 前処理） 214
6.5.8 Analytics - Reporting（解析 - レポート） 216
6.5.9 Analytics - Specific（解析 - 特殊） 220
6.5.10 Analytics - Statistics（解析 - 統計） 229
6.5.11 Analytics - Training Base（解析 - 教示ベース） 253
6.5.12 Analytics - Visualization Only（解析 - 可視化のみ） 255

6.5.13 Analytics - XY Path Analysis（解析 - XYパス解析） 262
6.5.14 Analytics - extension of the algorithms（解析 - アルゴリズムの拡張） 272
6.6 スコープとの連携 348
6.6.1 ネットワークテンプレートでのスコープ設定 352
6.7 履歴データの取り扱い 355
6.8 ランタイムへのデプロイ 357
6.8.1 アルゴリズムの特性 364
6.8.2 PLCコード 366
6.9 HMIワンクリックダッシュボード 425
6.9.1 Analyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理 428
6.9.2 カスタマイズされた独自コントロールの使用 438
6.9.3 ユーザ管理とアクセス権の設定 457
6.9.4 ダッシュボードの設定 460
6.9.5 HMIエンジニアリングで生成したダッシュボードを修正する 467
6.9.6 ランタイムでウェブブラウザ上でダッシュボードを設定 467
6.9.7 ウェブブラウザでチャートを作成 474
6.9.8 HMIダッシュボードで複数の機械を切り替える 499
6.9.9 言語切り替え機能の統合 505
6.9.10 標準 TwinCAT HMIプロジェクトでのインタラクティブ機能の使用 508
6.10 レポートの統合 510
6.10.1 基本的なコンセプト 510
6.10.2 解析レポート 512
6.10.3 TwinCAT Reporting Server 520
6.11 解析結果の比較 531
6.11.1 プロジェクト設定の読み込みと保存 533

7 API 535
7.1 Data Exchange API 535
7.1.1 .NET 535
7.1.2 Python 541

8 サンプル 544
8.1 ピック&プレース アプリケーション 544
8.2 ピック&プレースアプリケーション：DenStream 551
8.3 ピック&プレースアプリケーション：相関（Correlation） 553

9 付録 558
9.1 FAQ - よくある質問 558

1 序文

1.1 取扱説明書に関する注記

この説明書は対応する国内規格を熟知した、トレーニングを受けた制御、オートメーションエンジニアリングの有資格者のみの使用を対象としています。
本製品の設置およびコミッショニングの際は、必ず以下の注意事項と説明に従ってください。
有資格者は、常に最新版のドキュメントを参照してください。

本製品を使用する上での責任者は、本製品の用途および使用方法が、関連するすべての法律、法規、ガイドラインおよび規格を含む、安全に関するすべての要件を満たしていることを確認してください。

免責事項

この取扱説明書の記載内容は、一般的な製品説明および性能を記載したものであり、場合により記載通りに動作しないことがあります。製品の情報・仕様は予告なく変更されます。
この説明書に記載されているデータ、図および説明に基づいて、既に納品されている製品の変更を要求することはできません。
掲載されている写真やイラストと、実際の製品は異なる場合があります。この説明書は最新でない可能性があります。必ず最新バージョンの説明書を参照してください。

商標

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS®, XPlanar® は、Beckhoff Automation GmbH の登録商標です。
この取扱説明書で使用されているその他の名称は商標である可能性があり、第三者が独自の目的のために使用すると所有者の権利を侵害する可能性があります。

EtherCAT®は、Beckhoff Automation GmbHの登録商標および特許技術です。

著作権

© Beckhoff Automation GmbH & Co.KG, Germany.
明示的な許可なく、本書の複製、配布、使用、および他への内容の転載は禁止されています。
これに違反した者は損害賠償の責任を負います。ベッコフは、特許、実用新案、意匠の付与に関するすべての権利を留保しています。

第三者商標

本書では、他社の商標が使用される場合があります。商標に関する詳細は以下を参照してください。
https://www.beckhoff.com/trademarks

1.2 安全にご使用いただくために

安全に関する注意事項

本書の各項目に記載されている製品の安全に関する指示をよくお読みになり、それに従ってください。

免責事項

すべての製品は、用途に適した特定のハードウェア構成およびソフトウェア構成を有する状態で供給されます。ハードウェアまたはソフトウェアに取扱説明書に記載されている以外の変更を加えることは許可されていません。許可されていない変更を加えると、Beckhoff Automation GmbH & Co. KGの保証の対象外となります。

使用者の資格

この説明は対応する国内規格を熟知した、トレーニングを受けた制御、自動化、およびドライブ技術の有資格者のみを対象としています。

安全記号の説明

この取扱説明書では、以下の安全記号を使用します。人的傷害および物的損害を避けるため、安全に関する注意事項はよくお読みになり、必ず指示に従ってください。人的傷害および物的損害を避けるため、安全に関する注意事項はよくお読みになり、必ず指示に従ってください。

人的傷害に関する警告:

⚠ 危険
この記号がついた注意事項に従わない場合、死亡または重傷を負います。

⚠ 警告
この記号がついた注意事項に従わない場合、死亡や重傷を負う可能性があります。

⚠ 注意
この記号がついた注意事項に従わない場合、軽傷を負う可能性があります。

物的損害に関する警告

注記
環境汚染、物的損害またはデータ消失の可能性があります。

製品の取り扱いに関する情報

ここには
製品に関する推奨措置、援助、または追加情報などが記載されます。

1.3 情報セキュリティに関する注記

Beckhoff Automation GmbH & Co. KG（ベッコフ）の製品は、オンラインアクセスが可能であれば、プラント、システム、機械、ネットワークの安全な運用をサポートするセキュリティ機能を備えています。セキュリティ機能にもかかわらず、プラント、システム、機械、ネットワークをサイバー脅威から守るためには、運用のための全体的なセキュリティ コンセプトの作成、実施、継続的な更新が必要です。ベッコフが販売する製品は、全体的なセキュリティ コンセプトの一部に過ぎません。お客様は、プラント、システム、機械、ネットワークへの第三者による不正アクセスを防止する責任を負います。ネットワークは、適切な保護措置が講じられている場合にのみ、社内ネットワークまたはインターネットに接続すべきです。

また、ベッコフが推奨する適切な保護対策も遵守してください。情報セキュリティと産業セキュリティに関する詳細は、https://www.beckhoff.com/secguide を参照してください。

ベッコフの製品とソリューションは常に進化し続けています。これはセキュリティ機能にも当てはまります。継続的な開発により、ベッコフでは、製品を常に最新の状態に保ち、アップデートが提供され次第、製品にインストールすることを明示的に推奨しています。古いバージョンやサポートが終了した製品の使用は、サイバー脅威のリスクを高めるおそれがあります。

ベッコフ製品の情報セキュリティ情報については、RSSフィードをご購読ください https://www.beckhoff.com/secinfo。

2 概要

TwinCAT 3 Analytics Workbenchは、様々な分散型マシンコントローラから継続的に解析データを作成するためのTwinCAT 3エンジニアリング製品です。ワークベンチの設定はMicrosoft VisualStudio®に統合され、グラフィカルなユーザインターフェースとして設計されています。サイクルタイムモニタリング、ライフカウント、ライフタイム、最小値/最大値/平均値など、多くのアルゴリズムが、解析の設定に使用できるツールボックスに用意されています。

TwinCAT 3 Analytics Workbench には、信号曲線を簡単に可視化するための TE1300 TwinCAT 3 Scope View Professional が含まれています。ユーザは、Analyticsコンフィグレータ から解析結果をチャートツールにドラッグ&ドロップすることで、データストリーム内の重要なポイントを強調表示できます。 このようなマーキングには、単純な極小値や極大値、カウンタ値、または機械制御装置からの結果を論理的に組み合わせ、データストリーム内で識別できるようにする論理演算子の結果を使用できます。これにより、スコープビュー内の他の信号と正確な周期で相関をとることができます。

MQTTの入力データは、TwinCATTarget Browserで選択できます。ライブデータに加え、TF3520 TwinCAT 3 Analytics Storage Provider経由で履歴データも利用可能です。解析が完了し、グラフィカルエディタでテストされると、わずか数回のクリックで設定内容を読み取り可能なPLCコードと関連するHTML5ダッシュボードに変換できます。自動生成されたPLCコードとダッシュボードは、TF3550 TwinCAT 3 Analytics Runtimeを搭載したデバイスに直接ダウンロードできます。これを常時実行し、機械の実データと並行して解析結果を提供します。TwinCAT 3 HMI経由のディスプレイには、機械オペレータ、生産管理者、機械製造者のために解析結果が表示されます。

機能

Analytics Service Toolと比較して、Analytics Workbenchはコードやダッシュボードの生成を含む補完的な機能を提供します。Service Tool の全ての機能はWorkbenchにも含まれています。

コンポーネント

- Analyticsコンフィグレータ
- 基本となる解析アルゴリズム
- Analytics PLCライブラリ
- Analyticsストレージプロバイダ PLCライブラリ
- TwinCATスコープ（TE1300およびTF3300）
- IoT通信機能
- HMIエンジニアリング（TE2000）

機能	TE3500 Analytics Workbench		TE3520 Analytics Service Tool	
	7日間トライアルライセンス	フルライセンス	7日間トライアルライセンス	フルライセンス

機能	TE3500 Analytics Workbench		TE3520 Analytics Service Tool	
全般:				
解析コンフィグレータ	✓	✓	✓	✓
解析チャンネル	最大 5	無制限	最大 5	無制限
解析モジュール/アルゴリズム	最大 3	無制限	最大 3	無制限
長時間記録 >1時間	✗	✓	✗	✓
スコープビューとの連携	✓	✓	✓	✓
ストレージプロバイダレコーダ	✓	✓	✓	✓
MQTT	✓ (最大 5)	✓	✓ (最大 5)	✓
Analyticsファイル	✓ (最大 5)	✓	✓ (最大 5)	✓
ADS	✓ (自動デプロイなし)	✓ (自動デプロイなし)	✓	✓
データエクスポートツール	✓	✓	✓	✓
基本のデータエクスポート形式	✓ (最大 5)	✓	✓ (最大 5)	✓
拡張データエクスポート形式	✗	✓	✗	✓
AnalyticsData Scout:				
データのロード	✓	✓	✓	✓
ツールバーの切り取りボタン	✗ *	✓	✗ *	✓
データをAnalyticsファイルにエクスポート	✓	✓	✓	✓
ランタイムをデプロイ：				
デプロイウィザード	✓	✓	/	/
既存のSlnにPLCコードを追加	✗	✓	/	/
PLCコードのマージ	✗	✓	/	/
ブートプロジェクトの自動作成	✗	✓	/	/
ランタイムの自動アクティベート	✗	✓	/	/
ストリーム結果	✗	✓	/	/
HMIダッシュボード	✓ (デフォルト)	✓	/	/
PLCコードのリセットアルゴリズム	✗	✓	/	/
レポートの統合：				

機能	TE3500 Analytics Workbench		TE3520 Analytics Service Tool	
報告アルゴリズムによる常時レポート	✔	✔	/	/
オンデマンドレポート				
情報の追加	✔ (最大 1)	✔	✔ (最大 1)	✔
特定の解析情報の選択	✘	✔	✘	✔
✔	フルサポート			
✘	サポートなし			
/	製品機能ではない			
*	既存のTE1300 Scope Professionalでフルサポート			

3 インストール方法

3.1 システム要件

TwinCAT Analyticsを正しく機能させるには、以下のシステム要件を満たす必要があります。

対応OS

Windows 10

TwinCAT

TwinCAT Analytics Service Tool および Workbenchを使用した開発には、TwinCAT 3.1 ビルド 4022.29以上が必要です。

.NET Framework

エンジニアリングには.NET Framework 4.7.2が必要です。

Visual Studioの開発環境

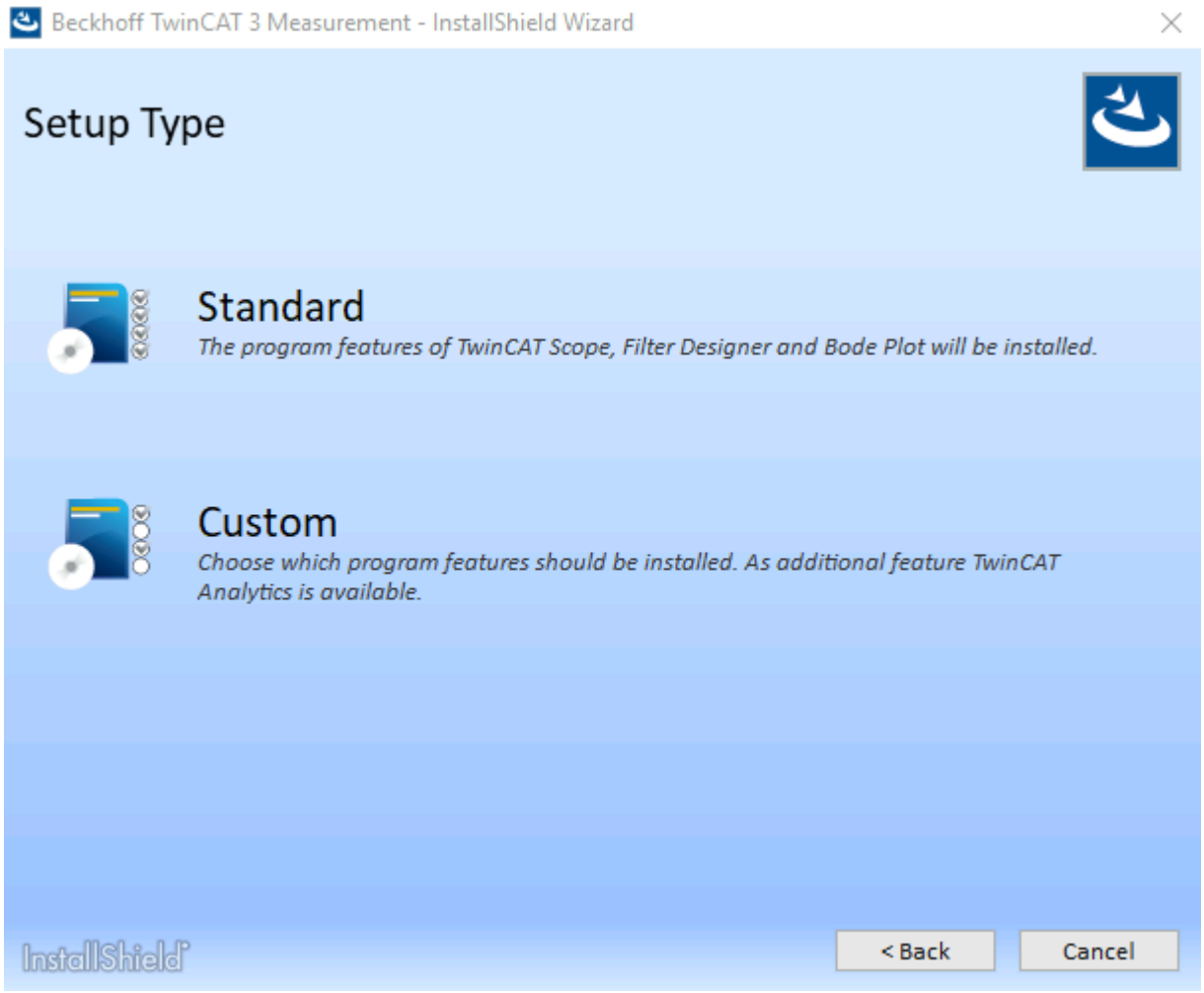
- Visual Studio® 2015
- Visual Studio® 2017
- Visual Studio® 2019
- TwinCAT XAE Shell

一般的には、Visual Studio®Shell版の使用で十分です。「Full」のセットアップを選択すると、TwinCAT XAE Shell版が自動的にインストールされます。「Update」のセットアップでは、Analyticsソースの更新のみが行われ、Visual Studio®Shellの更新は行われません。

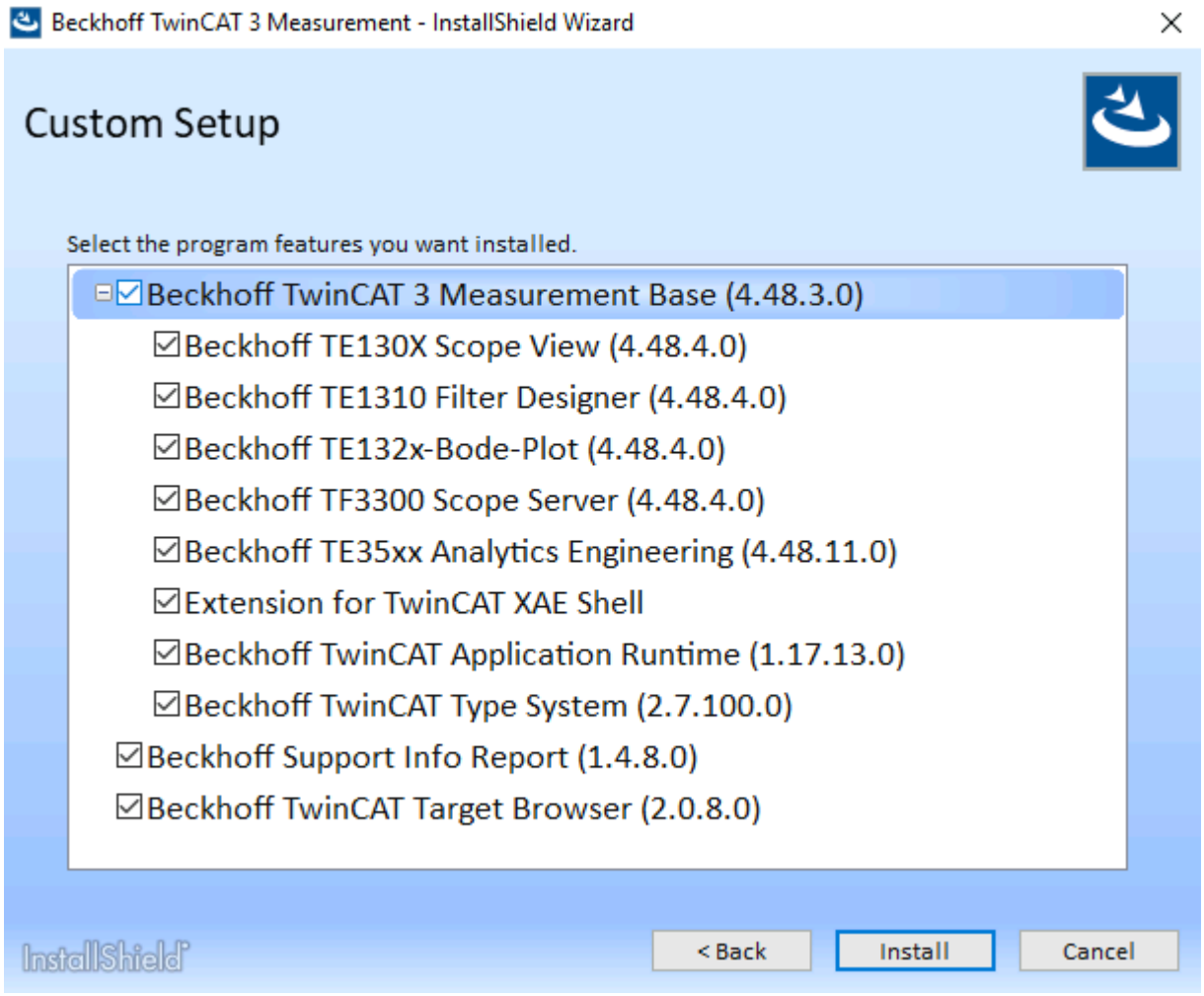
3.2 インストールとライセンス認証

TwinCAT Analyticsセットアップは、TwinCAT Measurement Suiteセットアップの一部です。セットアップは2種類から選択できます：
TC3-Measurement-Fullセットアップには、TwinCAT MeasurementコンポーネントとVisual Studio®環境が含まれます。
一方、TC3 Measurement Updateセットアップには、計測コンポーネントのみが含まれます。

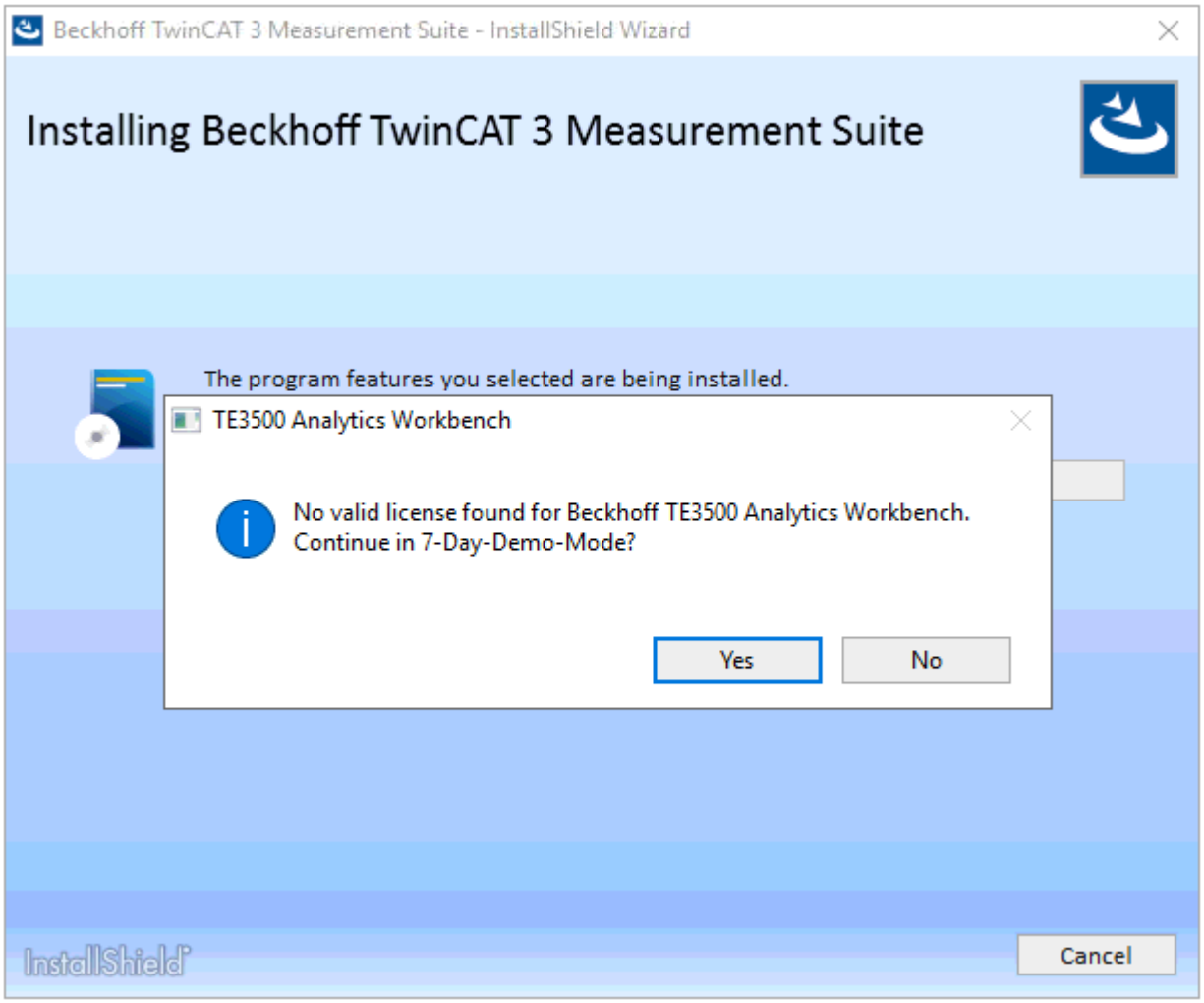
ライセンス契約の条件を確認した後、StandardかCustomのいずれかを選択できます。デフォルトでは、TwinCAT Analytics Engineeringを含む全ての計測コンポーネントが自動的にインストールされます。



Customでは、各コンポーネントの選択を解除して、システムにインストールされないようにできます。



Analyticsセットアップは、処理中にシステムに必要なAnalytics Engineeringライセンスがあるかどうかを確認します。ライセンスがない場合は、トライアルライセンスを有効にできます。トライアルライセンスは何度でも延長可能です。ただし、トライアルモードでのインストール後、このシステムでは新しいセットアップを使用して機能アップデートを実行できません。これらのアップデートをインポートできるようにするには、まずライセンスを購入する必要があります。トライアル版の機能制限の概要は、こちら [▶ 8]を参照ください。



AnalyticsエンジニアリングツールのTE3500 Analytics WorkbenchおよびTE3520 Analytics Service Toolのライセンスは、常に保守ライセンスと紐づいています。初回購入後、保守ライセンスは12ヶ月間有効です。この期間中、ソフトウェアの機能アップデートを実行できます。12ヶ月経過後も、ソフトウェアは制限なく使用できます。ただし、保守ライセンスを延長しない場合、新機能をインポートできません。インストール時に、関連するメッセージが表示されます。TE3501およびTE3521を購入することで、いつでも保守ライセンスの延長ができます。

新バージョンの拡張機能については、ベッコフの担当営業にお問い合わせください。

セットアップにはライセンスが必要

TwinCAT Analyticsエンジニアリングツールのアップデートは、有効な保守ライセンスがなければ実行できません。

他社デバイス用 TwinCAT 3 ライセンス

他社メーカのIPC（TwinCAT 3プラットフォームレベル >= 90）を使用する場合、TwinCAT 3ライセンスドングルの使用を強く推奨します（TwinCAT Analyticsのライセンス認証の必須条件ではありません）。

TwinCAT Analytics Workflow

完全版Analytics Workflowを使用するには、TE2000 HMIエンジニアリングセットアップが必要です。現時点でAnalytics Workbenchのセットアップには統合されていないためです。そのため、個別にインストールする必要があります。次のセットアップバージョンで統合される予定です。以降はこの手順は不要になります。

3.3 ライセンス認証

TwinCAT 3 ファンクションは、フルバージョンまたは7日間のテストバージョンとしてアクティベートできます。どちらのライセンスタイプも、TwinCAT 3開発環境（XAE）からアクティベートできます。

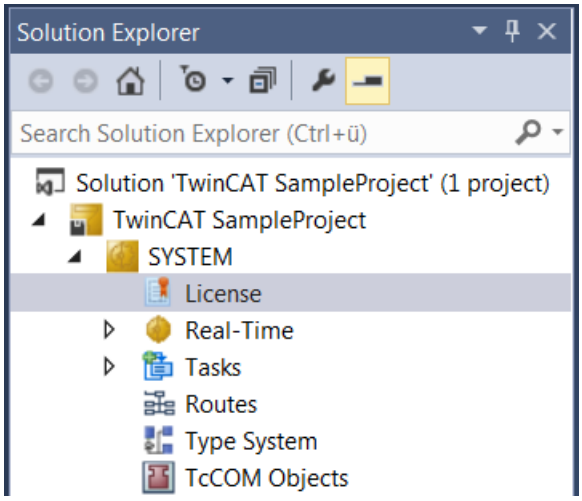
TwinCAT 3ファンクション（フルバージョン）のライセンス認証

フルバージョンのライセンス取得手順については、Beckhoff Information Systemのドキュメント TwinCAT 3 ライセンス認証を参照してください。

TwinCAT 3ファンクション（7日間テストバージョン）のライセンス認証

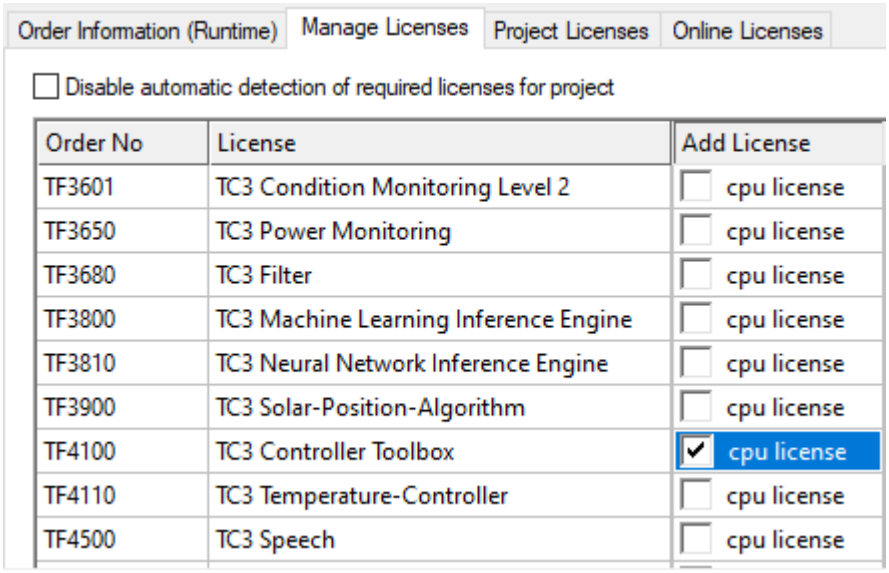
TwinCAT 3 ライセンスドングルでは、7日間テストバージョンを有効化できません。

1. TwinCAT開発環境（XAE）を起動します。
2. 既存のTwinCATプロジェクトを開くか、新規プロジェクトを作成します。
3. リモートデバイスのライセンスをアクティベートする場合は、目的のターゲットシステムを設定します。これを行うには、ツールバーの**Choose Target System**ドロップダウンリストからターゲットシステムを選択します。
   ⇨ ライセンス設定は、常に選択したターゲットシステムを参照します。ターゲットシステムでプロジェクトが有効になると、設定されたTwinCAT 3 ライセンスがこのシステムに自動的にコピーされます。
4. **Solution Explorer**で、**SYSTEM**サブツリーの**License**をダブルクリックします。

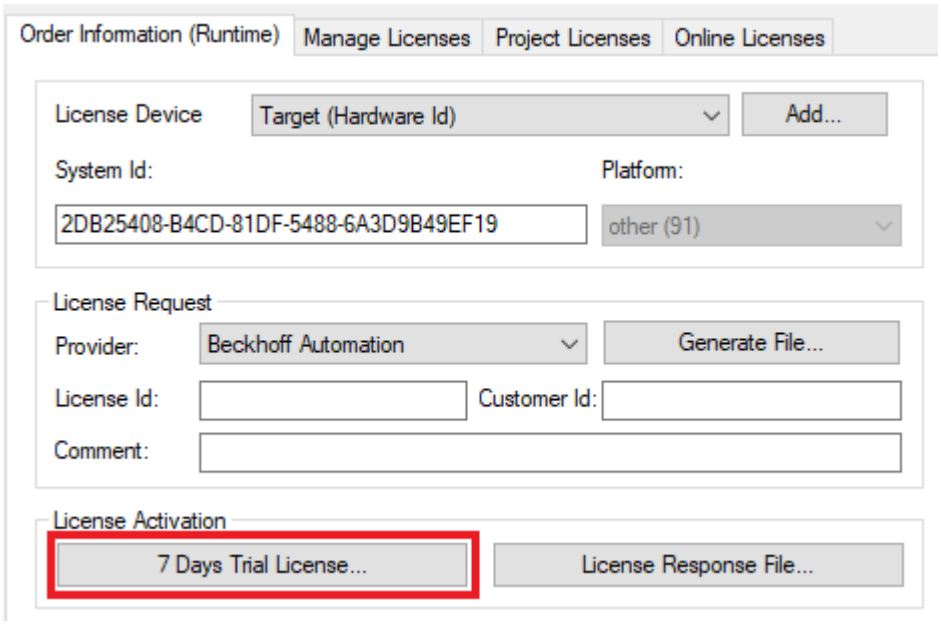


⇨ TwinCAT 3ライセンスマネージャが開きます。

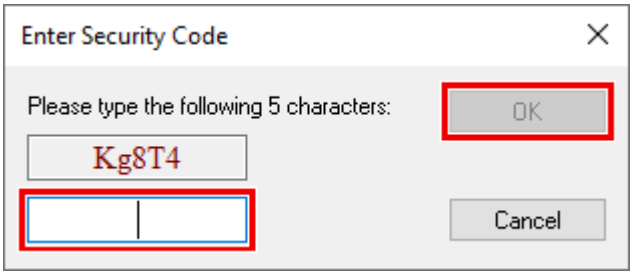
5. **Manage Licenses**タブを開きます。**Add License**の列で、プロジェクトに追加するライセンスのチェックボックスをオンにします（例：TF4100 TC3 Controller Toolbox）。



6. **Order Information (Runtime)**タブを開きます。
   ⇨ ライセンス一覧では、先に選択したライセンスが「missing（見つかりません）」というステータスで表示されます。
7. **7-Day Trial License...**をクリックして、7日間のトライアルライセンスをアクティベートします。



   ⇨ ダイアログが開きます。このダイアログに表示されたセキュリティコードを入力することが求められます。



8. 表示されたコードを正確に入力し、入力を確定します。
9. アクティベーションが成功したことを示すダイアログが表示されます。

⇨ ライセンス一覧で、ステータスにライセンスの有効期限が表示されるようになりました。

10. TwinCATシステムを再起動します。

⇨ 7日間の試用版が有効になりました。

4 解析ワークフロー - 最初のステップ

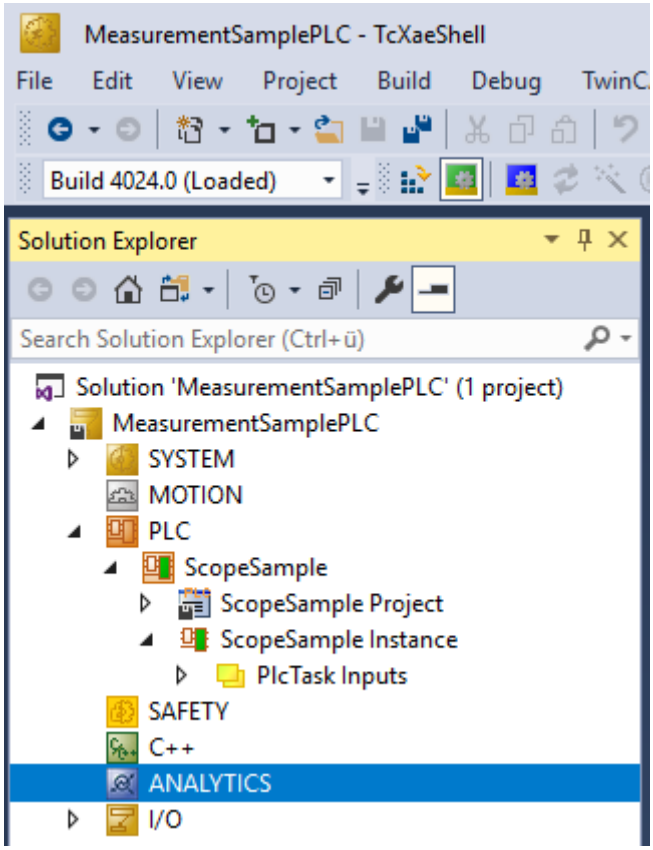
このドキュメントでは、TwinCAT Analytics のワークフロー全体を手順を踏んで紹介します。データの取得から通信、履歴管理、データの評価と解析、ウェブベースのダッシュボードでのデータの表示までを紹介します。

4.1 機械データの記録

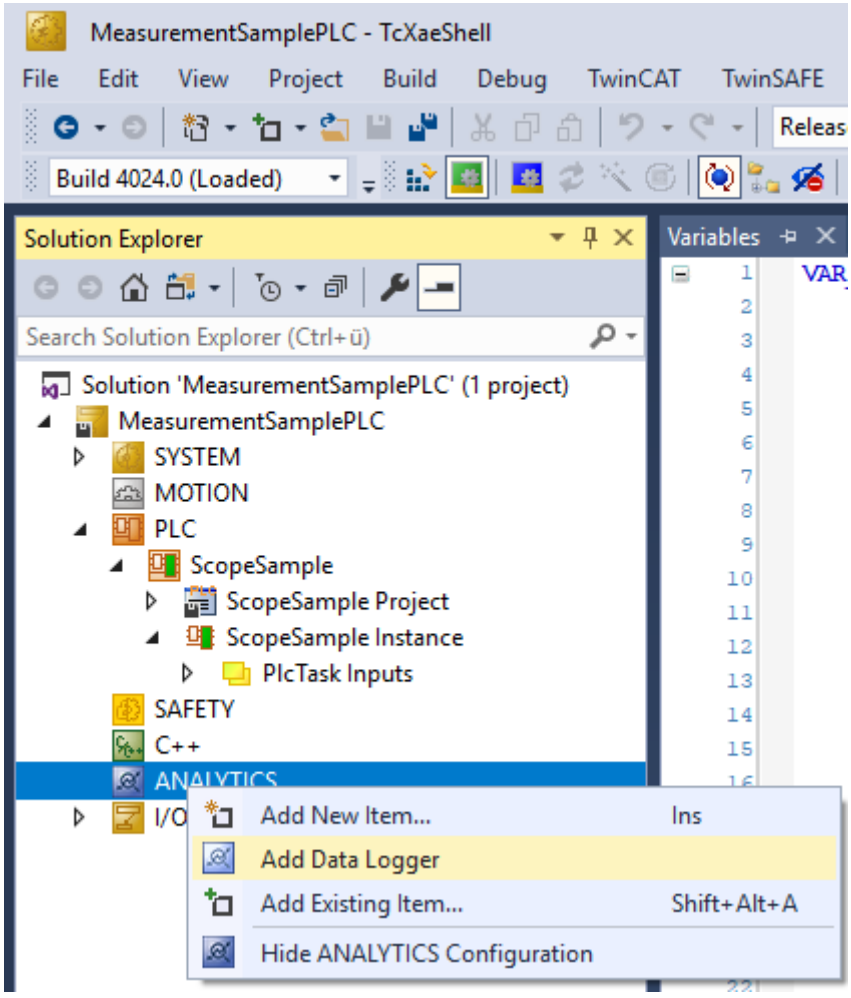
機械側には、機械の画像、PLC、NCなどからプロセスデータを記録する Analytics Loggerがあります。Loggerは、TwinCATのリアルタイムコンテキストで動作します。

TwinCAT Analytics Loggerは、TwinCAT XAEおよびXARと共にインストールされます。LoggerはMQTTクライアントとして動作し、記録されたデータをネイティブのMQTTメッセージブローカーに通信したり、同じデータ形式でローカルのバイナリファイルに保存したりできます。MQTTクライアントとして使用することで、リングバッファ機能を活用して、メッセージブローカーへの短い切断を回避できます。ローカルのバイナリファイル保存用にリングバッファを設定することもできます。

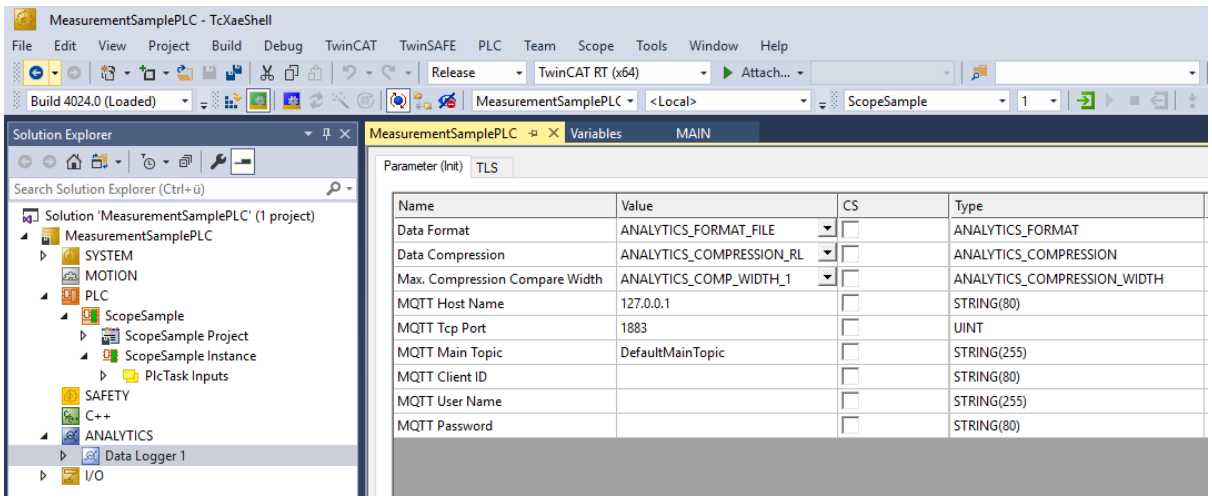
- Analytics Loggerを設定するには、既存のTwinCATプロジェクトのAnalyticsツリーノードに移動します。



- ノードを右クリックし、「Add Data Logger」をクリックして、新しいインスタンスを1つ追加します。



- 基本設定を行うには、新しいツリー項目をダブルクリックしてください。



独自の Analytics Loggerの設定が可能です。

-データ形式：バイナリファイルまたはMQTTストリーム

-ファイル形式：Analytics Loggerはデータをローカルのバイナリファイルに保存するので、その他の設定は不要 ファイルは、C:¥TwinCAT¥3.1¥Boot¥Analyticsに保存される

-バイナリ: 設定されたMQTT Message Brokerにデータが送信される 1つのTwinCATプロジェクトに複数のLoggerを設定して、異なるMQTTメッセージブローカーにデータを送信できる

-データ圧縮：オン（デフォルト）またはオフ

-最大圧縮：圧縮のモード

-MQTT ホスト名

-MQTT Tcpポート

-MQTTのメイントピックを階層化し、識別を容易にする

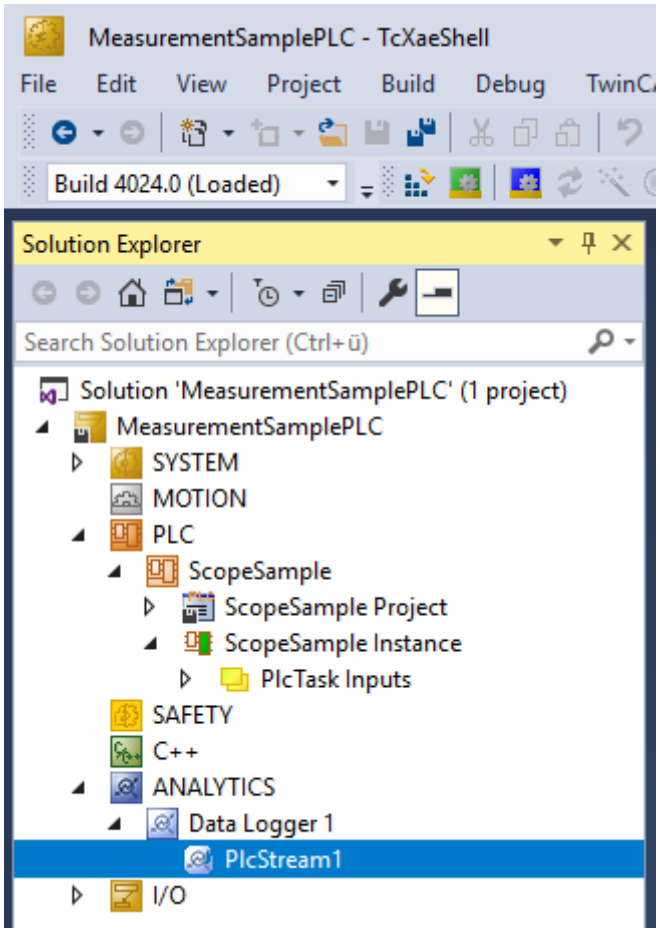
-MQTTクライアントIDはネットワーク内で一意にする必要がある

-MQTT ユーザ名

-メッセージブローカーで認証を行うためのMQTTパスワード

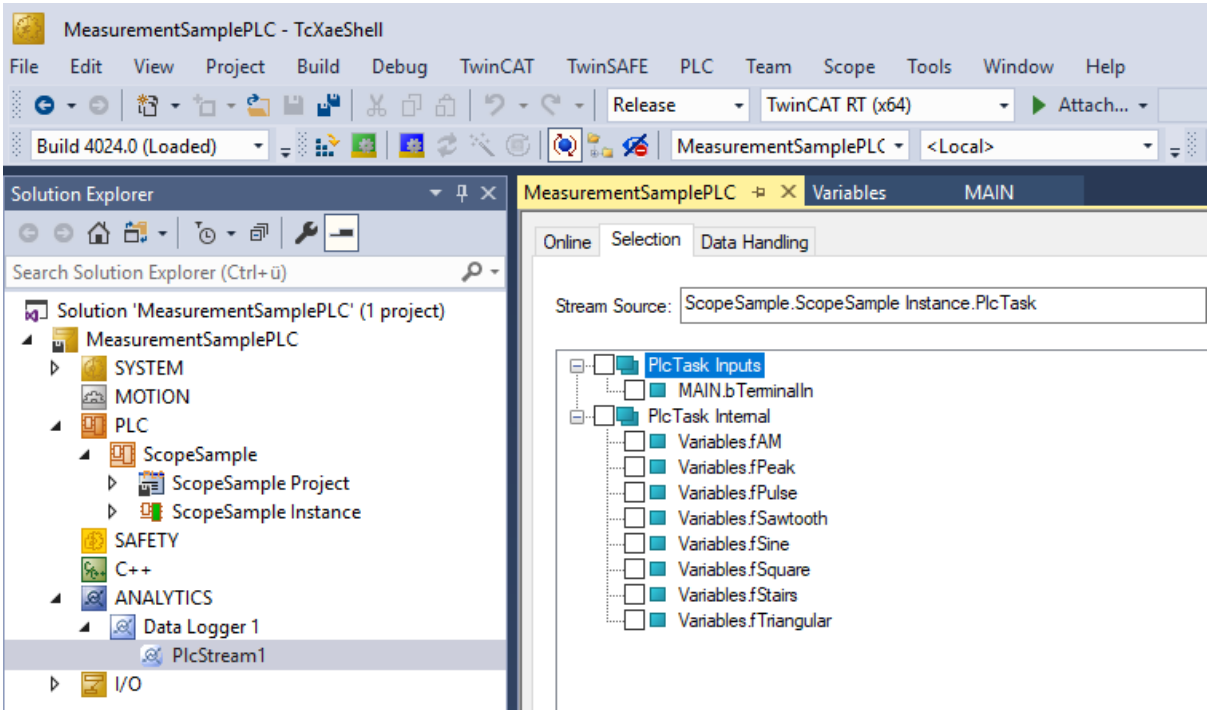
-トランスポート・レイヤー・セキュリティ（TLS）タブでは、セキュリティ設定ができます。TLSはクライアントとサーバ間の安全な通信チャンネルです。証明書の使用により、TCPポート8883はMQTT over TLS専用に予約されます。Analytics Loggerは、CA証明書、CA証明書＆クライアント証明書、および事前共有キー（PSK）モードをサポートしています。

- PLCアプリケーションの変数が{attribute 'TcAnalytics'} で宣言されている場合、自動的にData Loggerツリーノードの下にストリームとして表示されます。

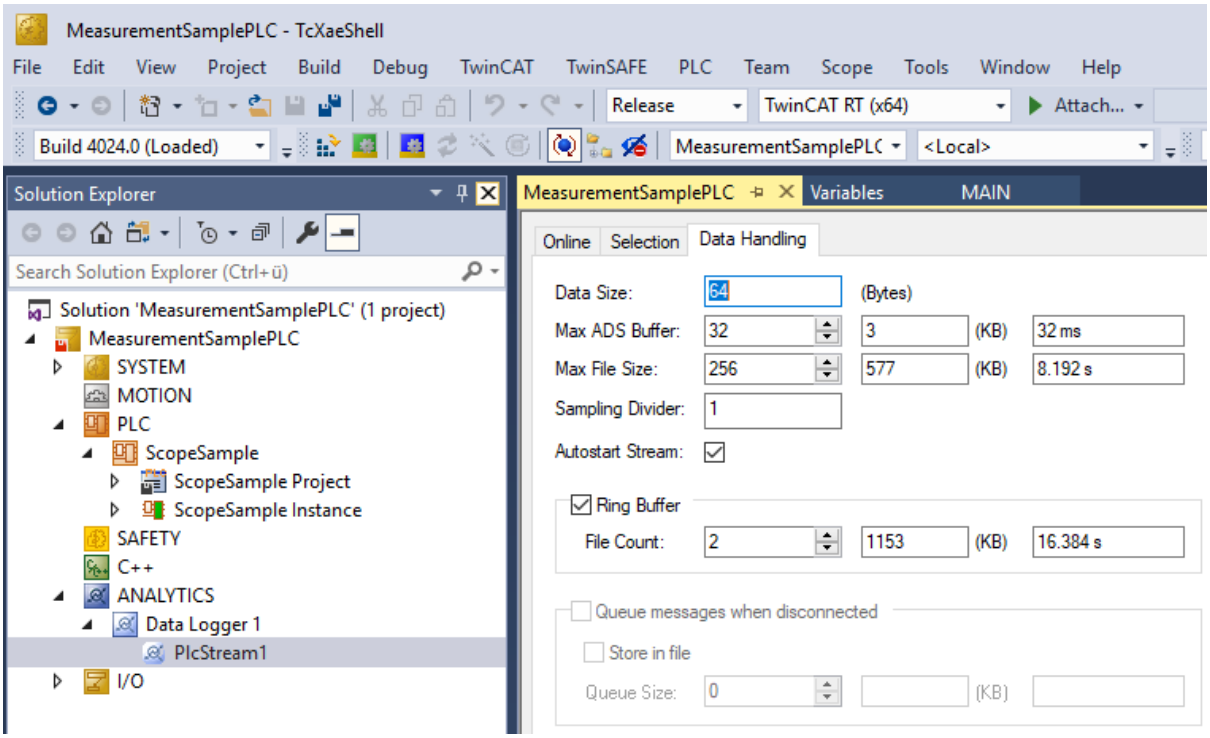


コンフィギュレーションがEtherCATプロセスイメージを提供する場合、追加のデバイスストリームが表示されます。

- ストリームでは、記録する変数を選択するためにSelectionタブを使用します。

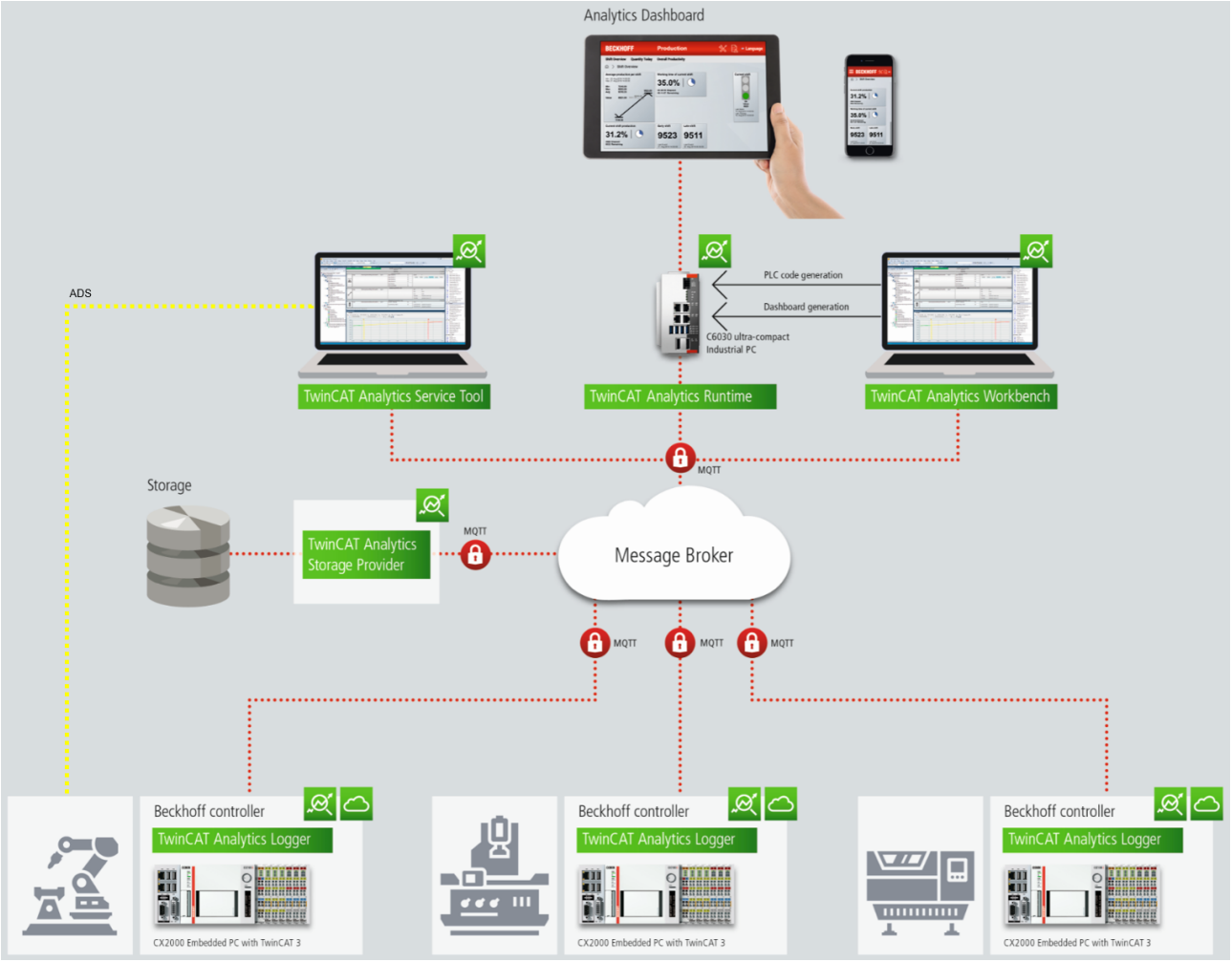


- 最後に、Data Handlingタブで、フレームのパッケージサイズを変更したり、通信切断やファイル用のリングバッファを設定できます。



4.2 通信

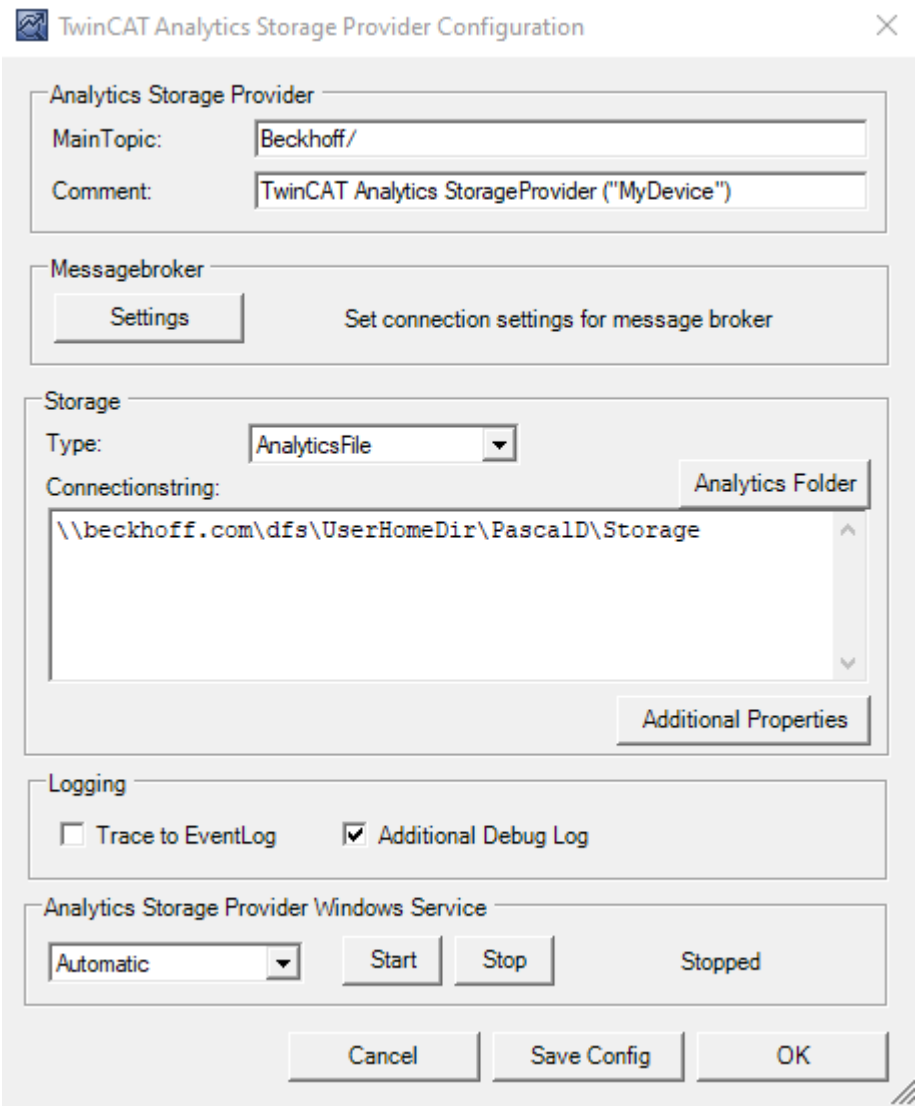
現在、解析ワークフローはMQTT経由で完全にマッピング可能です。エンジニアリングツールを使用して、ADSを介して機械データにアクセスし、解析を実行することもできます。



IoT通信プロトコルMQTTを選択した場合、ネットワークのどこかにネイティブのMQTTメッセージブローカーをセットアップする必要があります（クラウドシステムのVMも可能）。このメッセージブローカーは、解析ワークフロー内の各アプリケーションを疎結合にする役割を担っています。

4.3 データの履歴管理

TwinCAT Analytics Storage Providerをインストールした後、バックグラウンドで動作するサービスを設定できます。TwinCAT Analytics.StorageProvider.Configuratorアプリケーションは、フォルダ*C:¥TwinCAT¥Functions¥TF3520-Analytics-StorageProvider¥Tools*に格納されています。

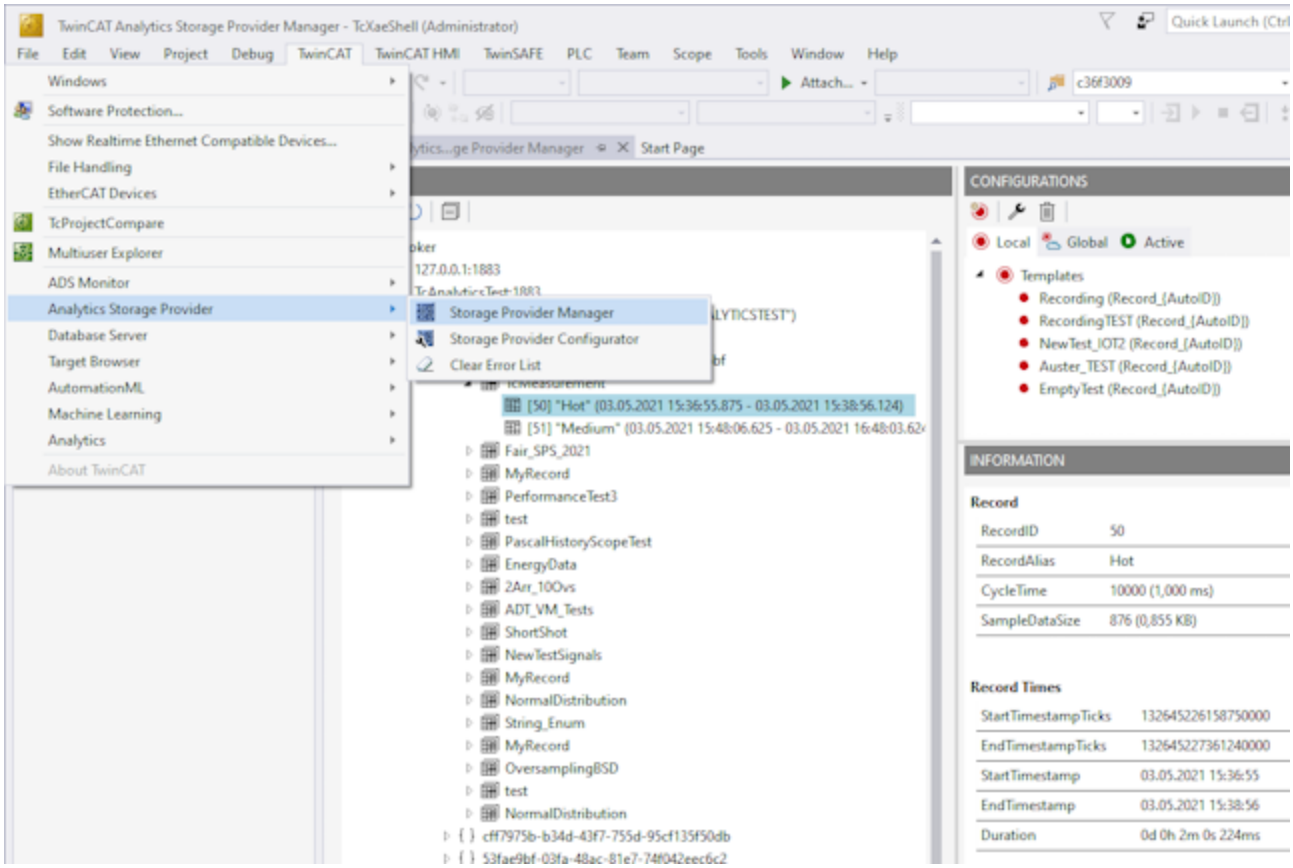


トピックの主要部分は、コンフィギュレーションファイルでコメントとともに定義できます。コメントは、複数のストレージプロバイダがメッセージブローカーに登録されている場合の識別に使われます。

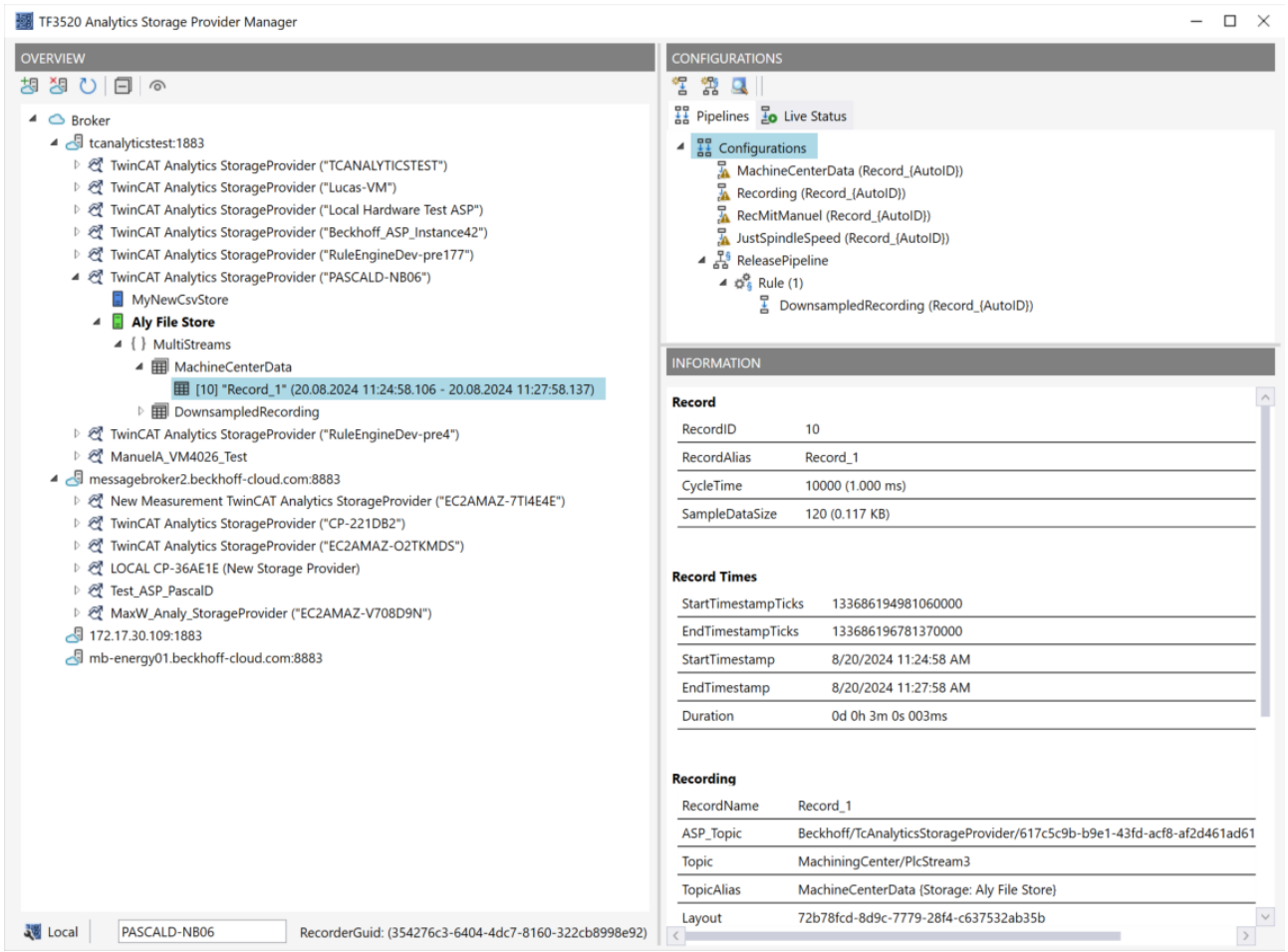
メッセージブローカーを設定し、ストレージの種類を決定します：

- Analyticsファイル（バイナリファイル）
- CSVファイル
- Microsoft SQL（バイナリ/プレーンテキスト）
- InlfuxDB（プレーンテキスト）
- Microsoft Azure Blob (Azure Cloudが必要)

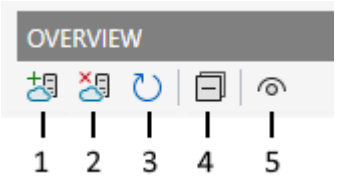
最後に、設定を保存してサービスを開始できます。次のステップでは、特定の記録方法を設定します。これには、開発環境で**Storage Provider Manager**を選択する必要があります。



Storage Provider Recorderでは、記録方法の定義や、記録の開始、管理ができます。さらに、個々のAnalytics Storage Providerのデータメモリを管理することも可能です。見つかったAnalytics Storage Providerと履歴データの全ての重要なプロパティが明確に表示されます。



Toolbar Manager ウィンドウ (OVERVIEW)

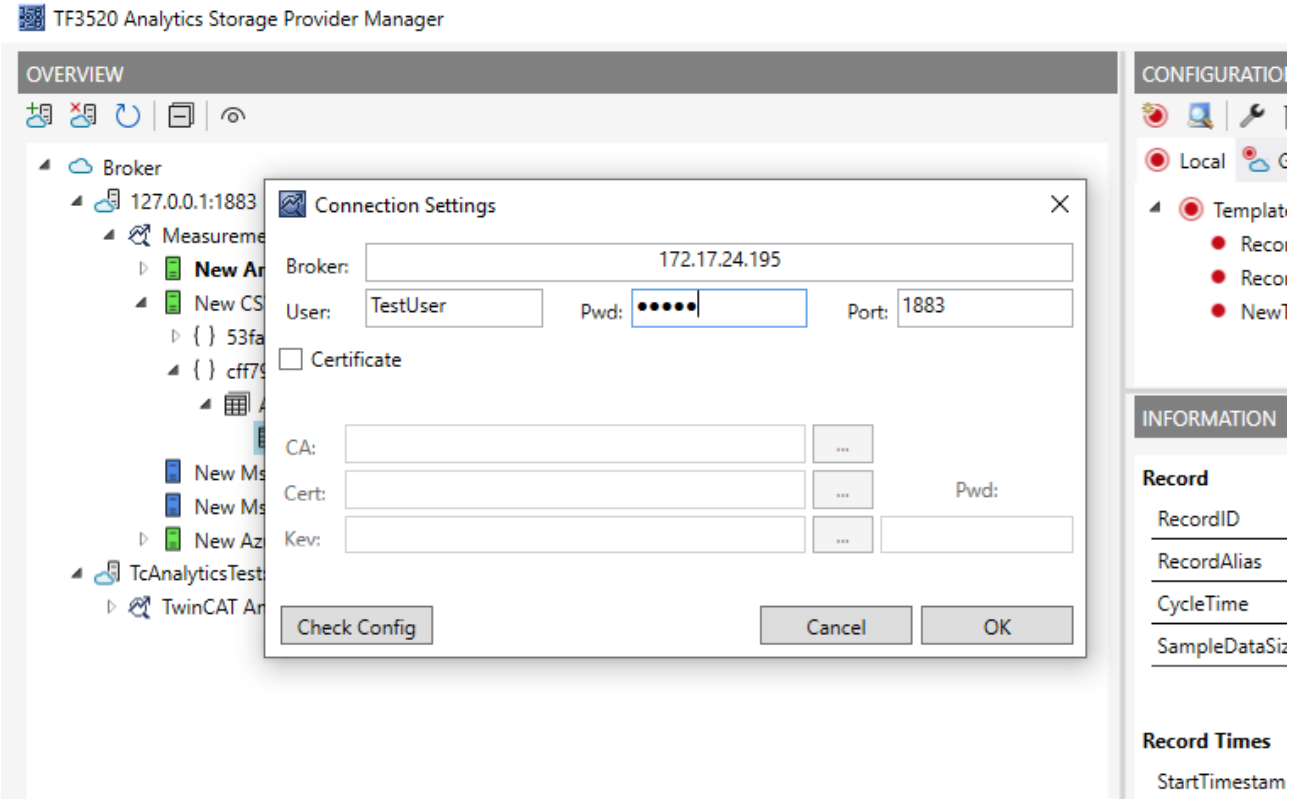


1	新規ブローカーの追加
2	選択したブローカーの削除
3	表示をリフレッシュ
4	全てのノードを折りたたむ
5	ダーク/ライトモードの切り替え

Function Manager ウィンドウ (OVERVIEW)

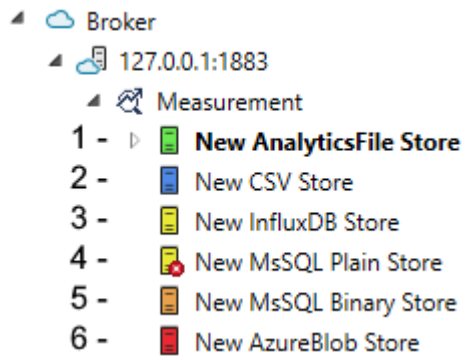
最初に**RecorderAlias**を割り当てます。これは、開始された記録をグループ化し、自動で開始された記録を再び見つけるのに役立ちます。

その後、複数のブローカーを設定できます。これは、既に知られているMQTT接続プロパティの入力マスク経由で行われます。



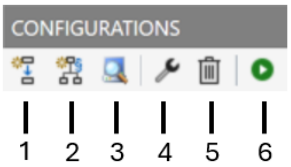
ブローカーへの接続が確立されると、ブローカーに接続されている全てのAnalytics Storage Providersが一覧表示されます。

ストレージのステータス



1	ストレージがオンライン
2	ストレージがオフライン
3	ストレージ起動
4	ストレージ起動がエラー （まだ起動途中）
5	ストレージがシャットダウン
6	ストレージがエラー状態

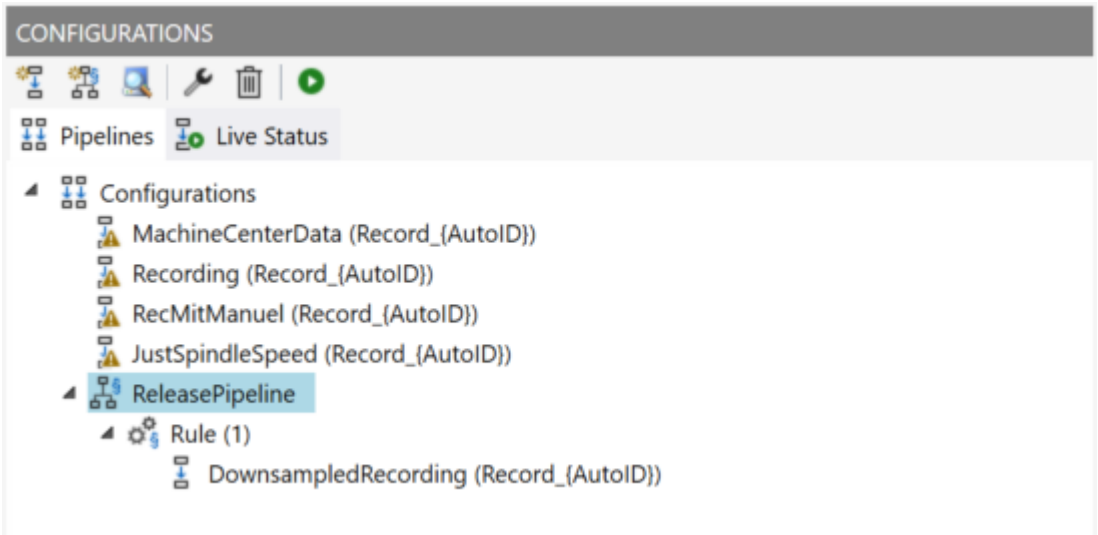
Toolbar Manager ウィンドウ (CONFIGURATIONS)



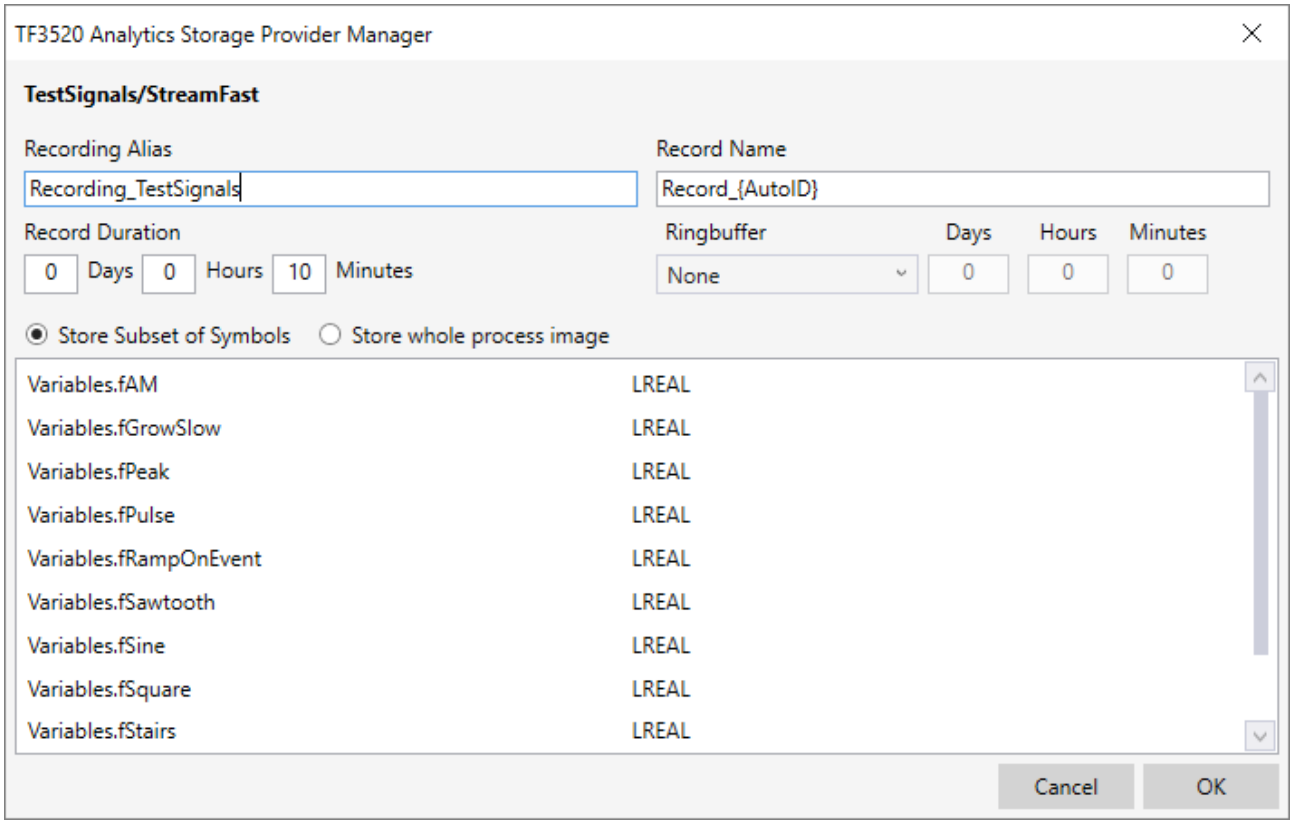
1	新規パイプラインを作成
2	Rule Engineで新規パイプラインを作成
3	シンプルなパイプライン接続のためTarget Browserを開く
4	選択したパイプラインを編集
5	選択したパイプラインを削除
6	選択したパイプラインを開始

Function Manager ウィンドウ (CONFIGURATIONS)

ウィンドウは2つのタブに分かれています。（PipelinesとLive Status）。Pipelinesタブの下でパイプラインの設定ができます。ここで新規パイプラインを定義できます。既存のパイプラインを編集したり 削除、開始できます。



シンプルなパイプラインを新規作成するには、「Create new pipeline」ボタンをクリックします。次のダイアログが開きます。



これで、記録したいシンボルをTarget Browserからダイアログにドラッグ＆ドロップできます。
Recording Alias（記録エイリアス）とRecord Name（記録名）も割り当てます。

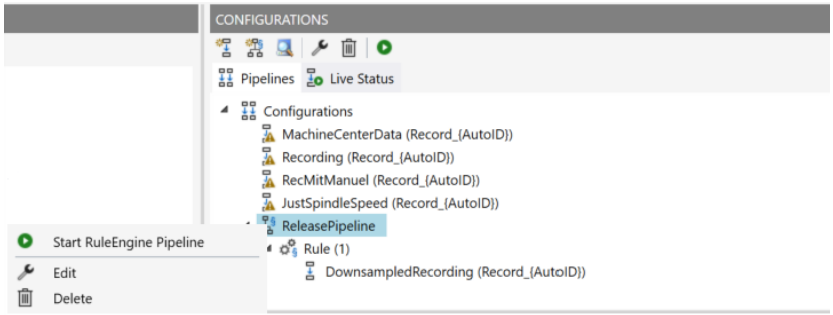
Record Nameには様々なプレースホルダが利用可能です：

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

記録名のほか、記録時間を設定できます（設定しないと、手動で停止するまで記録が継続します）。リングバッファは、ストレージ容量や記録時間に応じて設定します。

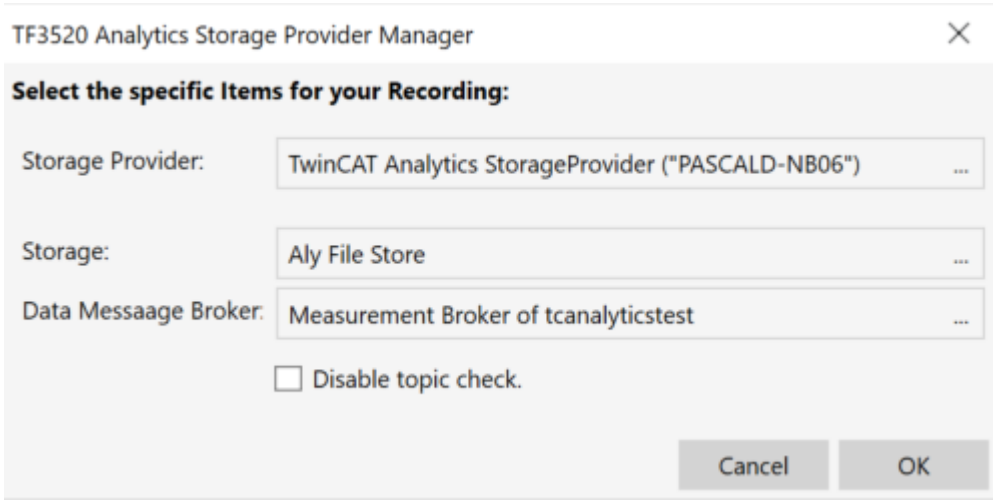
入力内容を**OK**で確定すると、新規のローカル記録設定が作成されます。

この設定は、ツールバーまたはコンテキストメニューから開始できます。



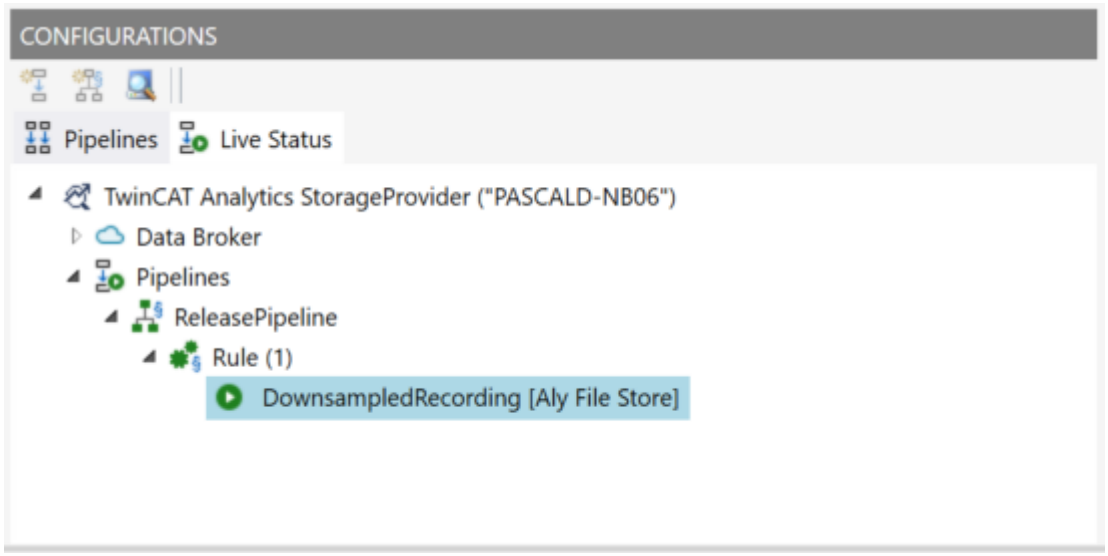
この設定をグローバルアクセス可能にすることもできます。これはコンテキストメニューの**Publish Recording**から設定できます。

次のダイアログが開きます。



ここで、設定をパブリッシュするAnalytics Storage Providerを選択します。さらに、この設定は、選択した Analytics Storage Provider のストレージおよびデータブローカーに割り当てられます。選択後、記録の設定は**OK**で確定され、選択したAnalytics Storage Providerにパブリッシュされます。この設定は、MQTTブローカーに接続されているStorage Provider Managerであればどこからでも見つけることができます。

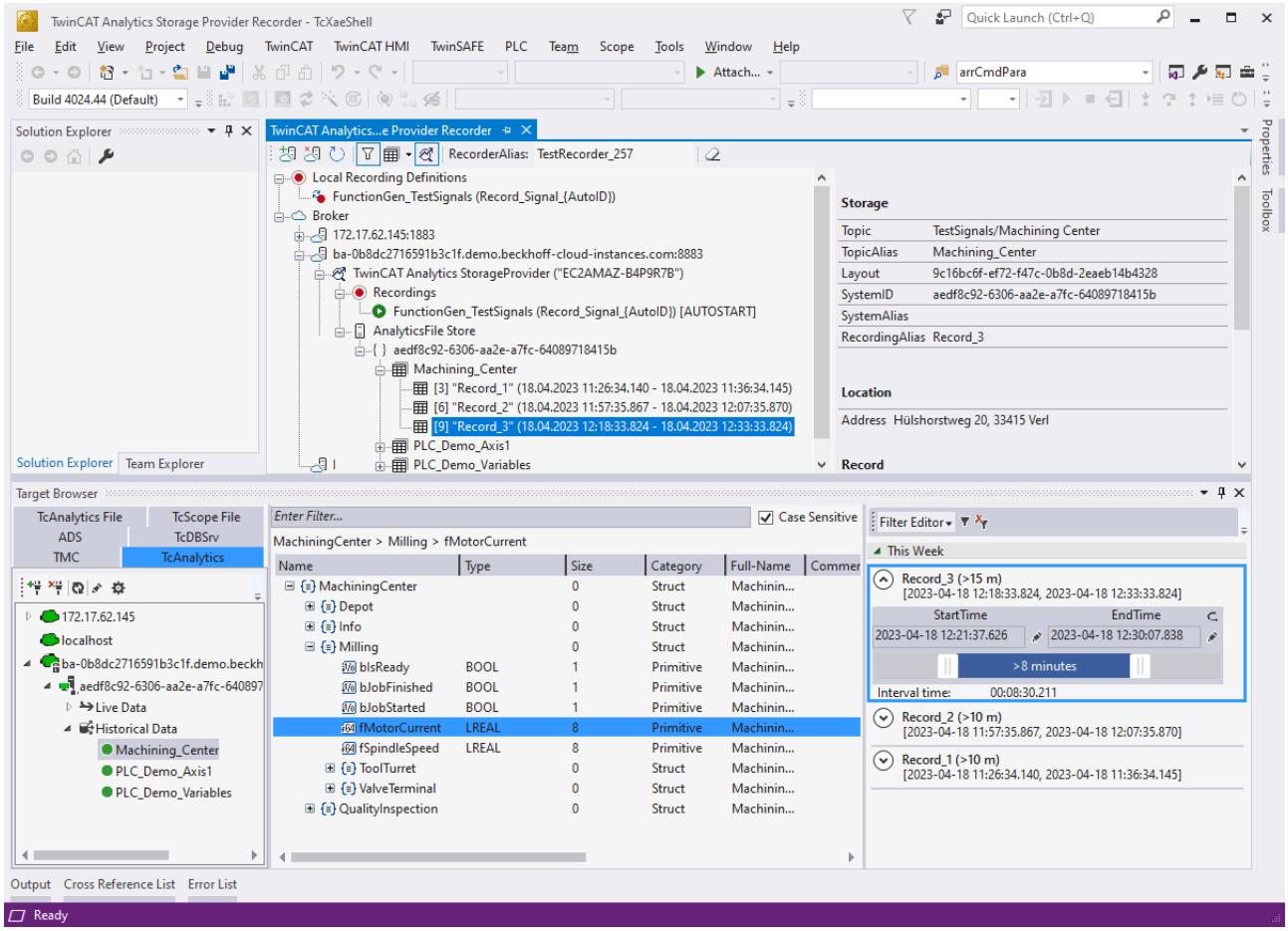
パイプラインを開始すると、表示は自動的に2番目のタブ、Live Statusにジャンプします。



全ユーザの有効な記録がここに一覧表示されます。このタブで記録の終了ができます。また、記録結果へジャンプすることもできます。

履歴データの使用

記録後や記録中に、Target Browserで解析用の入力として履歴データを選択できます。Target Browserの右側に、履歴データ用の新しいコントロールがあります。そこで、データのタイムスパンを選択できます。

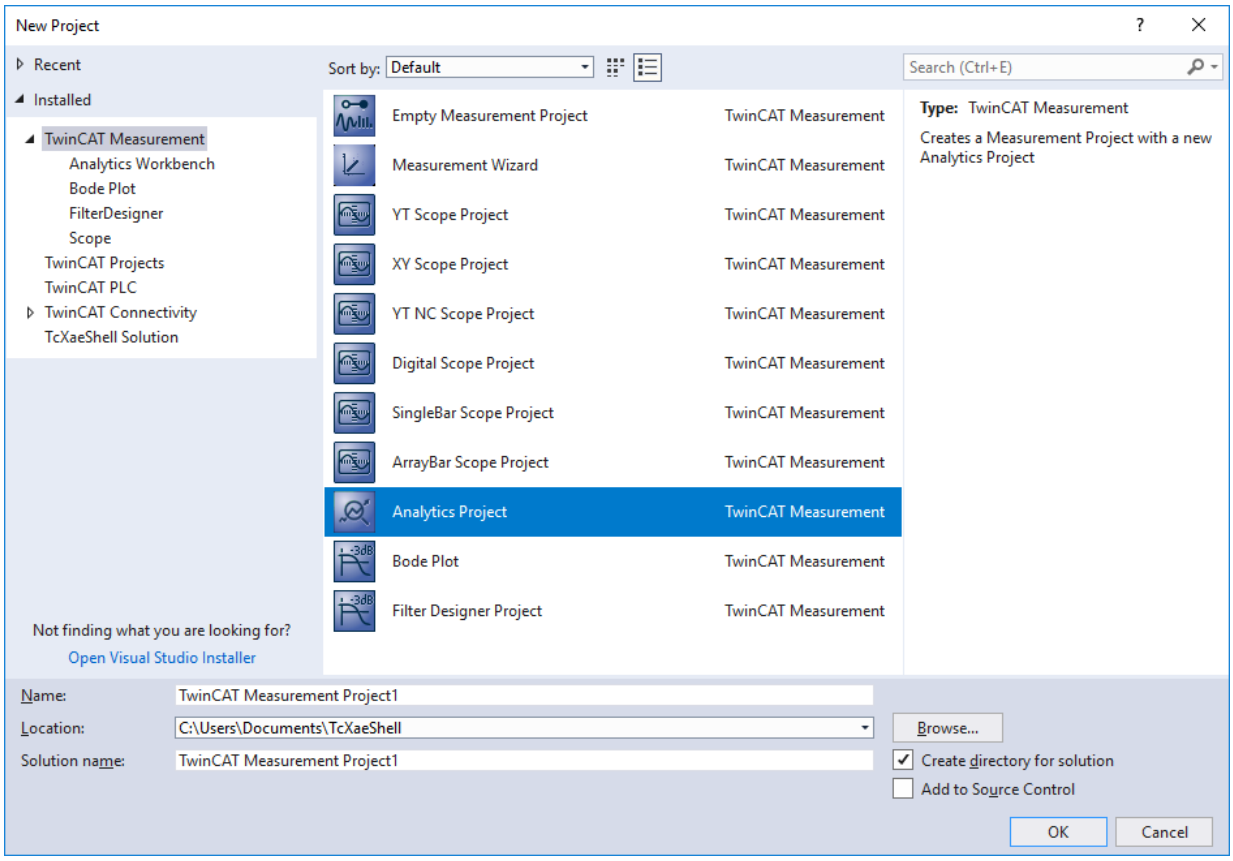


4.4 データの解析

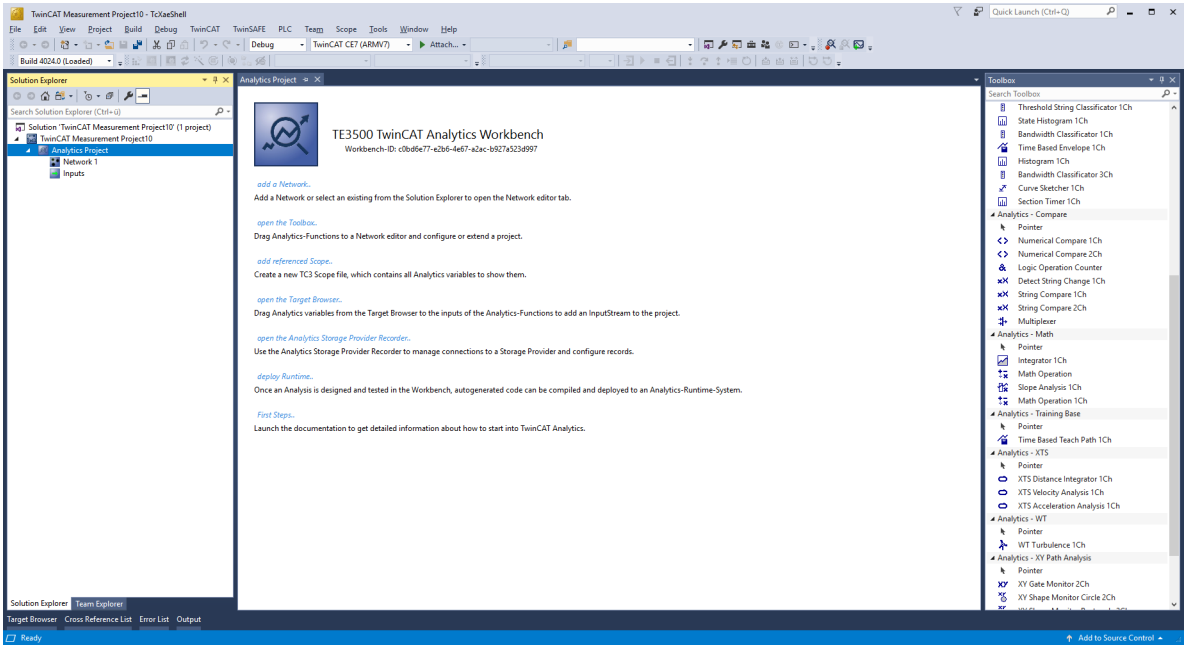
✓ TwinCATエンジニアリング環境を開き、データ解析を開始します。

1. **Visual Studio® > File > New > Project…**の順に開きます。

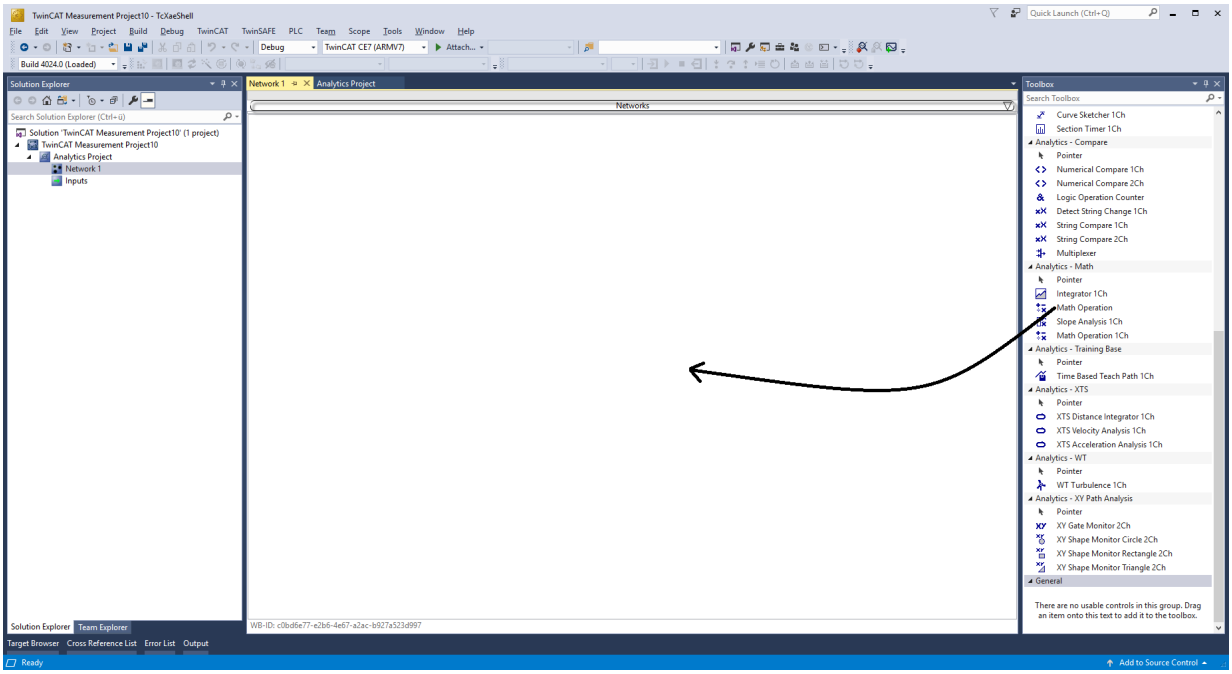
2. **TwinCAT Measurement**から **Analytics project template**を選択します。



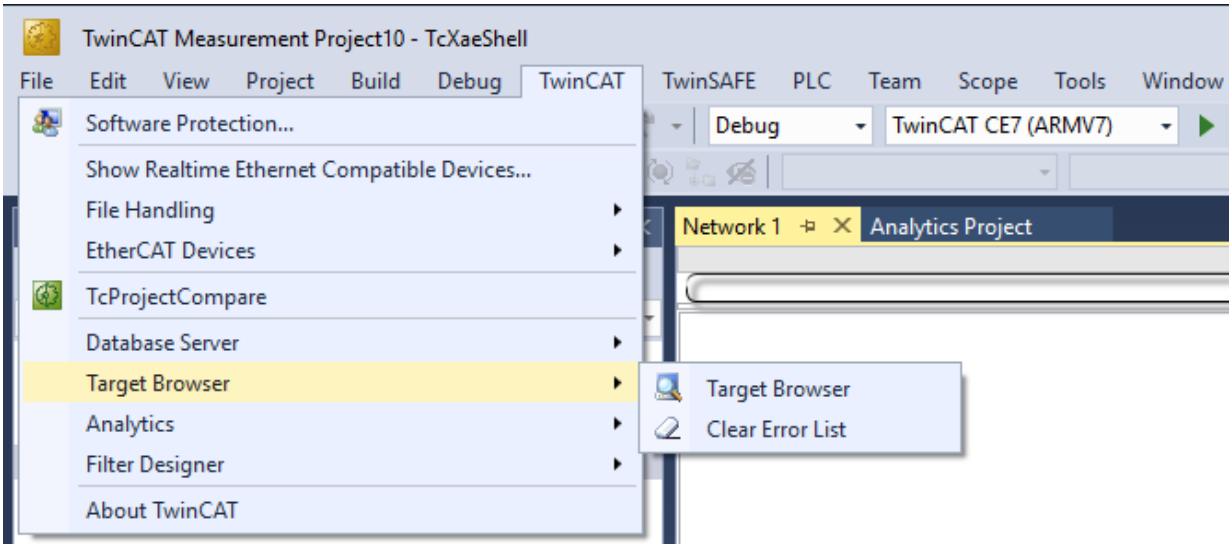
⇨ Solution Explorerに新規プロジェクトが表示されます。**Analytics Project**のツリー要素をクリックすると、開始ウィンドウが開き、最初のアクションを選択できます。ここから、ネットワークを追加したり、**Toolbox**、**Target BrowserAnalytics Storage Provider Recorder**を開いたりできます。以下のステップで、これら全てのアクションを実行します。



3. まず、Visual Studio®の**Toolbox**を開きます。TwinCAT Analyticsがサポートする全てのアルゴリズムが表示されます。アルゴリズムはネットワークごとにグループ化、整理する必要があります。**Analytics Project**を右クリックして新規ネットワークを追加するか、スタートページを使用してネットワークを追加します。最初のネットワークは常にデフォルトで作成されます。

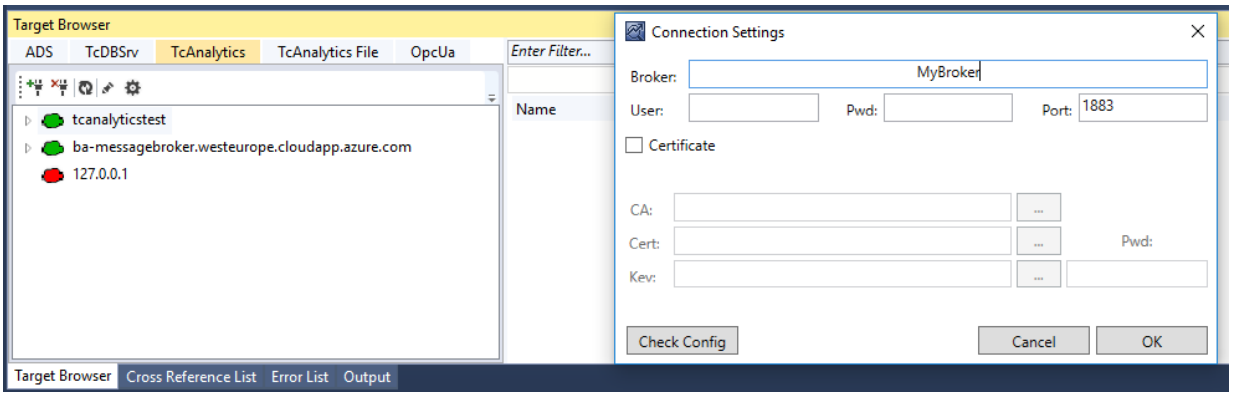


4. ネットワークをクリックすると、エディタが開きます。これで、目的のアルゴリズムをエディタのインターフェースにドラッグ＆ドロップできます。
5. アルゴリズムを選択したら、入力変数をモジュール（アルゴリズム）に接続する必要があります。これを行うには、**Target Browser**を開きます。
TwinCAT > Target Browser > Target Browser
の順に開きます。

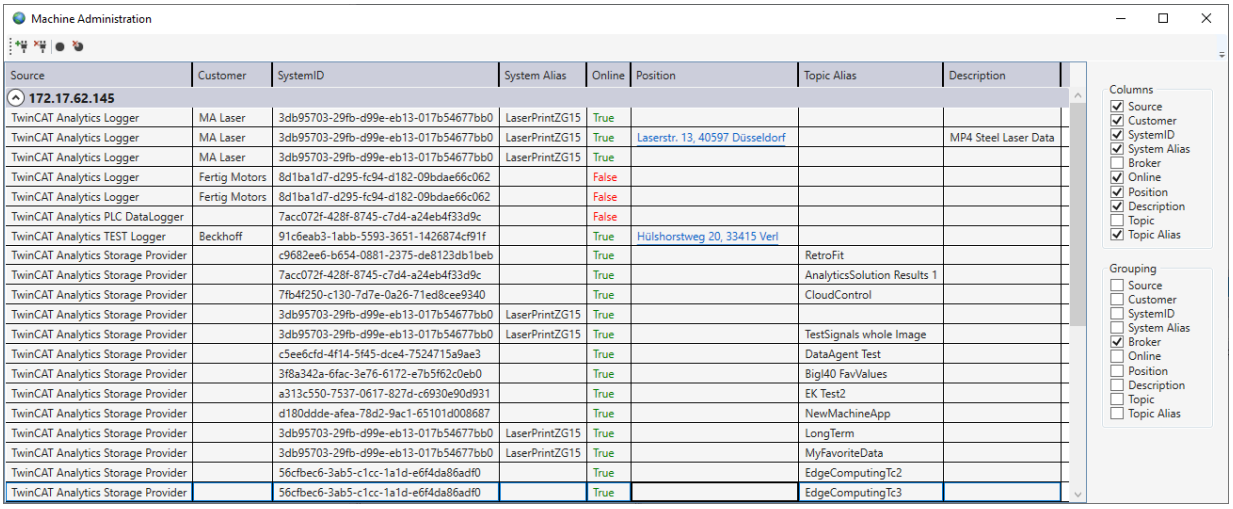


6. Target Browserで**TcAnalytics**タブまたは**TcAnalyticsFile**タブを選択します。**TcAnalytics**(MQTT)タブに進みます。

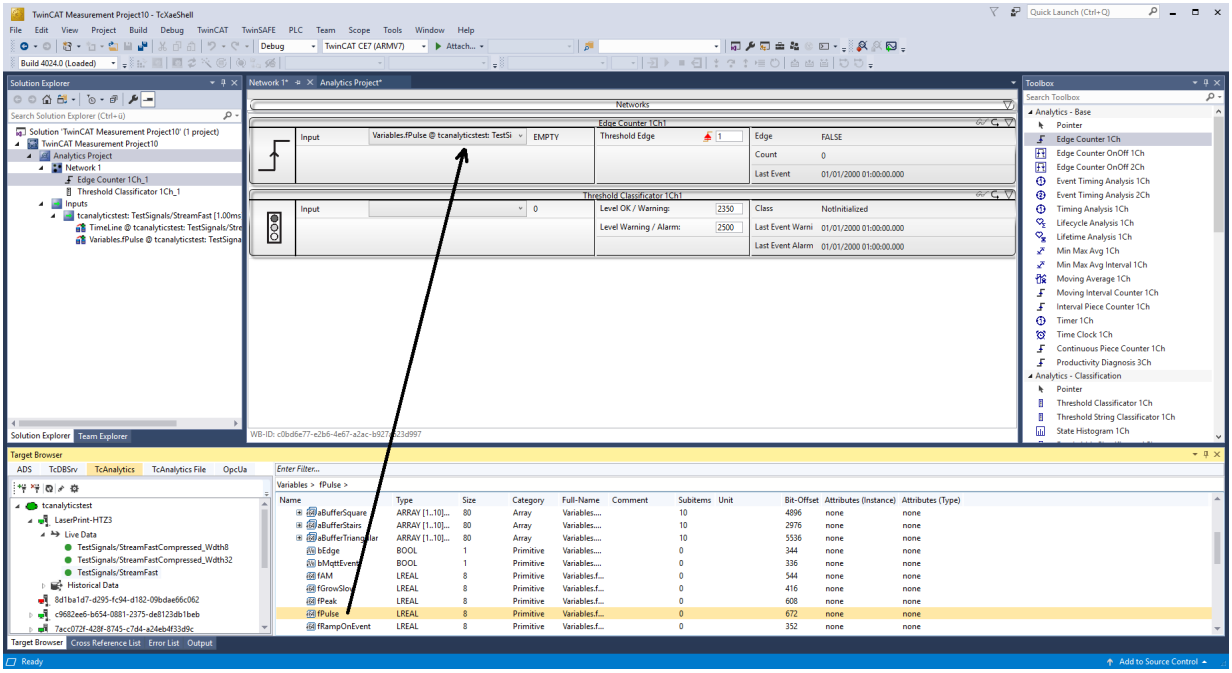
7. Analyticsエクステンションのツールバーで、緑色でハイライトされているアイコンをクリックします。メッセージブローカーの接続データを指定するウィンドウが開きます。



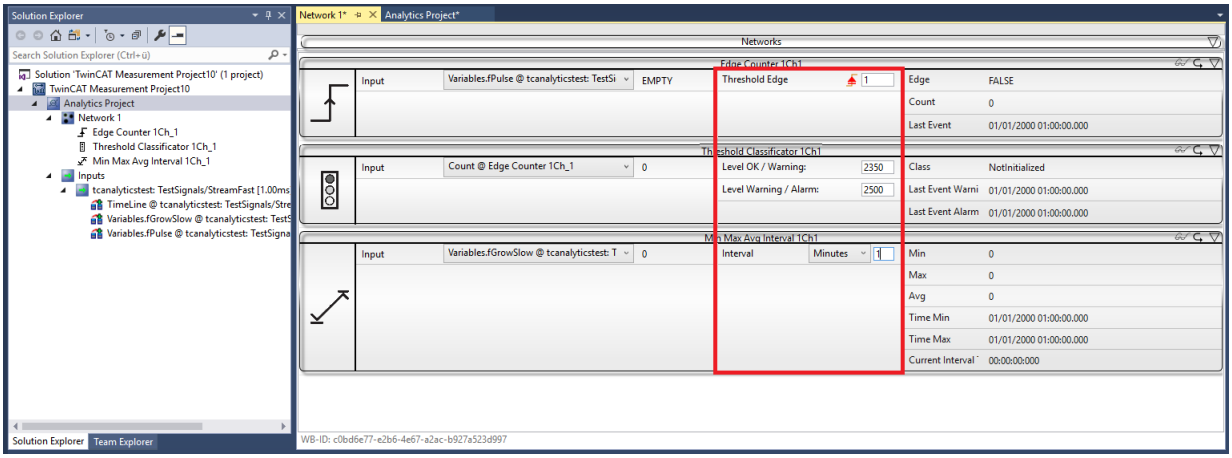
8. MQTT Analyticsクライアント（TwinCAT Analytics Logger、TwinCAT IoT Data Agent または Beckhoff EK9160）を選択します。各コントローラには固有のIDがあります。このIDはTarget Browserに表示されます。
9. **歯車アイコン**をクリックすると、Machine Administration（機械管理）ページに移動します。ここでは、IDの代わりにTarget Browserに表示されるシステムエイリアス名を割り当てることができます。



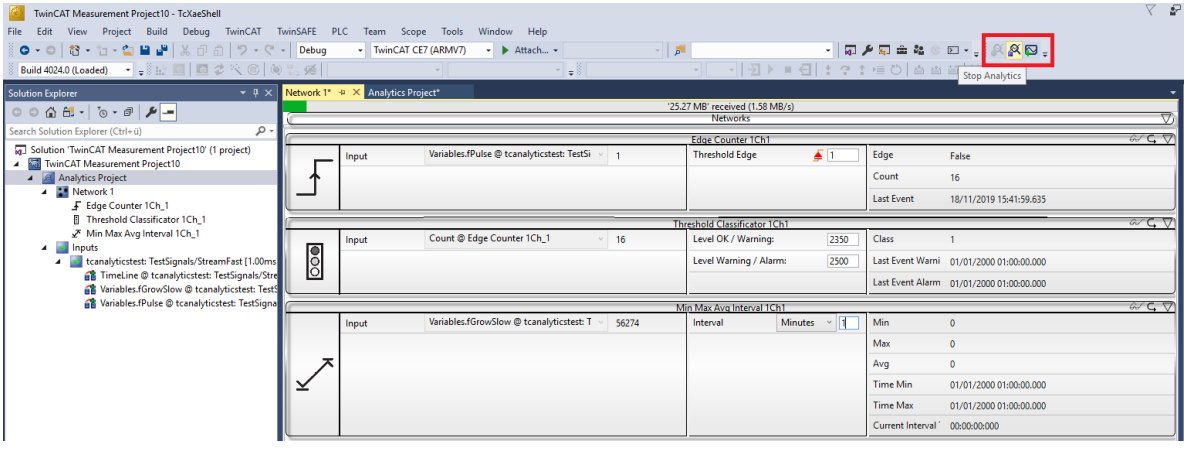
10. 次のステップでは、各MQTT Analyticsクライアントのライブデータと履歴データのいずれかを選択できます。この場合、履歴データはTwinCAT Analytics Storage Providerから提供されます。



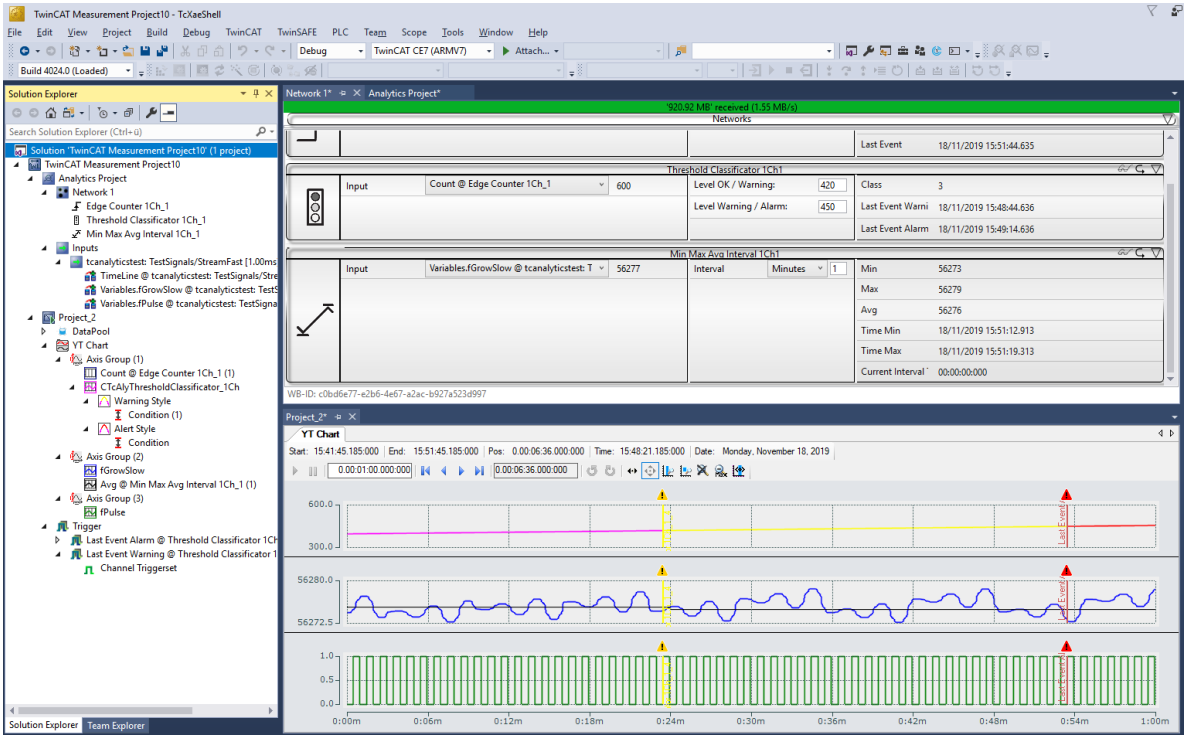
11. 変数を特定のアルゴリズムの入力にドラッグ＆ドロップできます。ほとんどのアルゴリズムで、閾値、時間区間、論理演算子などの条件を指定できます。これらは、各モジュール中で設定します。



⇨ これで、最初のAnalyticsプロジェクトの設定が完了しました。解析を開始するには、**Start Analytics**をクリックします。解析を停止するには、**Stop Analytics**をクリックします。



⇨ 解析の開始前や実行中に、**Add Reference Scope**ボタンをクリックできます。これにより、Analyticsプロジェクトに合ったScopeのコンフィギュレーションが自動生成されます。

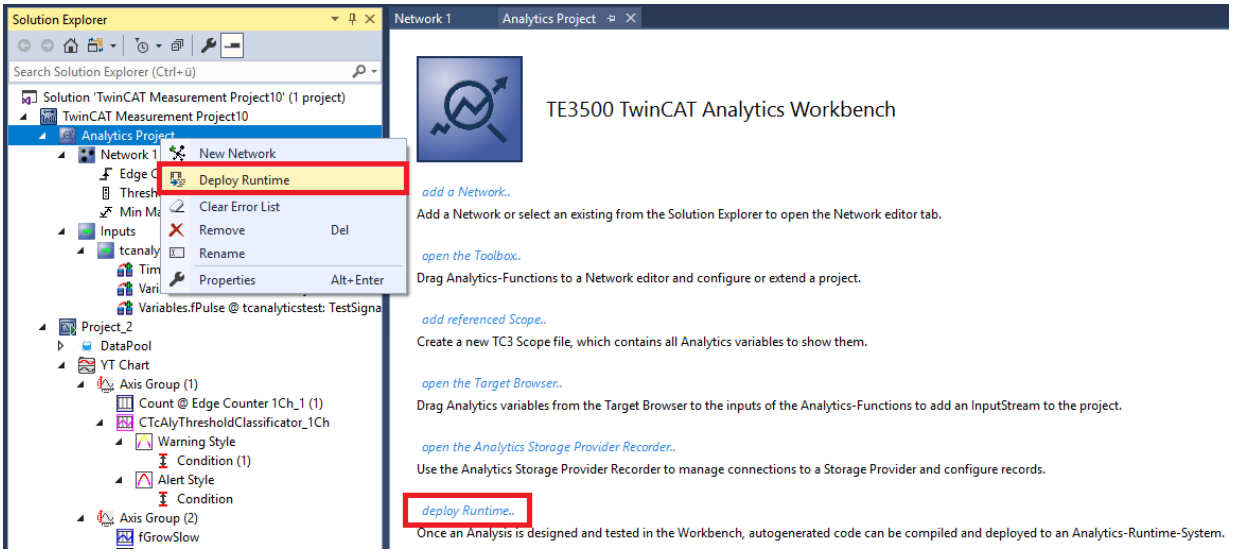


⇨ 解析結果は、ドラッグ＆ドロップでスコープビューのグラフに表示できます。例えば、平均値をビューの新規チャンネルとして表示できます。タイムスタンプは、X軸のマーカとして重要な値を示しています。

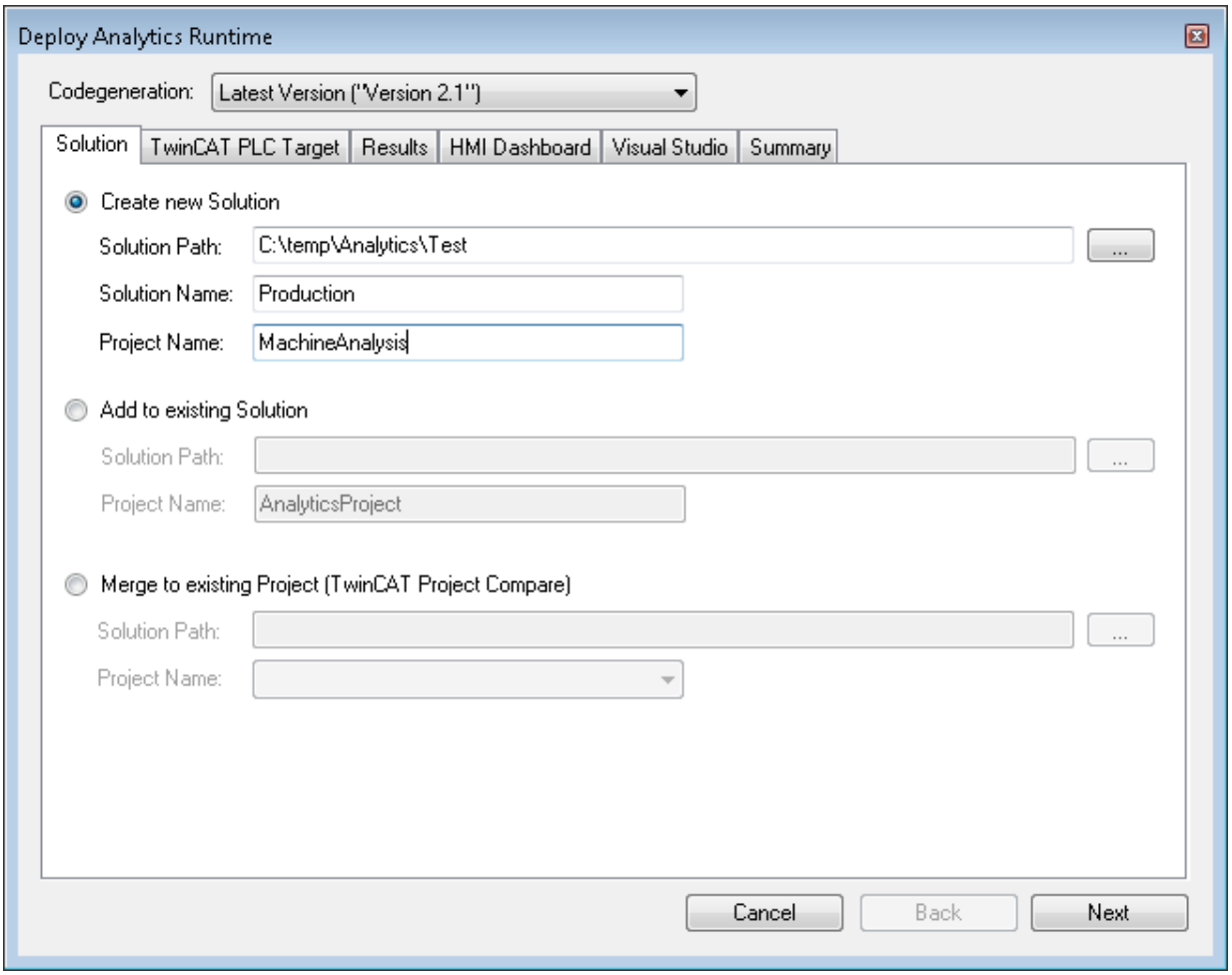
4.5 24時間解析アプリケーション

TwinCAT Analyticsのワークフローにおける最後の重要なステップは、24時間の継続的な機械解析です。これは、現場の機械アプリケーションと並行して実行します。これを簡単にするため、TwinCAT Analytics Workbenchでは、解析設定用のPLCコードとHTML5ベースのダッシュボードを自動生成できます。どちらもTwinCAT Analyticsランタイム（TC3 PLCおよびHMIサーバ）にダウンロード可能で、エンジニアリング環境のコンフィグレータツールと同じ解析結果を提供します。

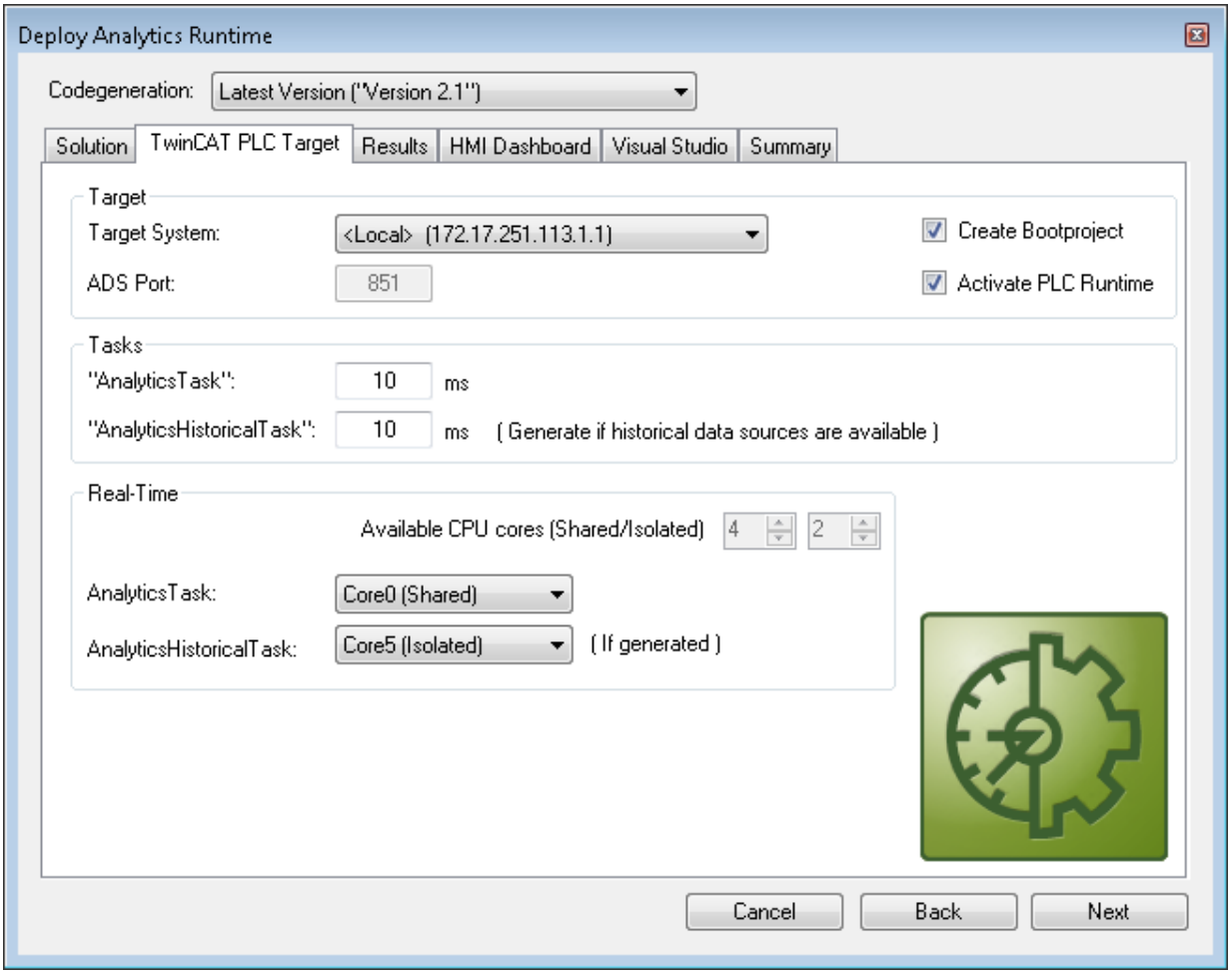
✓ まず、設定を保存し、Analytics Deploy Runtimeウィザードを開きます。これは、Analytics Projectツリー項目のコンテキストメニュー、またはスタートページから可能です。



1. ウィザードが開いている場合は、複数のタブをクリックして操作できます。最初のものはSolutionと呼ばれます。ここで、AnalyticsプロジェクトをPLCコードでどのように使用するかを決定できます：
   完全に新しいソリューションとして
   既存のソリューションの一部として
   既存のAnalyticsソリューションのアップデートとして

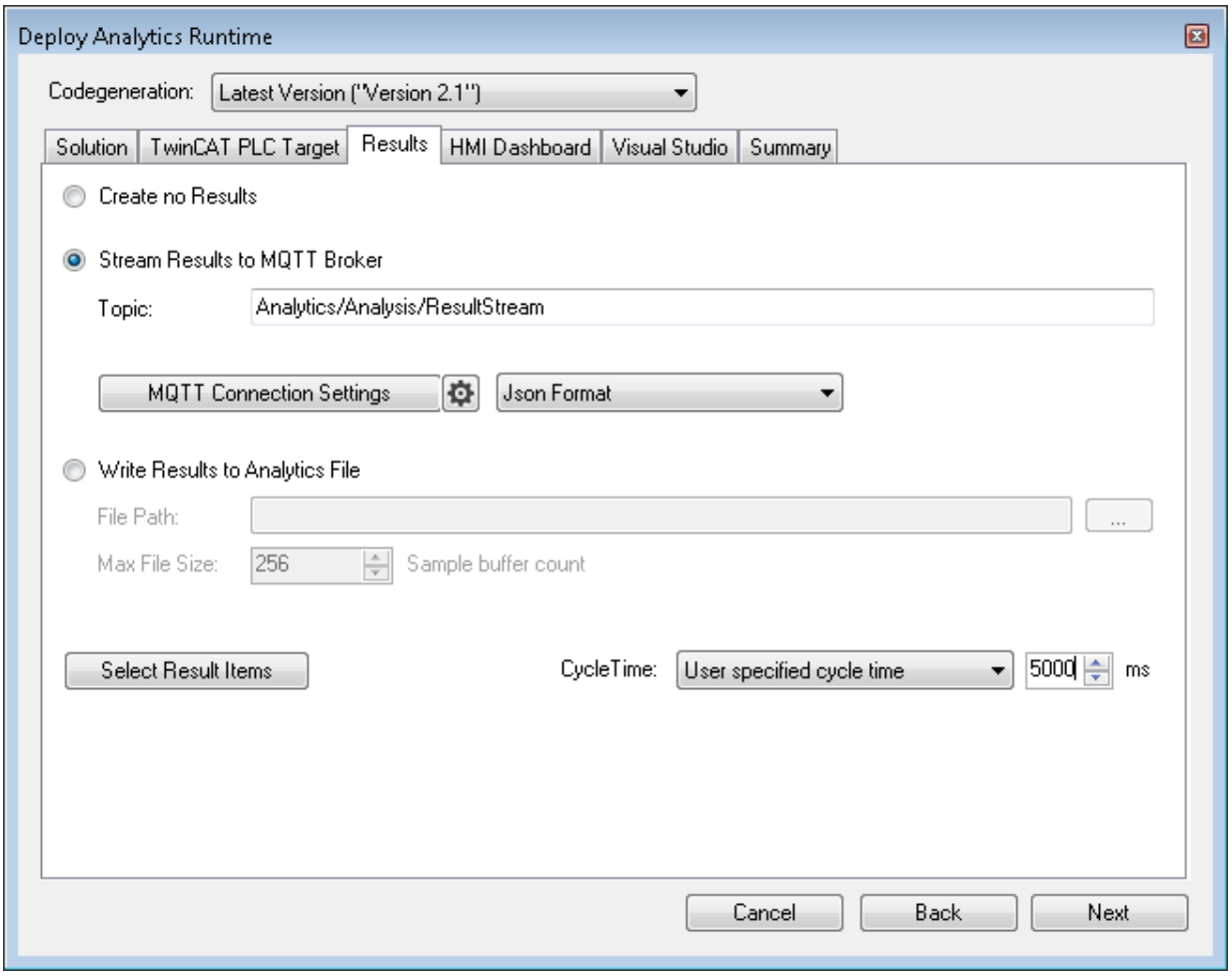


2. **TwinCAT PLC Target**タブで、TwinCAT Analytics Runtime（TF3550）を実行するADSターゲットシステムを選択します。作成されたプロジェクトはすぐに実行可能です。このために、Activate PLC Runtime（PLC ランタイムをアクティベート）のオプションを設定します。加えて、ブートプロジェクトを直接作成する方法も選択できます。

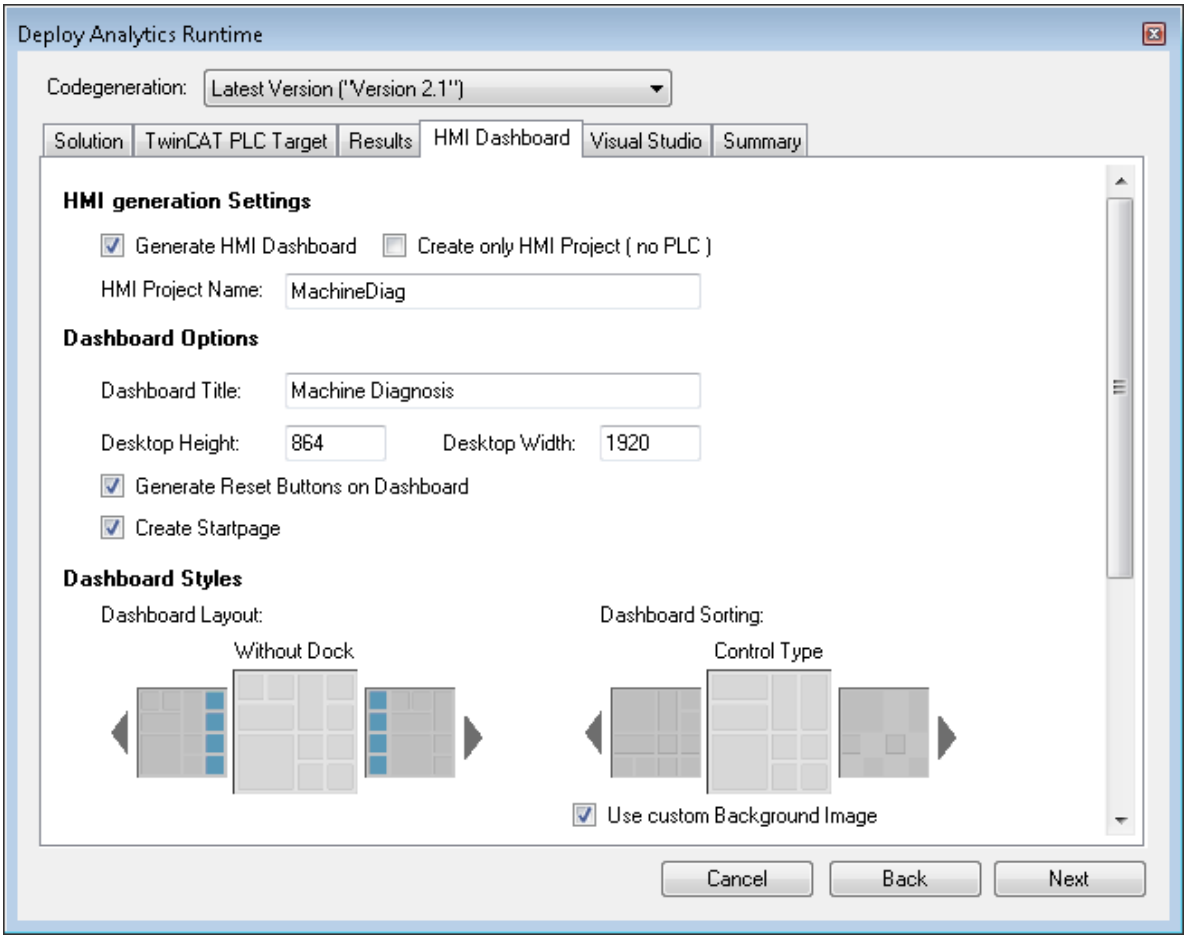


3. 特に仮想マシンの場合、独立したコアでプロジェクトを実行することが重要で、このタブにそのオプションがあります。次のタブ**Results**は、アルゴリズムのプロパティで**Stream Results**オプションを選択した場合のみ必要です。結果を送信したい場合は、どのような方法（ローカルにファイル/MQTT経

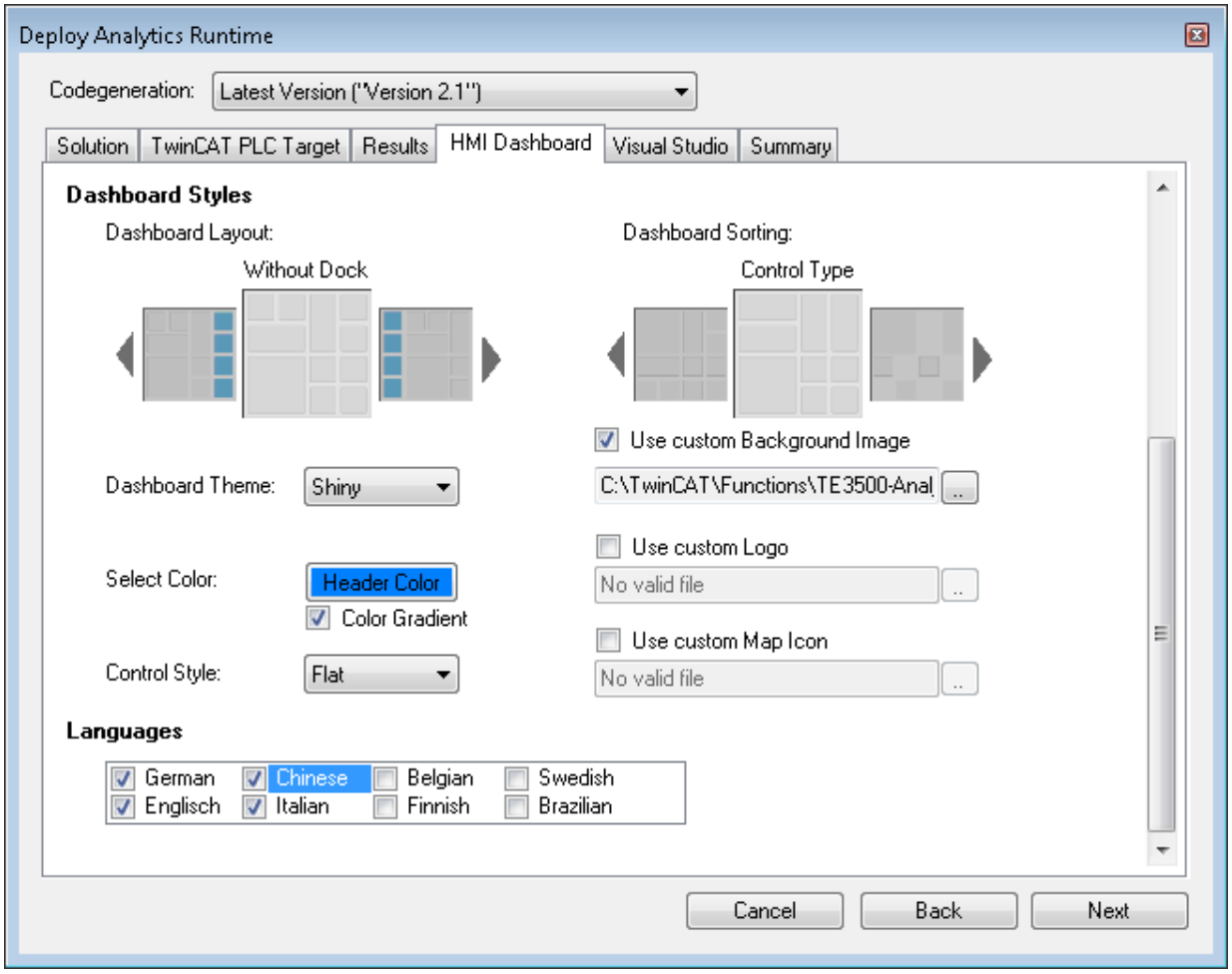
由）で、どのような形式（バイナリ/JSON）で送信するかを、ここで設定できます。アクティベーション直後に自動的に生成され、実行されます。



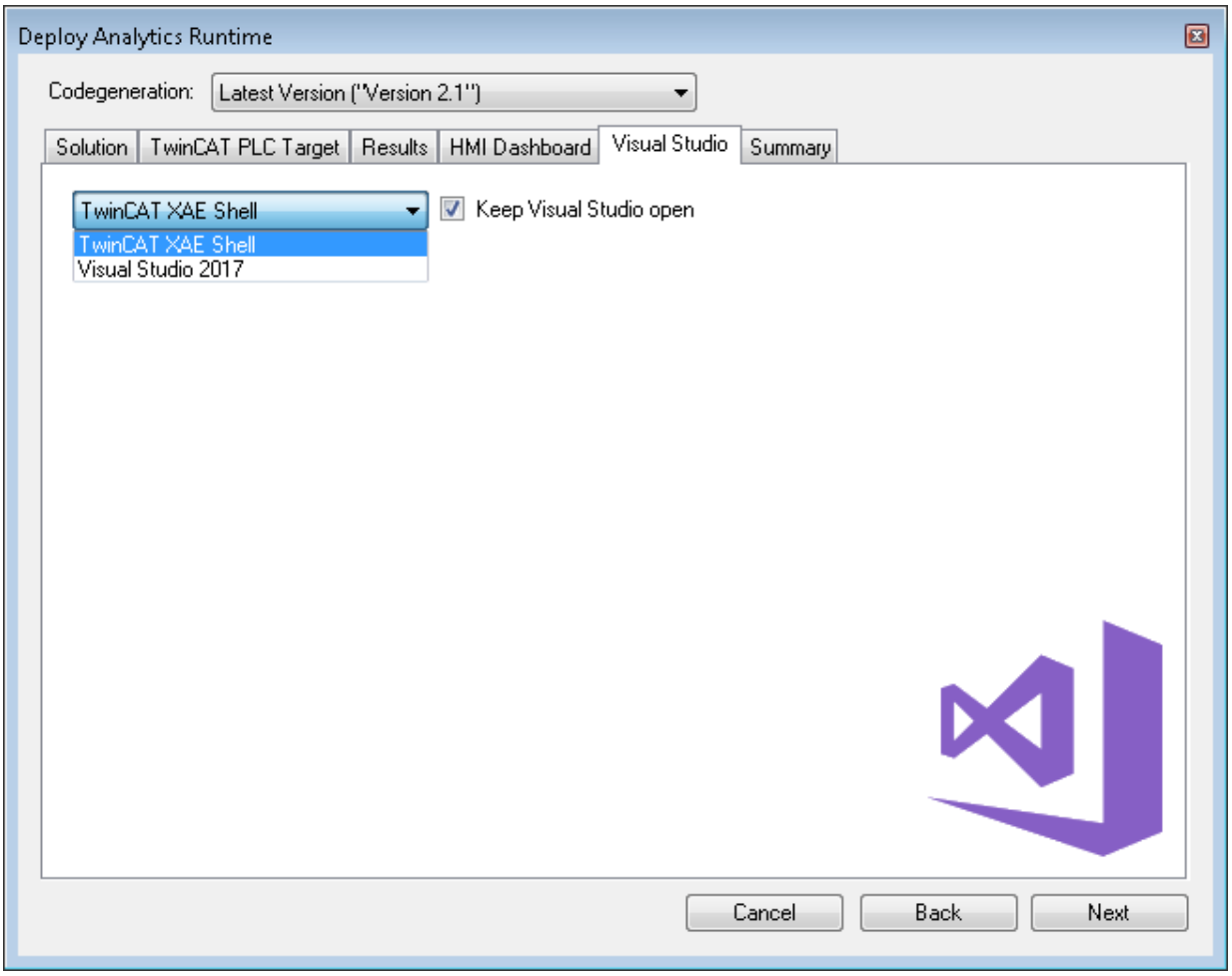
⇨ サイクルタイムを指定することで、結果のダウンサンプリングも可能です。次のタブは**HMI Dashboard**です。ダッシュボードを自動生成するための前提条件は、結果を表示したいアルゴリズムに対して、対応するHMIコントロールを選択することです。



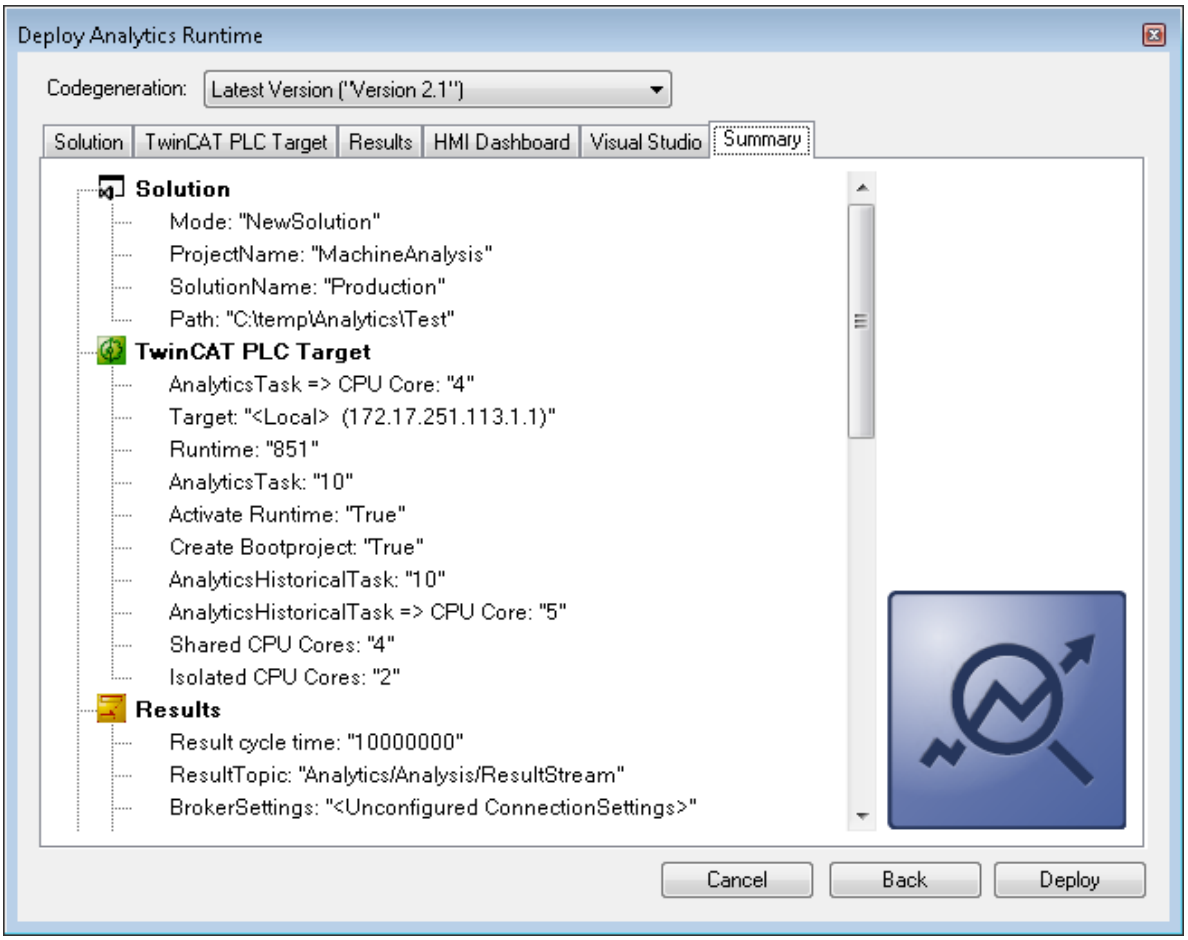
4. Analyticsダッシュボードでは、マップ付きのスタートページ、レイアウト、ソートアルゴリズム、カスタムカラー、ロゴなど、様々なオプションを選択できます。Analytics Controlsで複数の言語を選択した場合、言語切り替えメニューも生成されます。



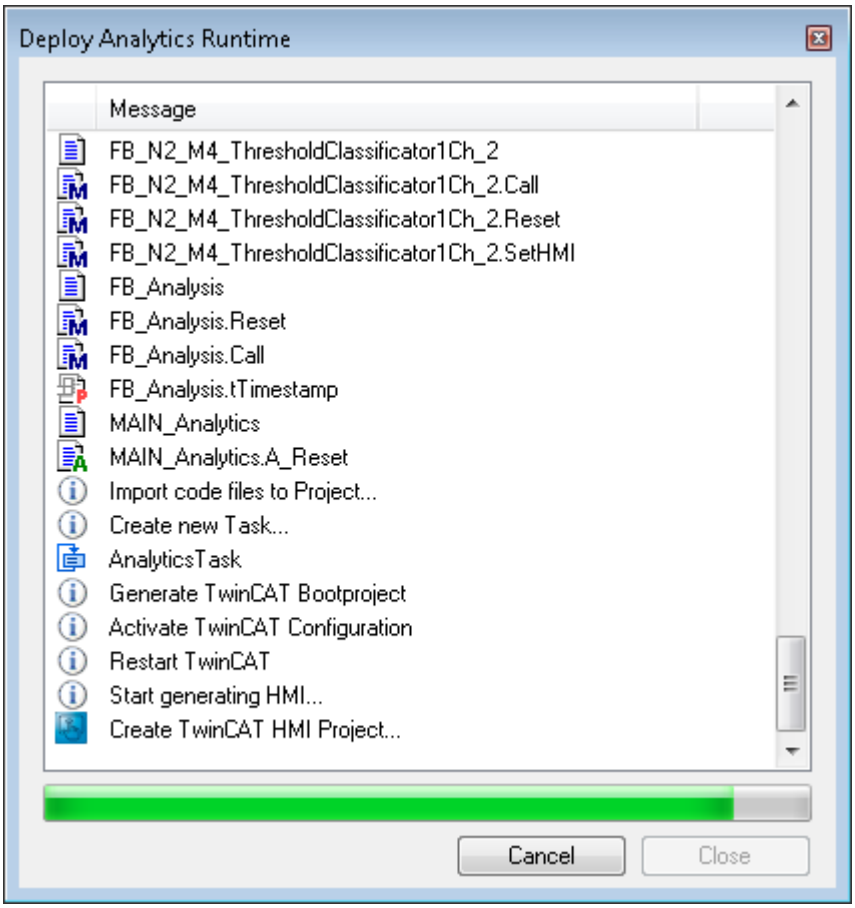
5. インストールされているVisual Studio®のバージョンのいずれかを選択し、インスタンスを目に見える形で起動するか、バックグラウンドでセットアップして起動するかを指定します。

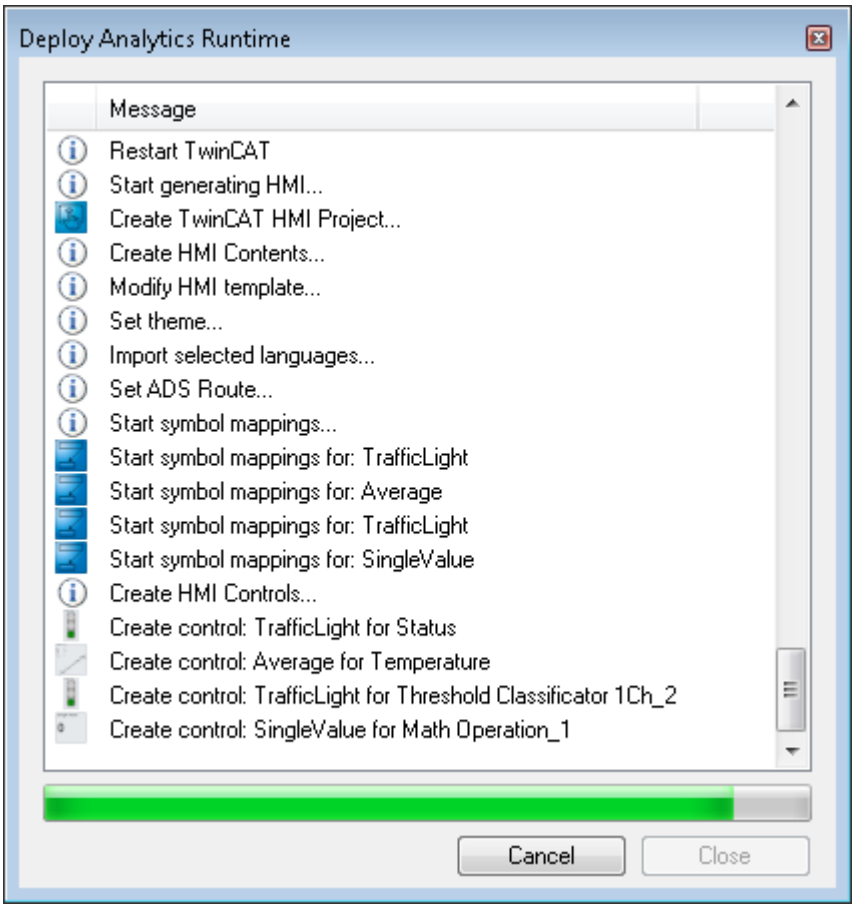


⇨ 最後に概要があります。

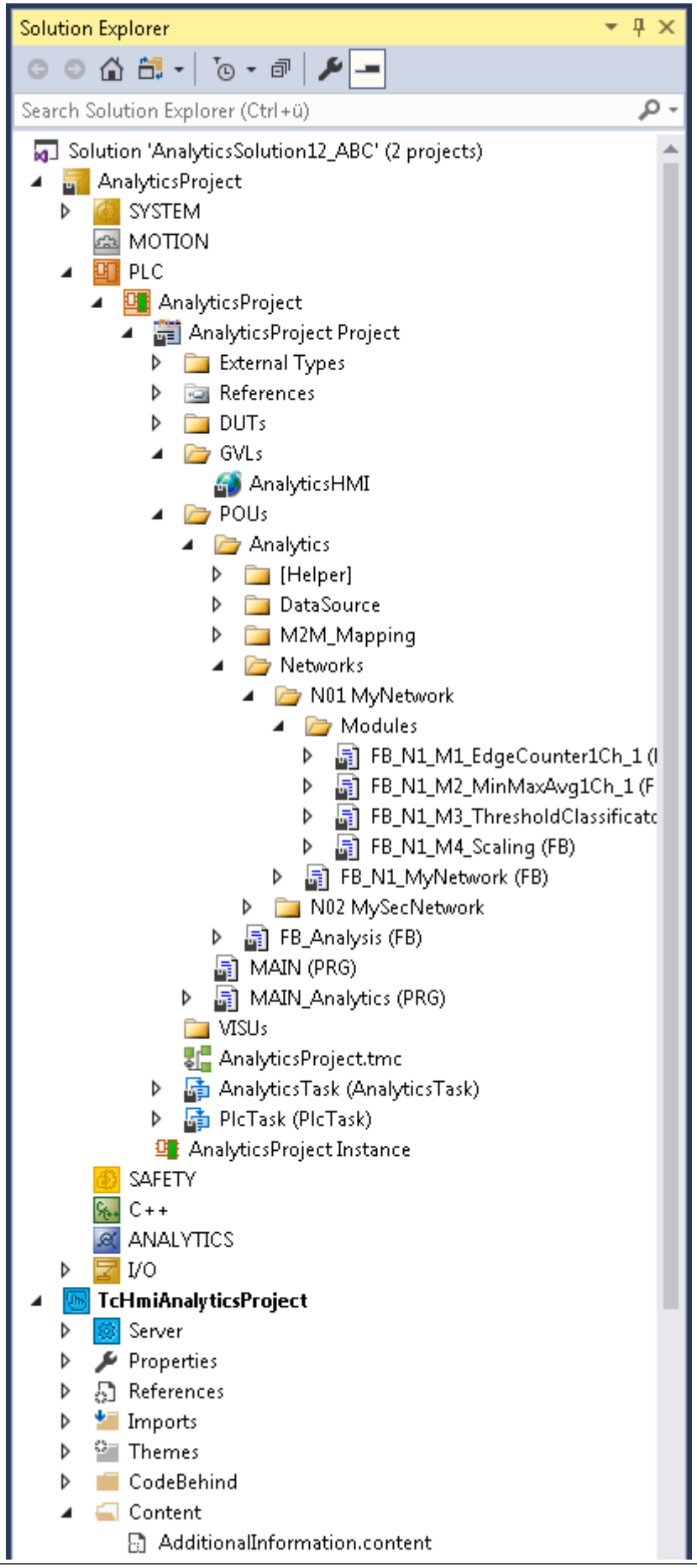

Deploy Analytics Runtime
Codegeneration:
Latest Version ("Version 2.1")
Solution
TwinCAT PLC Target
Results
HMI Dashboard
Visual Studio
Summary
Solution
Mode: "NewSolution"
ProjectName: "MachineAnalysis"
SolutionName: "Production"
Path: "C:\temp\Analytics\Test"
TwinCAT PLC Target
AnalyticsTask => CPU 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>"
Cancel
Back
Deploy

6. ここで**Deploy**ボタンをクリックして生成プロセスを開始します。PLCプロジェクトとHMIダッシュボードが生成されました。





⇨ 「Deploy Runtime succeeded（ランタイムのデプロイに成功しました）」のメッセージの後、デスクトップに新しいVisual Studio®/XAE shellのインスタンスが表示されます。新規ソリューションと両方のプロジェクトが作成されます。



5 技術概要

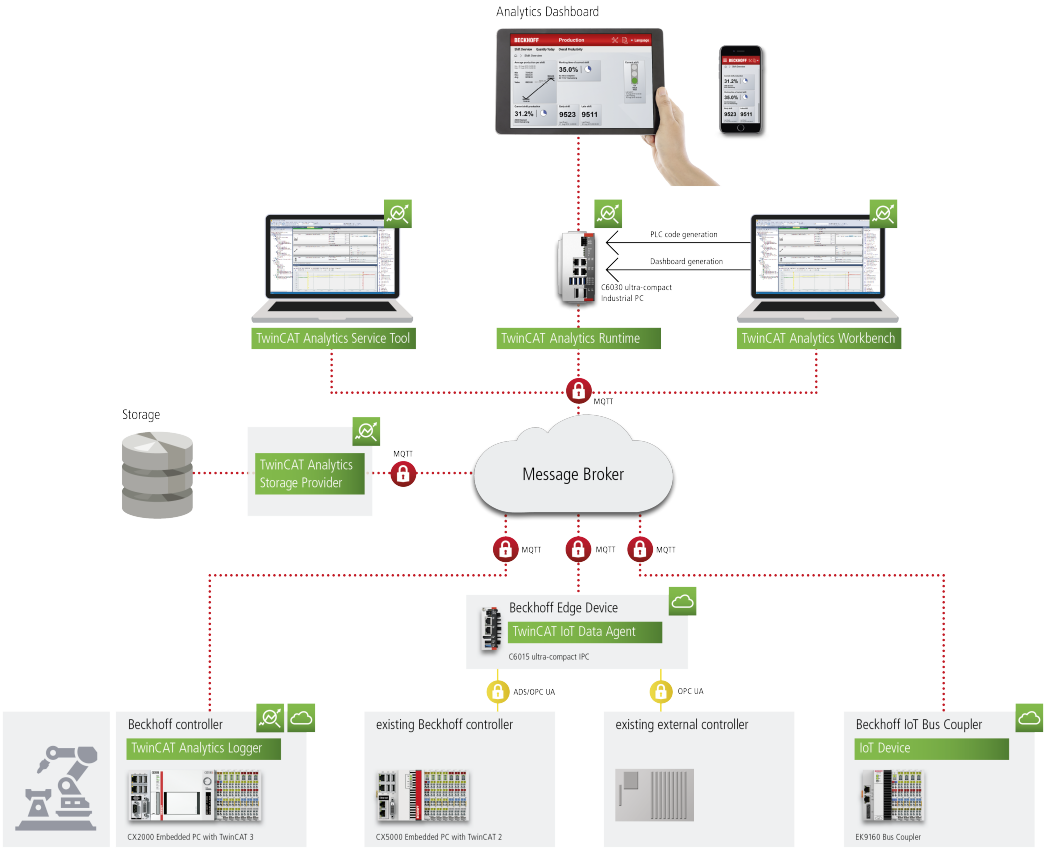
5.1 基本的なコンセプト

次の図は、データソースからAnalyticsダッシュボードまで、TwinCAT 3 HMIをベースにTwinCAT Analyticsの基本概念を示しています。Analyticsでの通信は、IoT通信プロトコルであるMQTTによって実現されます。

データソース

現在、TwinCAT Analyticsには3種類のデータソースがあります。これらのソースは全て、TwinCAT Analyticsの特定のバイナリ形式データと通信できます。この形式は、高い性能を達成するために必要です。

- TF3500 TwinCAT Analytics LoggerによるTwinCAT 3 制御
- TF6720 TwinCAT IoT Data AgentのゲートウェイによるTwinCAT 2、TwinCAT 3および外部制御
- 全てのEK9160 IoTバスカプラ



ストレージ:

TwinCAT Analyticsでは、ライブデータおよび履歴データを解析できます。TwinCAT Analytics Storage Providerは、ネイティブなMQTTメッセージブローカーと複数のストレージとの間のインターフェースです。TwinCAT Analyticsはストレージとして、Azure BlobストアとMicrosoft SQLデータベースをサポートしています。ストレージの設定は、Storage Providerによって自動的に行われます。したがって、通信を実装するために従来のSQLコマンドを使用する必要はありません。また、特別なテーブル構造の設定も不要です。

解析：

サービス技術者および機械の試運転向け

TE3520 TwinCAT Analytics Service Tool は、TwinCAT Analyticsのデータソースを解析する専門家に最適なツールです。Microsoft Visual Studio®に統合されています。ユーザは、グラフィカルなコンフィグレータで、様々なアルゴリズムの中から解析方法を設定できます。スコープビューとの並行動作も可能です。ユーザは、コンフィグレータからスコープビューのデータ・ストリームにドラッグ＆ドロップで、重要な値を簡単に見つけることができます。

継続的な常時機械解析

TE3500 TwinCAT Analytics Workbenchは、Service Toolと同じ機能を備えています。さらに、コンフィグレータの解析設定に基づいて、HMIダッシュボードを備えたPLCコードを自動生成することも可能です。PLCコードはすぐに使用できるので、コンフィグレータと同様に、即座にデータ解析を開始できます。現在は、必要に応じて常時稼働が可能です。自動生成されたコードは、TF3550 TwinCAT Analytics Runtimeにダウンロードできます。または、HMIダッシュボードなしで作成した純粋なPLC プロジェクトを、TF3551 Analytics Runtime Base（HMIサーバなし）にダウンロードすることもできます。どちらのランタイム製品も基本的にはライセンスバンドルで、従来のIPCや組込み型PC上だけでなく、仮想マシン上でも実行できます。

製品情報:

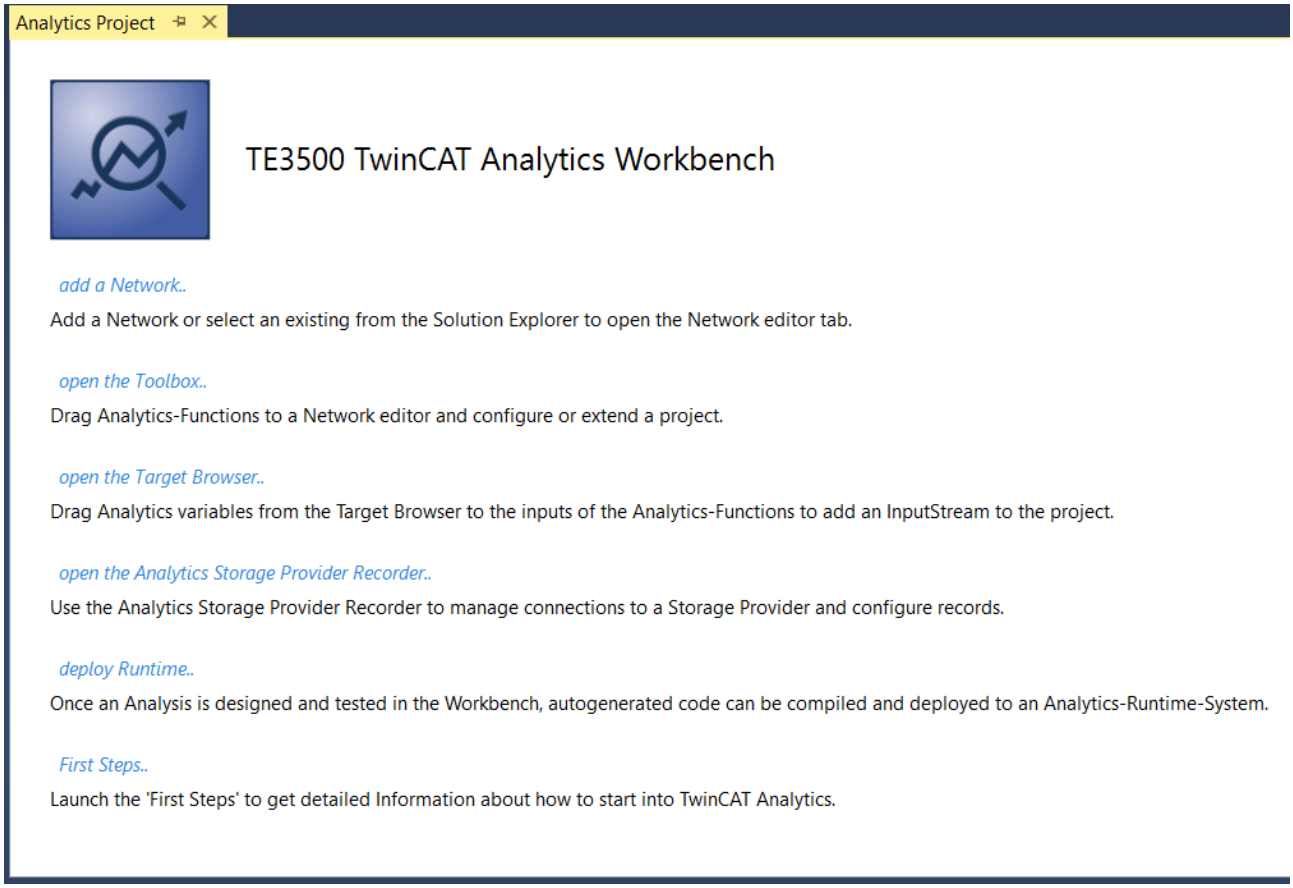
TwinCAT Analytics Workflowには、様々な単品の製品があります。以下の全製品リストを参照してください。

製品型番	製品型番
TE3500	Analytics Workbench
TE3520	Analytics Service Tool
TF3500	Analytics Logger
TF3510	Analytics Library
TF3520	Analytics Storage Provider
TF3550	Analytics Runtime - HMIサーバ/クライアントパックを含む
TF3551	Analytics Runtime Base - HMIなし
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カプラ

TwinCAT Analytics Service Toolは、ADSチャンネルを介してScope ++の一種として使用できます。これは、提示された基本コンセプトに基づいて、自動的に最適な最小構成が選択されます。Analytics Loggerは、データ収集と解析を疎結合にするためにも使用できます。ローカルの機械コンピュータにデータを保存できます。データはサービスツールを使用して評価できます。データストレージを分散させるのではなく、一元管理したい場合は、MQTT経由でAnalytics Storage Providerを使用できます。利用可能なデータソースはTF3500/TF6720/EK9160です。また、TwinCATシステムでTF3510 Analytics Libraryを使用することも可能です。

6 設定

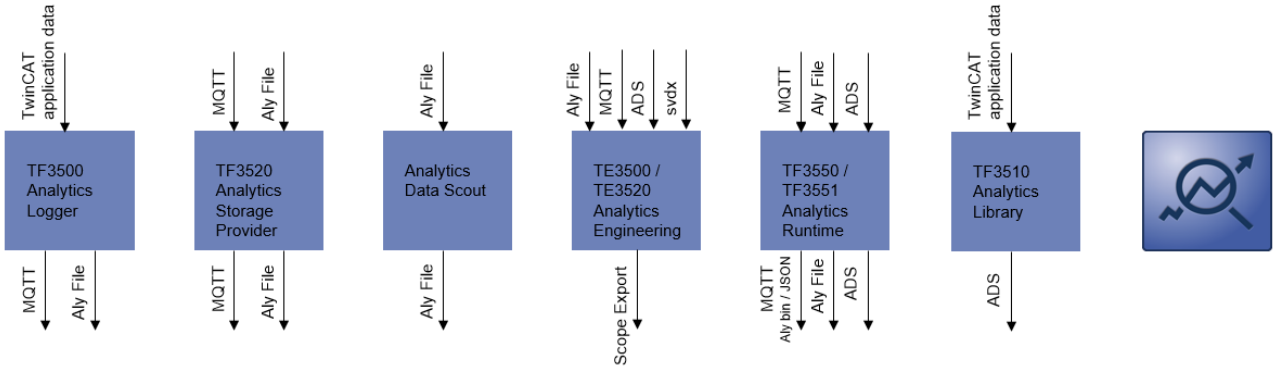
解析設定を作成する場合、以下の画面のようなAnalytics Projectのスタートページから始めます。



このスタートページでは、次に進むにあたり、いくつかのオプションがあります。これらのオプションについては、各オプションごとに以下に説明します。

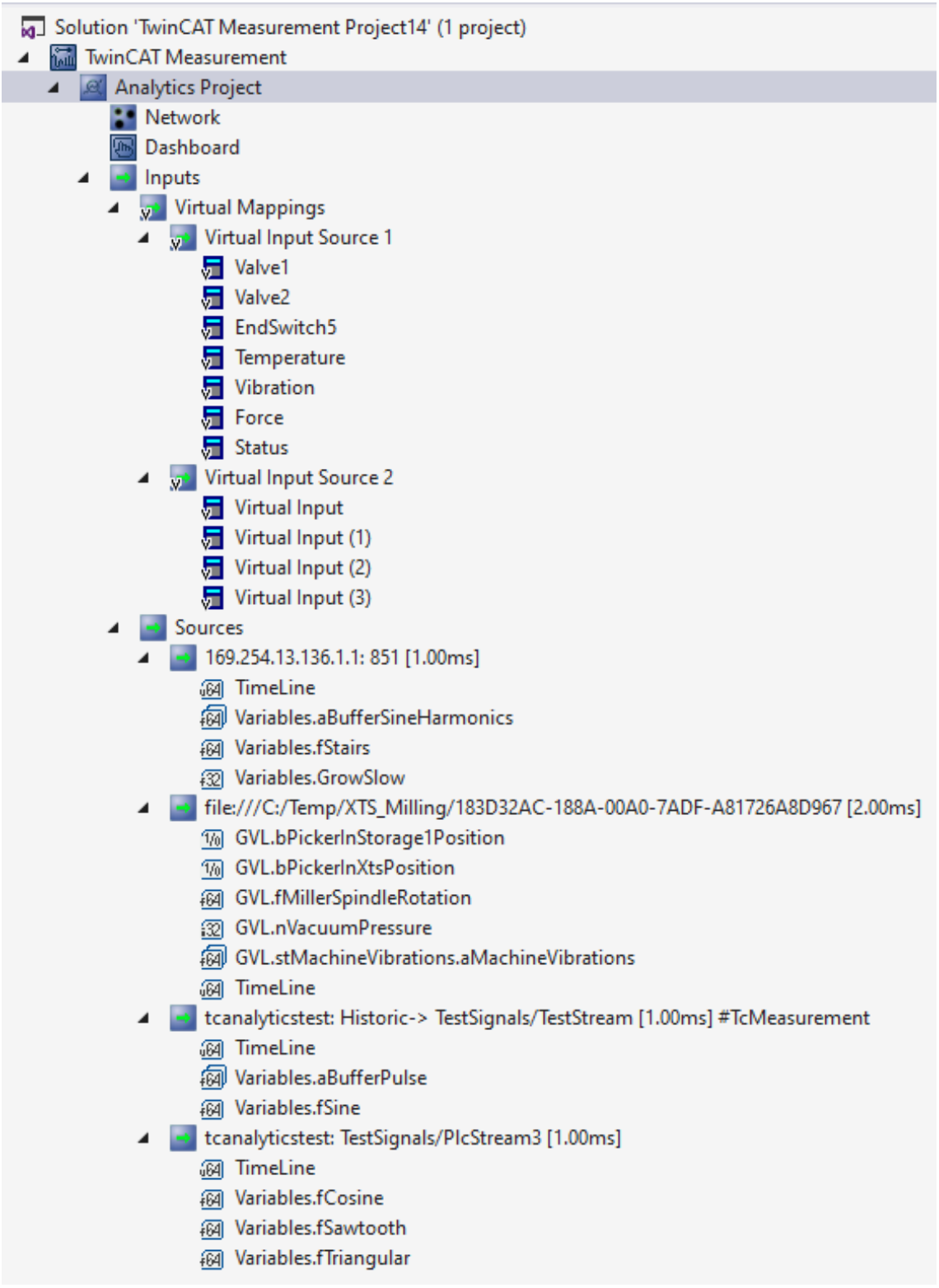
6.1 データソース

TwinCAT Analytics製品には多くのデータソースがあります。同様に、多くの出力形式に対応しており、製品から製品へ、ツールからツールへのシームレスなデータ移行を保証しています。以下の図は、概要を説明するためのものです。



最も汎用的なのは、Analyticsファイル形式およびMQTTストリーム形式のAnalyticsバイナリフォーマットです。

エンジニアリングでは、Solution Explorerの入力エリアで、データソースをVirtual Input SourceとSourceに分けて見ることができます。ソースは、上図に示すように、実際のデータソースを具現化しています。開発製品であるTE3500およびTE3520では、MQTT（ライブストリームおよび履歴ストリーム）、Analyticsファイル、およびADSが使用されます。



Virtual Input Sourceは、実際のソースを抽象化したマッピングです。Virtual Input Sourceは、アルゴリズムの入力とリンクさせるために使用されます。このマッピングによって、実際のデータソースは、アルゴリズムを再リンクすることなく、迅速かつ便利にデータ交換できます。

複数のソースの設定は、Virtual Input Sourcesで作成できます。設定には、ソースの各シンボルと仮想入力のマッピングが含まれます。また、1つのソースに対して異なる設定をすることも可能です。その場合、シンボルのマッピングは仮想入力ごとに異なります。

ソースを追加したり置き換えたりする場合、オプションでソースウィザードを使用できます。

6.1.1 ウィザード

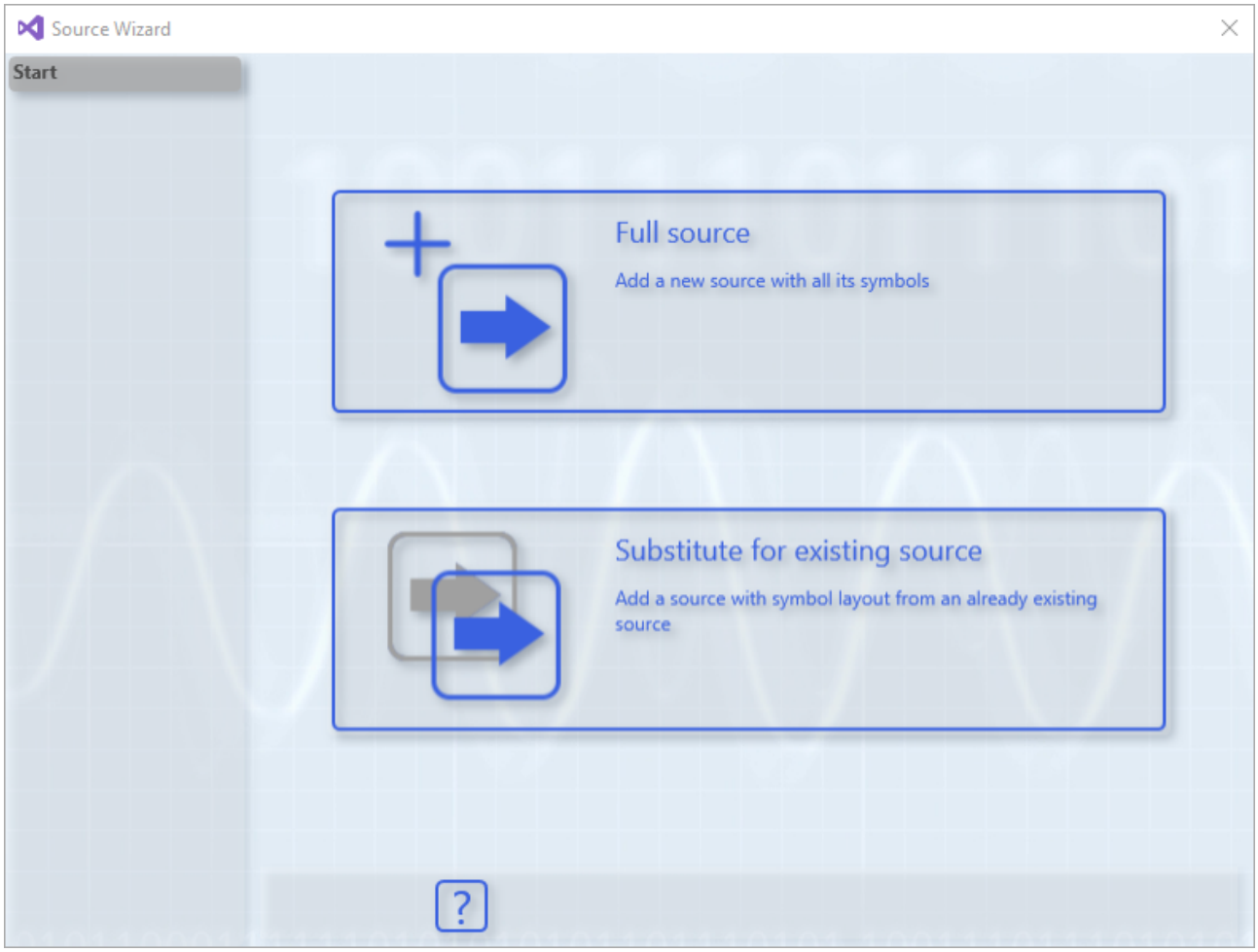
ソースウィザードを使用して、新規データソースを追加できます。このウィンドウでは、データソース全体に追加したり、既存データソースの代替を検索するオプションが提供されます。

ソースウィザードを使用するには、TwinCAT Target Browserをインストールする必要があります。

ページ：

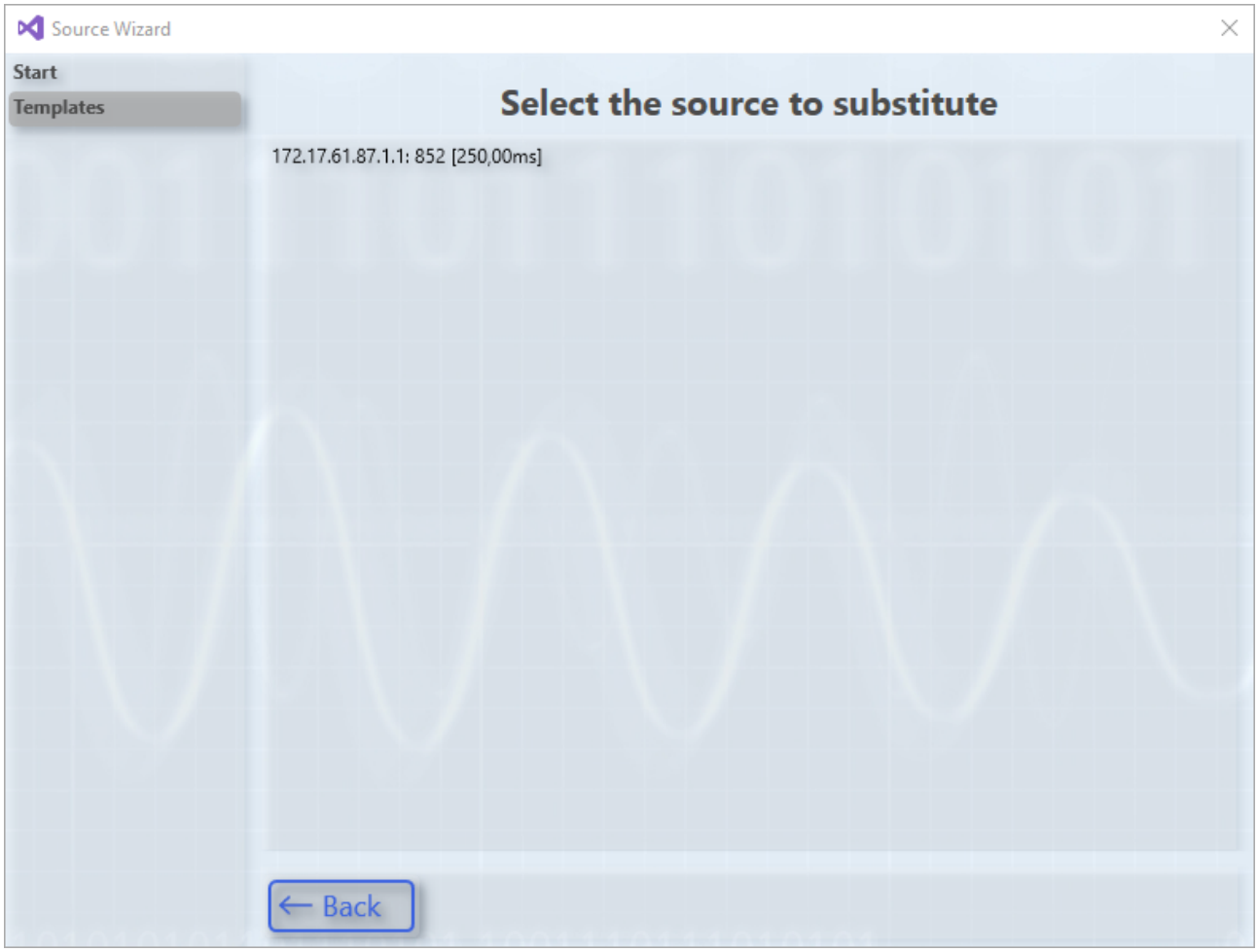
ソースウィザードは複数のページに分かれています。既に表示されているページは左側にリストアップされています。以前に開いたページは、このリストまたは**Back**ボタンおよび**Next**ボタンで再表示できます。

スタートページ



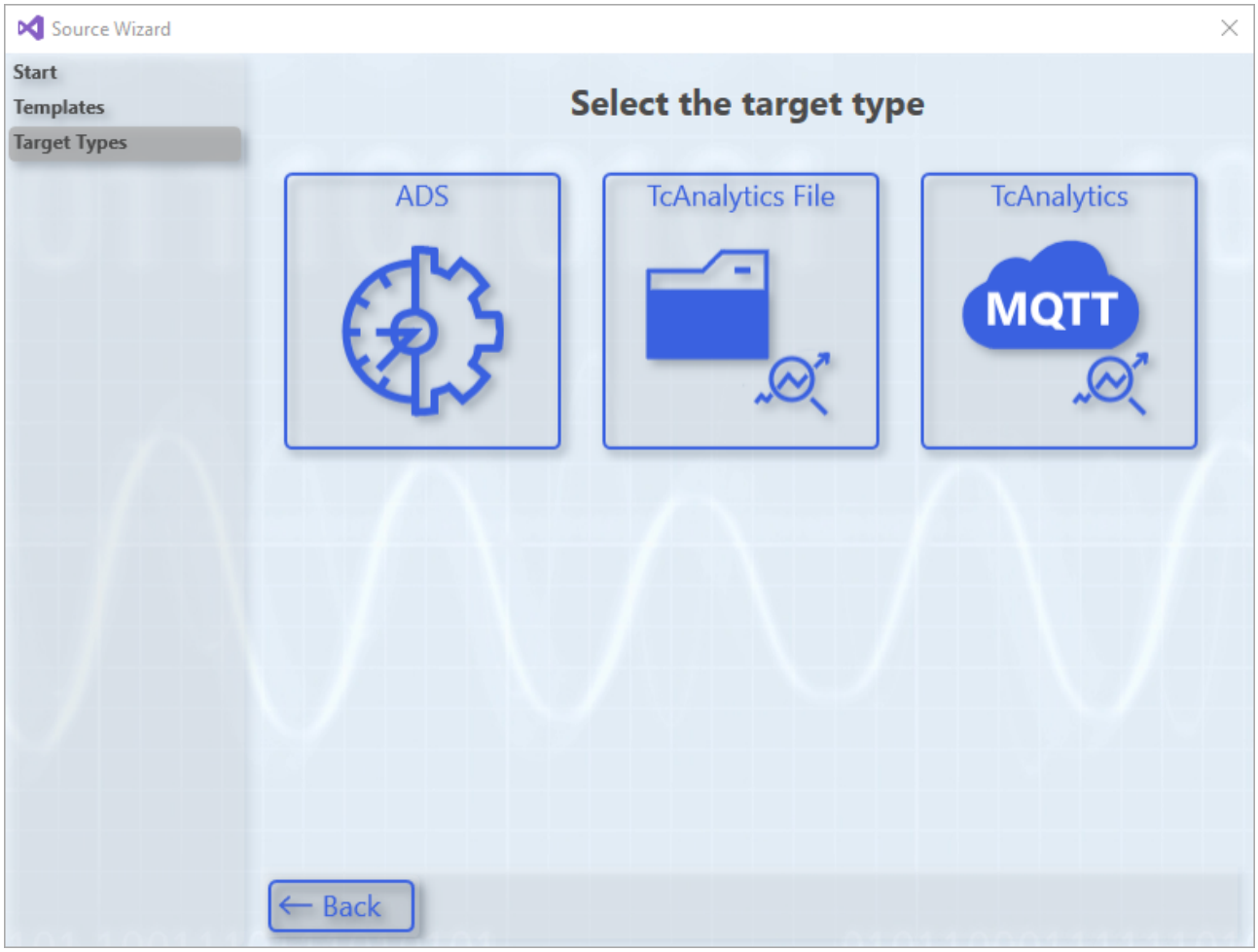
このページでは、ソースの全てのシンボルを追加するか、既存ソースに基づいて求められたシンボルのみを追加するか選択できます。ソースを置き換えるオプションは、プロジェクトに既にソースが存在する場合にのみ選択できます。

テンプレートページ



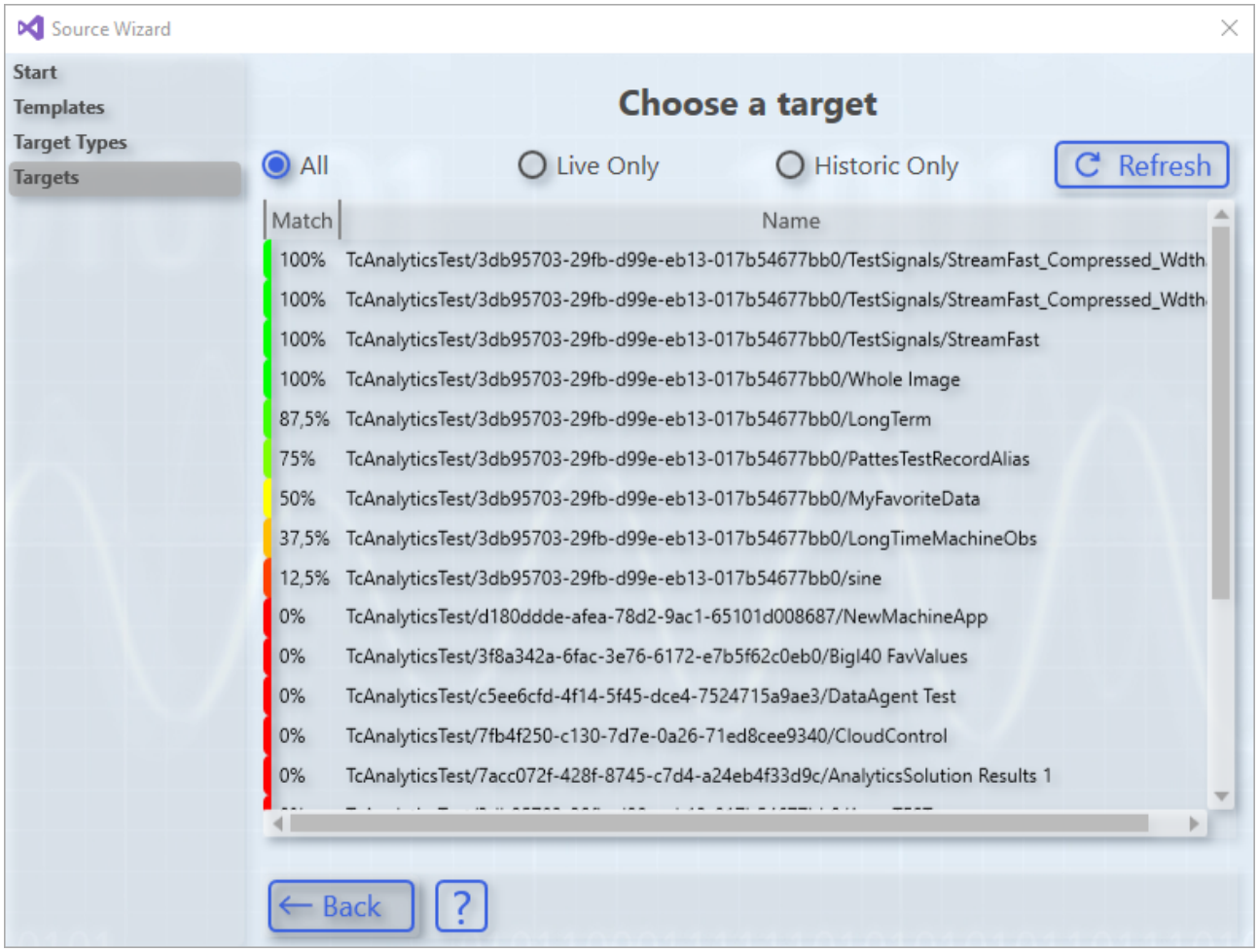
このページはスタートページでReplace（置き換え）オプションが選択されている場合のみ表示されます。プロジェクトの既存ソースはここに表示されます。続行するには、新規ソースのテンプレートになるソースを選択する必要があります。

ソースの種類



このページには、ADSやAnalytics Fileなど、様々なソースタイプが表示されます。タイプの数は、ソースウィザードに対応するTarget Browserにインストールされている拡張機能とは異なります。

ソース



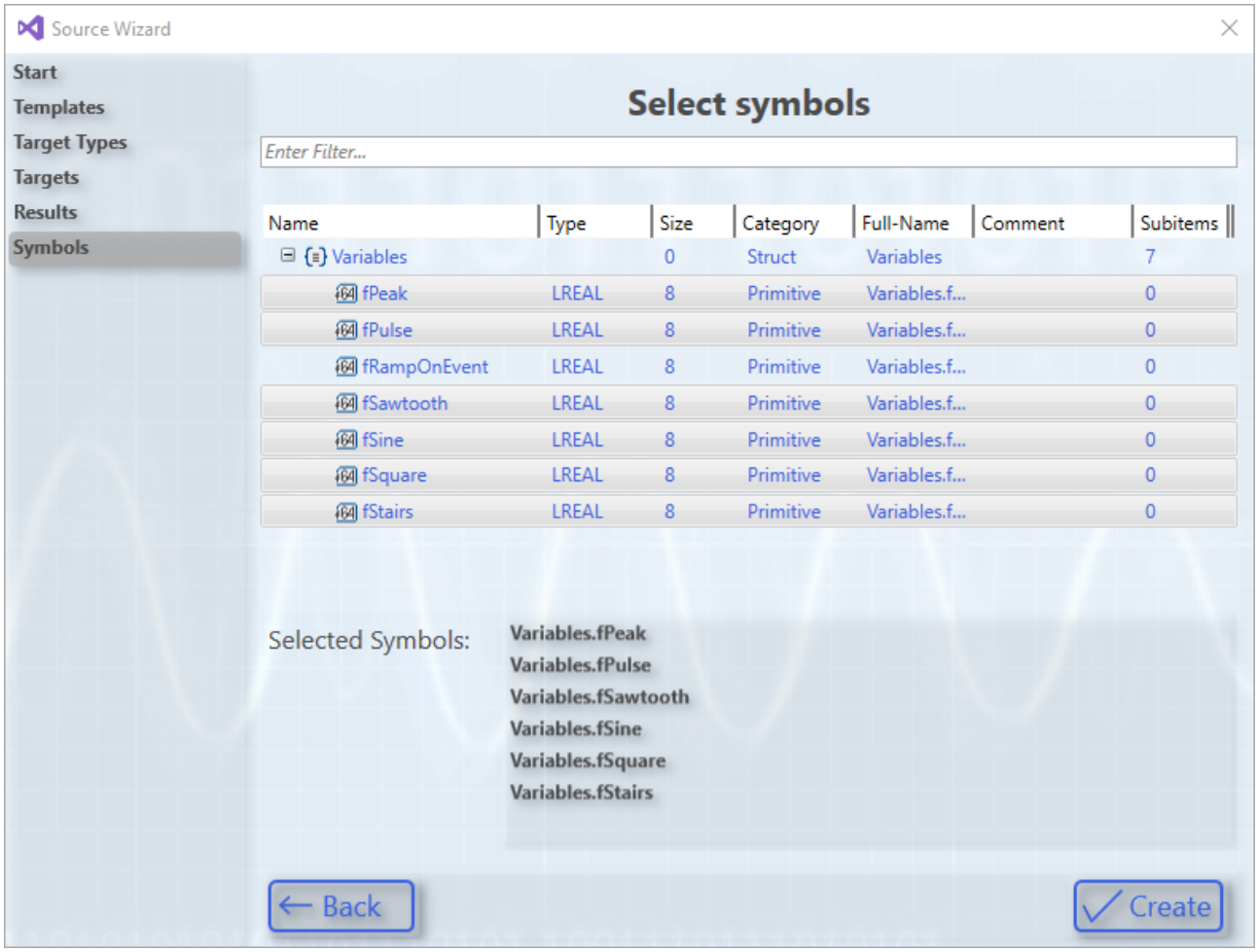
タイプを選択すると、利用可能なソースが表示されます。追加ソースを見つけるには、新規ソースをTarget Browserに保存する必要があります。スタートページでReplace（置き換え）オプションが選択されている場合、ソース名に加えて、テンプレートページで選択したテンプレートとの相関関係が表示されます。相関関係は、テンプレートのシンボルがいくつ見つかったかを示します。さらに、ライブソースのみ、または履歴ソースのみが表示されるようにリストをフィルタリングできます（履歴ソースのみ、またはライブソースのみがある場合を除く）。

結果ページ

Replace（置き換え）オプション使用時の結果ページ

結果ページには、追加する全てのシンボルがリストアップされています。また、見つからなかったテンプレートのアイコンも表示されます（スタートページでソース全体を追加するオプションが選択されている場合、このリストは省略されます）。**Create**をクリックしてウィザードを閉じます。「Matches」リストにある全てのシンボルがプロジェクトに追加されます。選択したシンボルが条件を満たしていない場合は、「Select Symbols（シンボルの選択）」をクリックすると、シンボルのページが開きます。

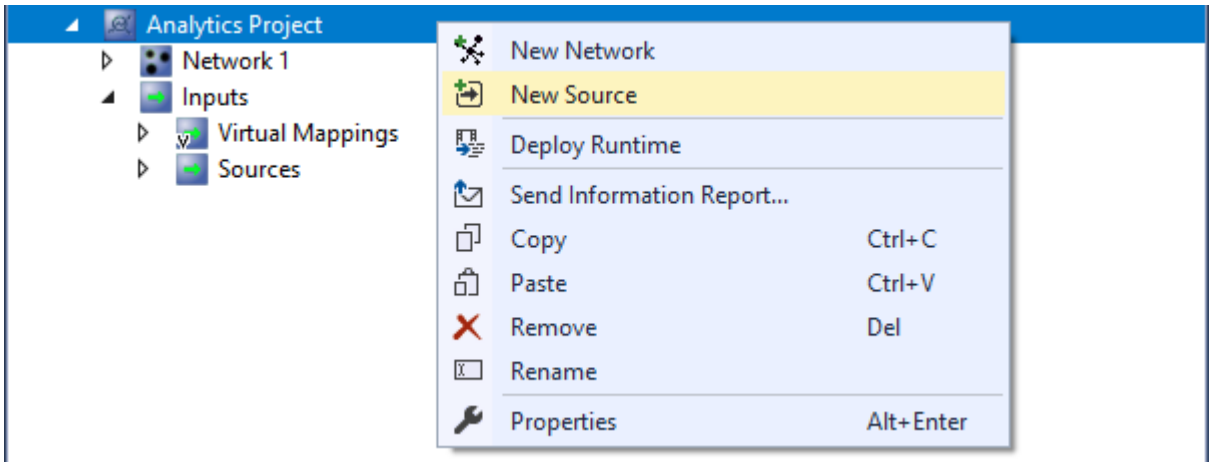
シンボル



このページを開くと、選択したソースの全てのシンボルがリストアップされます。結果ページにもリストアップされているシンボルは既にハイライトされています。さらに、現在ハイライトされているシンボルは全て、ツリーシンボルの下に表示されます。ここで、必要なシンボルを選択したり、不要なシンボルの選択を解除できます。「Create」ボタンをクリックするとウィザードは終了し、選択したシンボルがプロジェクトに追加されます。

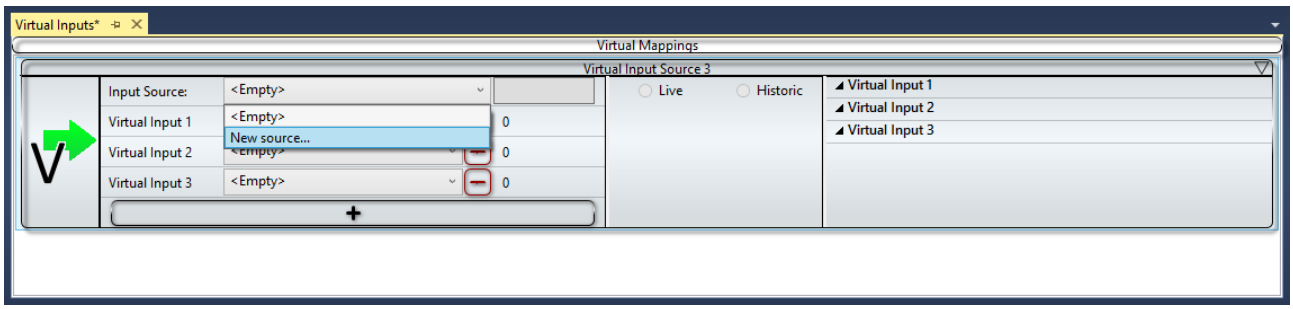
呼び出し:

コンテキストメニュー



Solution Explorer でAnalyticsプロジェクトまたはその下のノードを右クリックすると、コンテキストメニューが開き、**New source**という項目が表示されます。これを選択すると、ウィザードが開始します。正常に完了すると、目的のソースがプロジェクトに追加されます。

仮想ソース



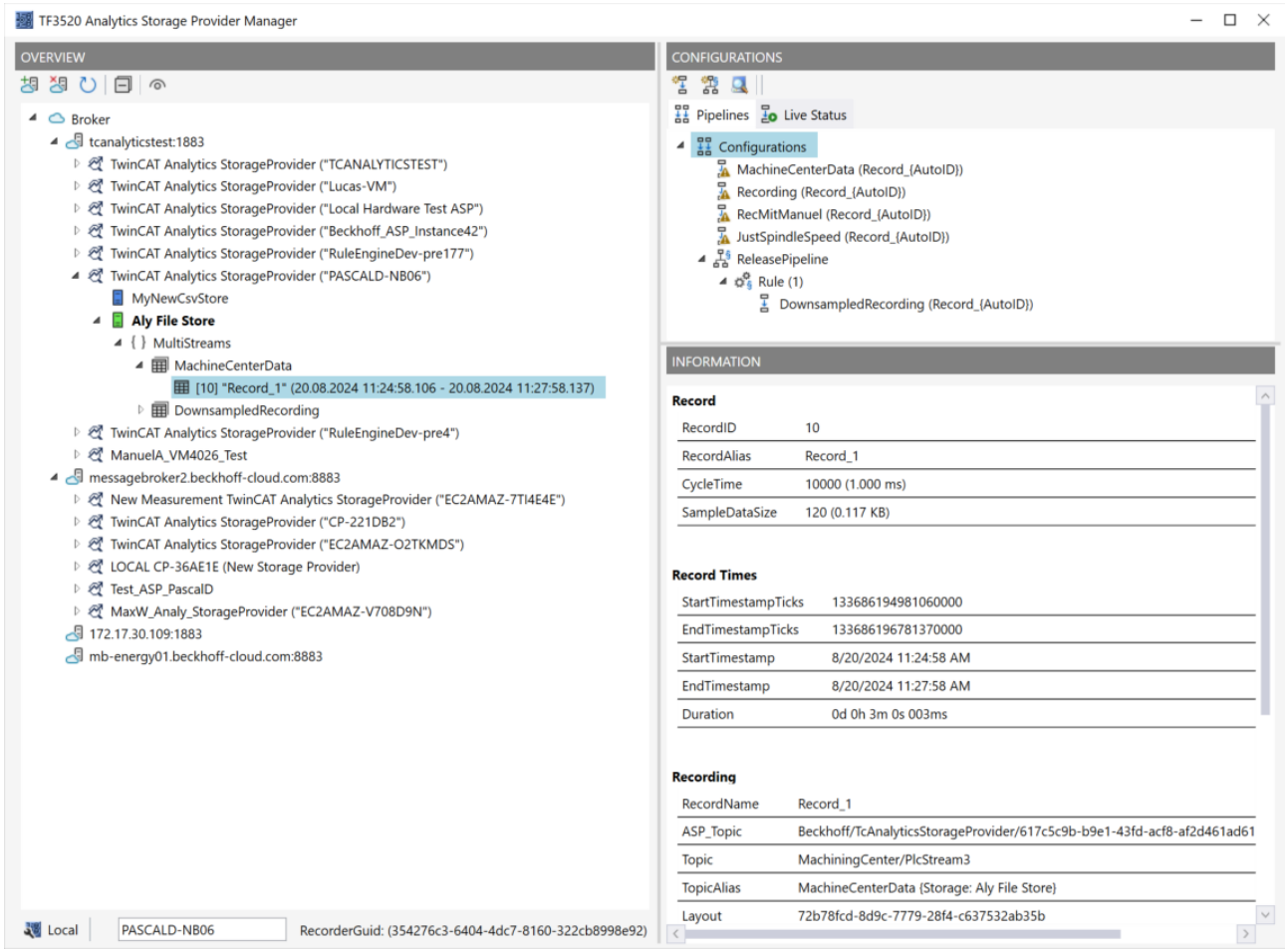
仮想ソースのモジュールでは、**Input Source**のボックスに**New source...**という項目があります。これを選択すると、ウィザードが開始します。この方法でウィザードを開くと、「スタート」ページからではなく、「テンプレート」ページから開始し、現在選択しているテンプレートが仮想ソースの元ソースなります。処理が正常に完了すると、新しいソースが仮想ソースに選択され、選択モードでシンボルを仮想入力に割り当てることができます。

クエリ

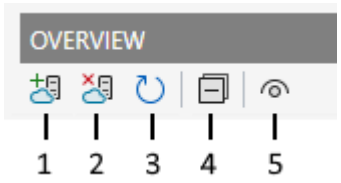
プロジェクト内のどのソースにも属さないシンボルがドラッグ＆ドロップでファンクションに追加された場合、3つのオプションから選択できます。新しい仮想ソースを追加し、そのソースを他のソースと同時に使用するために、既存の仮想ソースを変更するか、ソースウィザードを開きます。スタートページでの選択に応じて、ウィザード完了後に実行されるアクションが異なります。ソース全体が追加されている場合、他のソースと同時に使用できるように、仮想ソースが追加で作成されます。Replace が選択されている場合、新しいソースが仮想ソースに追加され、選択されます。その後、選択モードでシンボルを仮想入力に割り当てることができます。

6.2 Storage Provider Recorder

Storage Provider Recorderでは、記録方法の定義や、記録の開始、管理ができます。さらに、個々のAnalytics Storage Providerのデータメモリを管理することも可能です。見つかったAnalytics Storage Providerと履歴データの全ての重要なプロパティが明確に表示されます。



Toolbar Manager ウィンドウ (OVERVIEW)

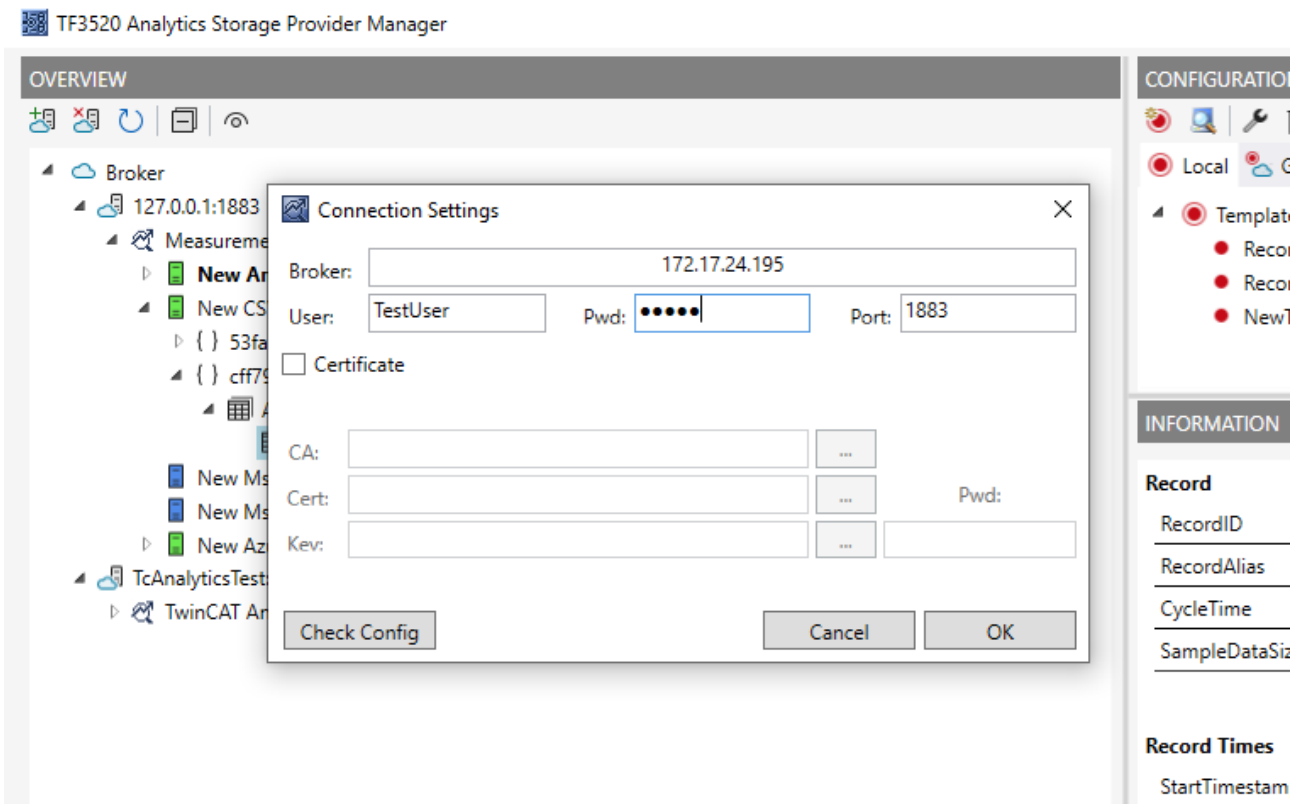


1	新規ブローカーの追加
2	選択したブローカーの削除
3	表示をリフレッシュ
4	全てのノードを折りたたむ
5	ダーク/ライトモードの切り替え

Function Manager ウィンドウ (OVERVIEW)

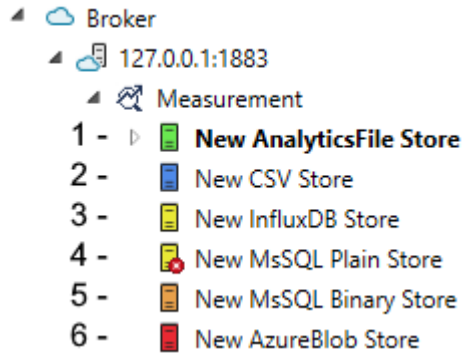
最初に**RecorderAlias**を割り当てます。これは、開始された記録をグループ化し、自動で開始された記録を再び見つけるのに役立ちます。

その後、複数のブローカーを設定できます。これは、既に知られているMQTT接続プロパティの入力マスク経由で行われます。



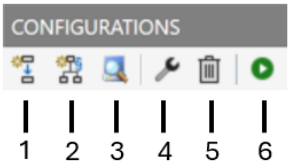
ブローカーへの接続が確立されると、ブローカーに接続されている全てのAnalytics Storage Providersが一覧表示されます。

ストレージのステータス



1	ストレージがオンライン
2	ストレージがオフライン
3	ストレージ起動
4	ストレージ起動がエラー （まだ起動途中）
5	ストレージがシャットダウン
6	ストレージがエラー状態

Toolbar Manager ウィンドウ (CONFIGURATIONS)

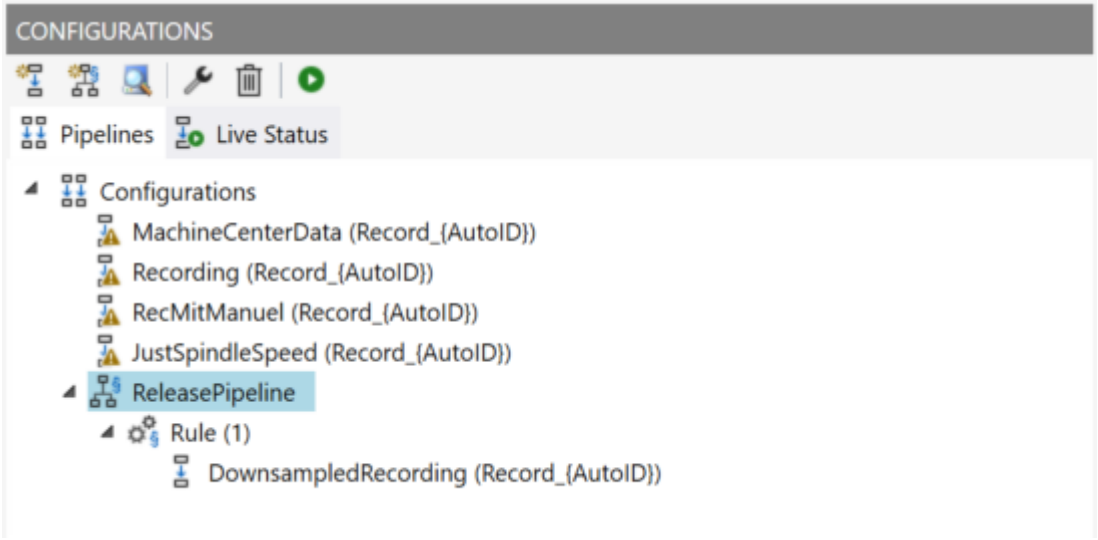


1	新規パイプラインを作成
2	Rule Engineで新規パイプラインを作成

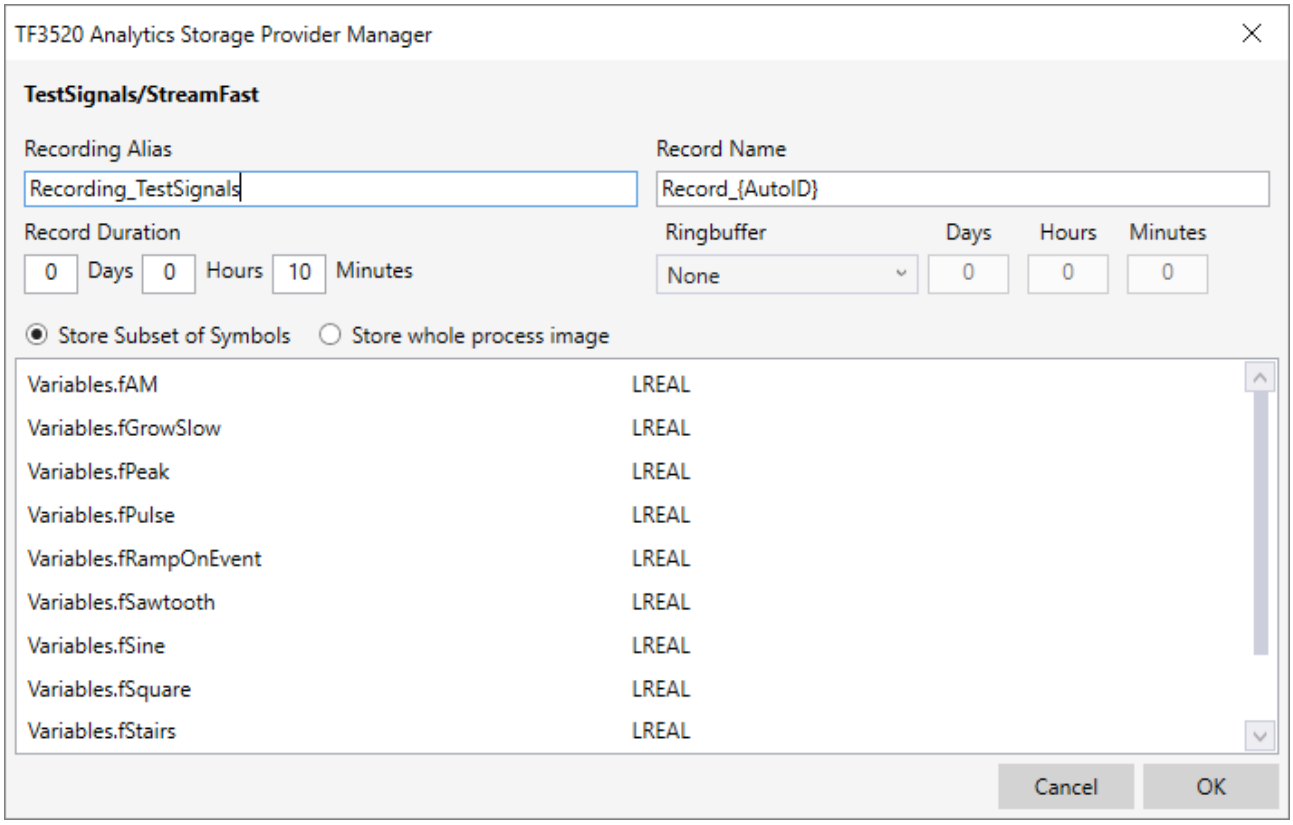
3	シンプルなパイプライン接続のためTarget Browserを開く
4	選択したパイプラインを編集
5	選択したパイプラインを削除
6	選択したパイプラインを開始

Function Manager ウィンドウ (CONFIGURATIONS)

ウィンドウは2つのタブに分かれています。（PipelinesとLive Status）。Pipelinesタブの下でパイプラインの設定ができます。ここで新規パイプラインを定義できます。既存のパイプラインを編集したり 削除、開始できます。



シンプルなパイプラインを新規作成するには、「Create new pipeline」ボタンをクリックします。次のダイアログが開きます。



これで、記録したいシンボルをTarget Browserからダイアログにドラッグ＆ドロップできます。
Recording Alias（記録エイリアス）とRecord Name（記録名）も割り当てます。

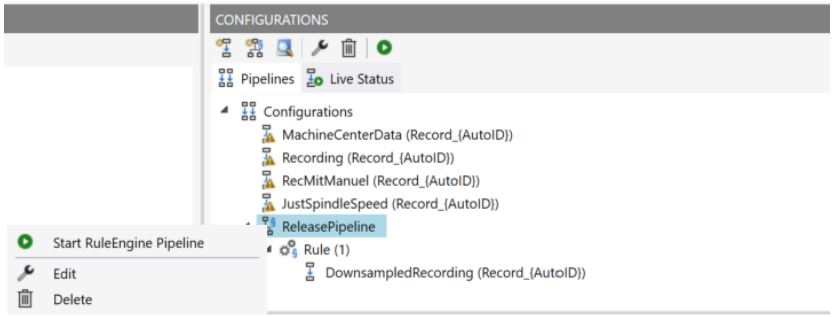
Record Nameには様々なプレースホルダが利用可能です：

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

記録名のほか、記録時間を設定できます（設定しないと、手動で停止するまで記録が継続します）。リングバッファは、ストレージ容量や記録時間に応じて設定します。

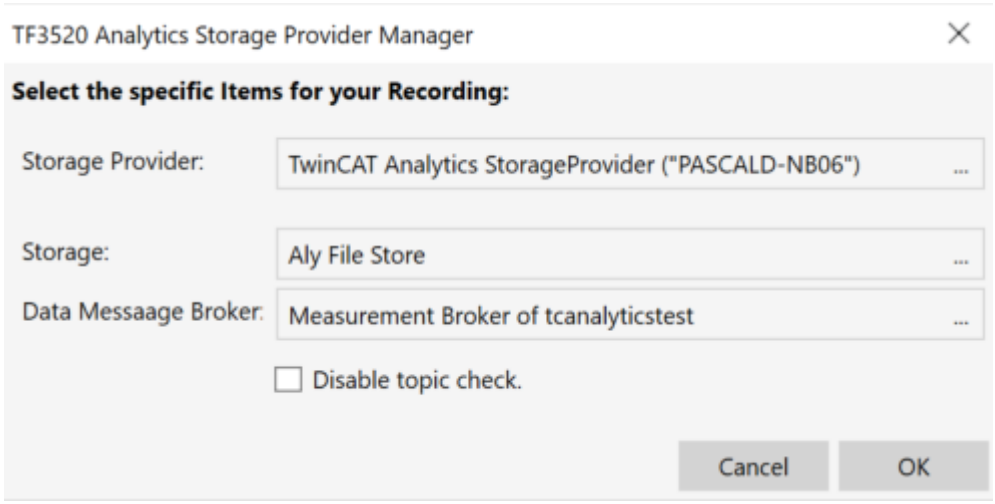
入力内容を**OK**で確定すると、新規のローカル記録設定が作成されます。

この設定は、ツールバーまたはコンテキストメニューから開始できます。



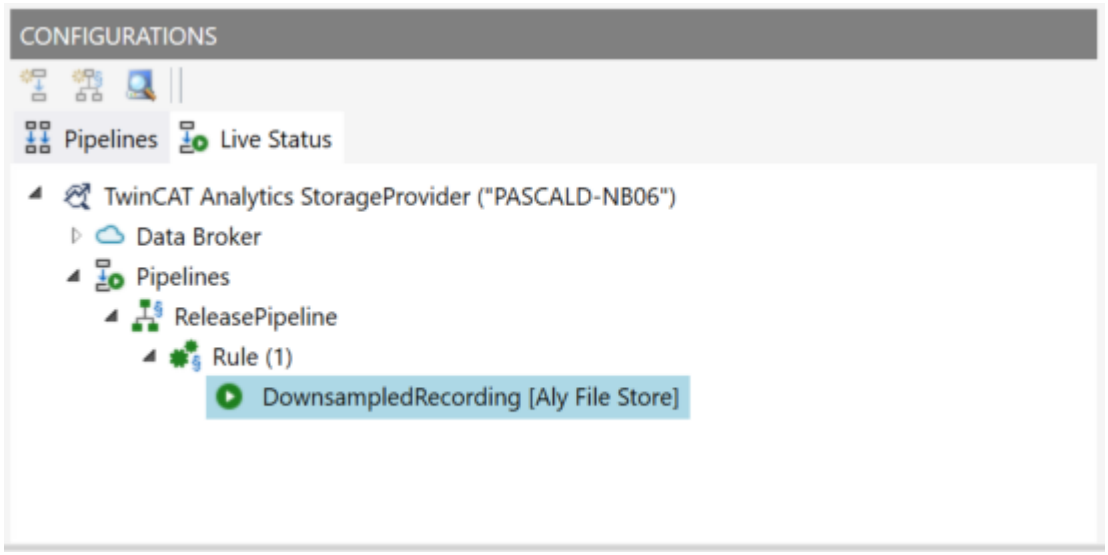
この設定をグローバルアクセス可能にすることもできます。これはコンテキストメニューの**Publish Recording**から設定できます。

次のダイアログが開きます。



ここで、設定をパブリッシュするAnalytics Storage Providerを選択します。さらに、この設定は、選択した Analytics Storage Provider のストレージおよびデータブローカーに割り当てられます。選択後、記録の設定は**OK**で確定され、選択したAnalytics Storage Providerにパブリッシュされます。この設定は、MQTTブローカーに接続されているStorage Provider Managerであればどこからでも見つけることができます。

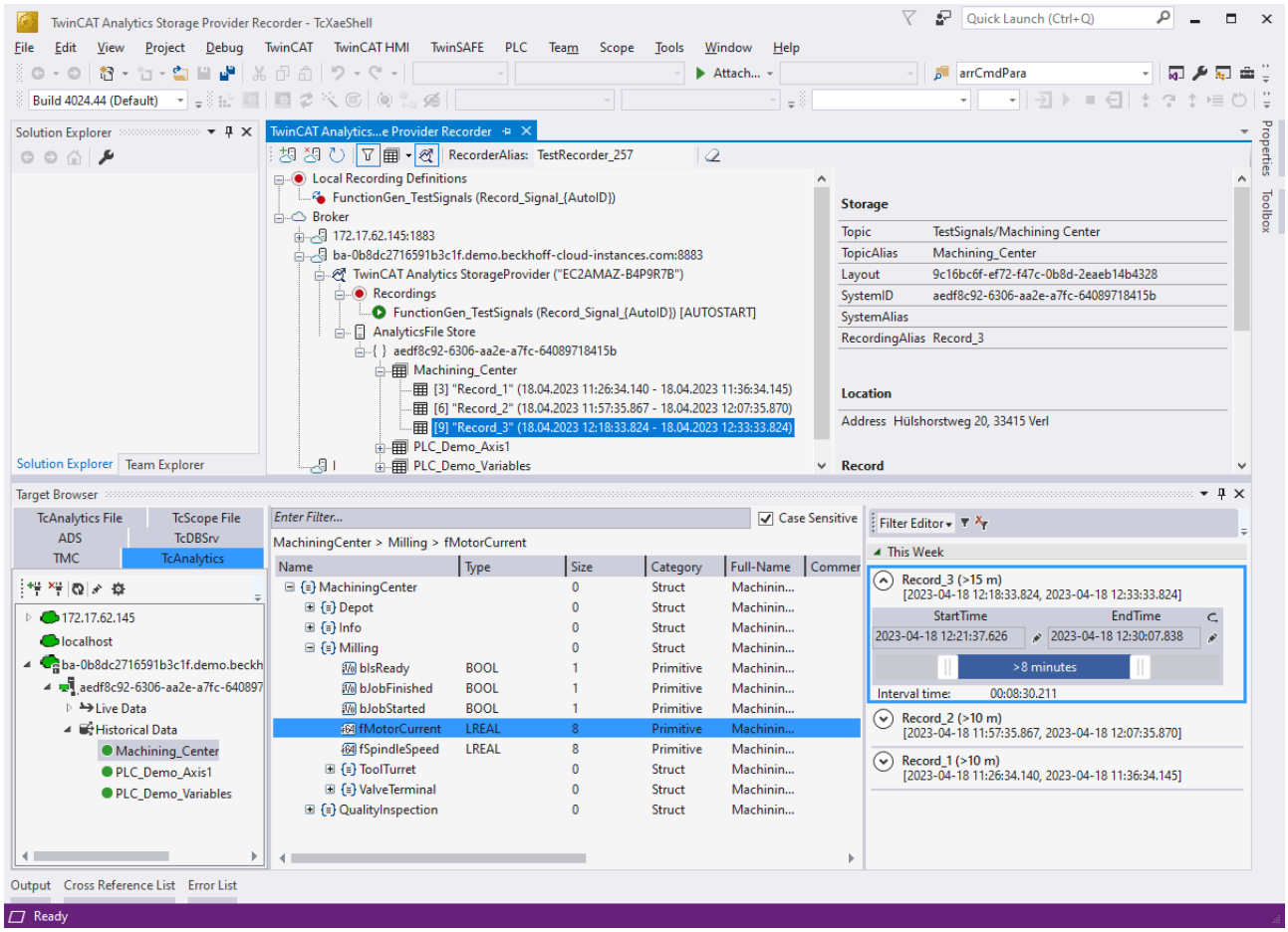
パイプラインを開始すると、表示は自動的に2番目のタブ、Live Statusにジャンプします。



全ユーザの有効な記録がここに一覧表示されます。このタブで記録の終了ができます。また、記録結果へジャンプすることもできます。

履歴データの使用

記録後や記録中に、Target Browserで解析用の入力として履歴データを選択できます。Target Browserの右側に、履歴データ用の新しいコントロールがあります。そこで、データのタイムスパンを選択できます。



6.3 Data Scout

TwinCAT Analytics Data Scoutは、TwinCAT Analyticsの処理フローにおいて非常に重要なツールです。これは、データの閲覧や、データやその記録を操作するために使用します。例えば、複数の記録を1つの記録にマージしたり、不要なデータセクションを削除したりできます。人工的に作成された新しい画像を実際

の解析用のデータソースとして使用できます。Data Scoutは、既に記録されたデータの処理にのみに使用します。入力形式、出力形式は、共にAnalyticsファイルです。このエンジニアリングツールは、TE3500 Analytics WorkbenchとTE3520 Analytics Service Toolで利用できます。

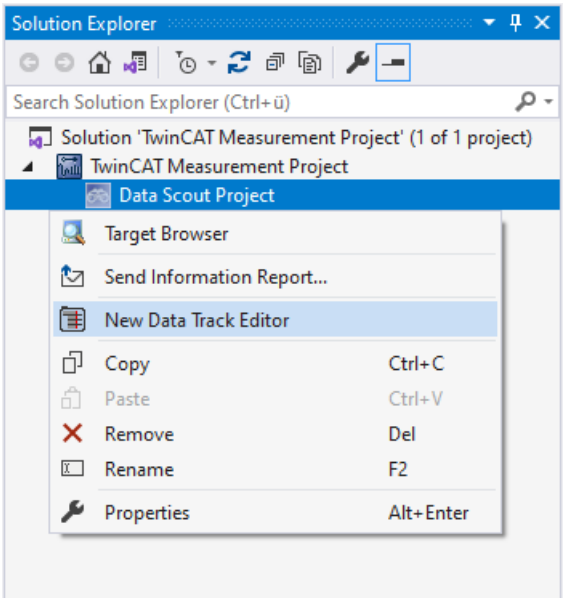
Data Scoutプロジェクトは、計測プロジェクト下のエンジニアリング環境に作成できます。

6.3.1 Data Track Editor

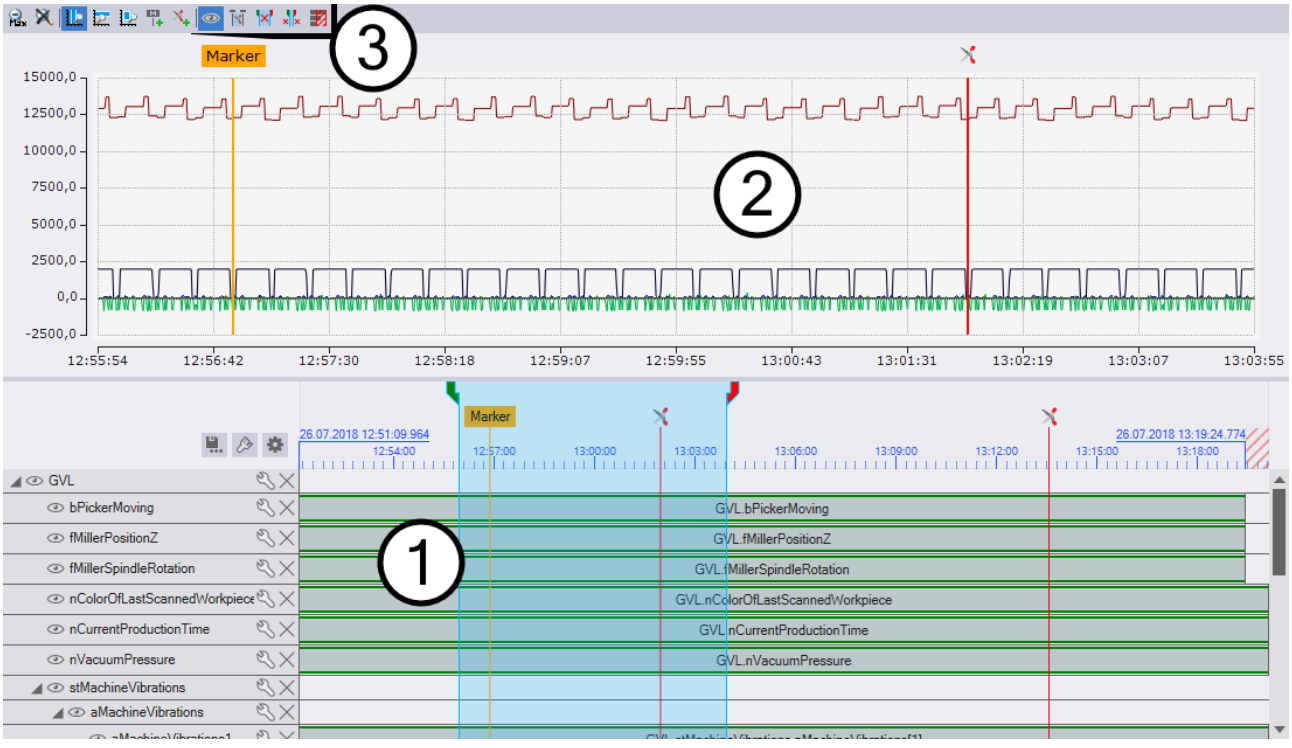
Data Track EditorはAnalytics Data Scout.の一部です。データの表示、複数ソースのデータマージ、データの一部やシンボル全体を切り取る等に使用できます。

新規作成

Data Scoutプロジェクトのコンテキストメニューにある New Data Track Editorをクリックすると、新規のTrack Editorを追加できます。



構造

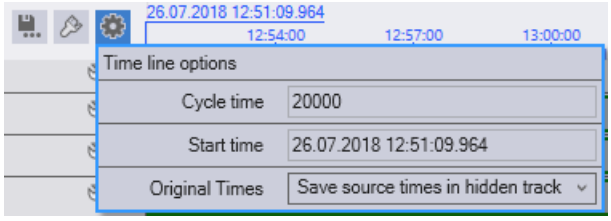


1. Tracks [▶ 61]は、Data Track Editorのシンボルとデータを表します。
2. Chart [▶ 68]は、トラック内のデータをグラフとして可視化するために使用します。
3. toolbar [▶ 68]には、表示エリアを変更したり、削除したりする機能があります。

export [▶ 69]するには、 ボタンをクリックします。

Timeline（タイムライン）

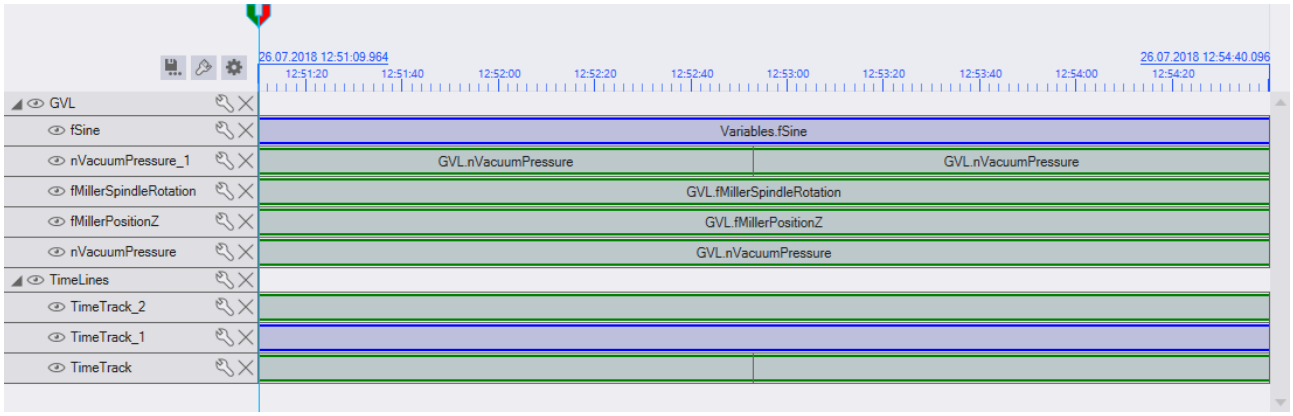
Track Editorで異なるソースを追加すると、新しい時間が割り当てられます。開始時間とサイクルタイムはTimeline Optionsで設定できます。これらの時間はエクスポートとチャートで使用されます。開始時間とサイクルタイムがまだ設定されていない場合、最初に追加したソースの開始時間とサイクルタイムが引き継がれます。



終了時間は最もサンプルの少ないトラックを使用して自動的に計算されます。

Time Track（タイムトラック）

ソースの元の時間を失わないために、タイムトラックを作成できます。これは、Timeline Optionsから設定できます。タイムトラックの生成には3つのバリエーションがあります。タイムトラックを表示トラックとして作成し、全て1つのコンテナにまとめる方法です。2つ目は、同じようにタイムトラックを生成する方法であるが、タイムトラックとそのコンテナがエディタに表示されない方法です。最後の方法は、タイムトラックを作成せず、エクスポート時に元の時間を破棄する方法です。



タイムトラックは、トラックのデータ要素を介して作成します。データ項目の組み合わせごとにタイムトラックがあります。時間範囲とソースのみが考慮され、データ要素のシンボルは考慮されません。

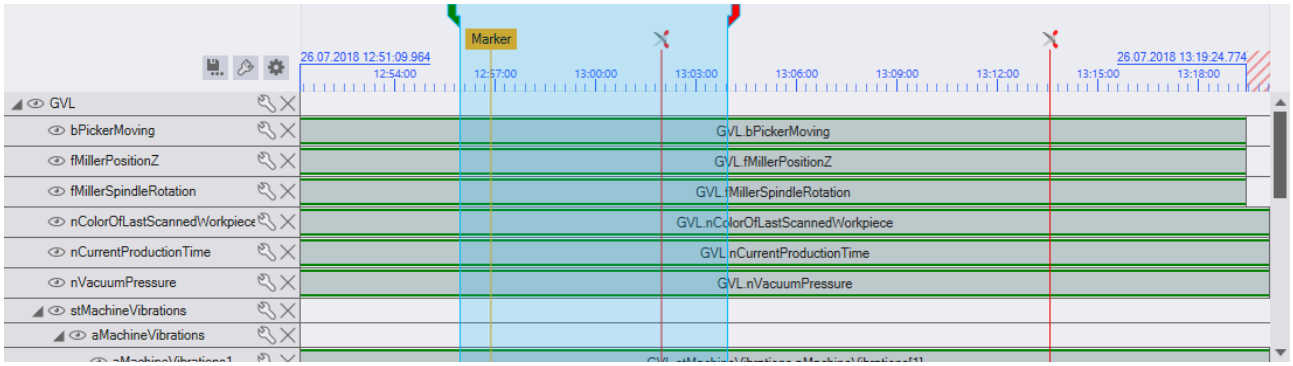
表示の例では、タイムトラックが可視化された形で作成されています。上記のトラックで3つの組み合わせが見つかったため、自動的に3つ作成されました。青いデータ要素が 1 つのトラック、緑のデータ要素が2つのトラック、さらに緑のデータ要素のトラックが3つあります。

タイムトラックはSingle Track（シングルトラック）のような構造になっています。データアイテムを含むことができますが、アイテムにシンボルはなく、時間範囲とソースのみです。

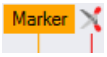
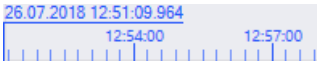
6.3.1.1 Track（トラック）

トラックはData Tracks Editorの要素で、データおよびシンボルの構造に関する情報を含みます。

表示



Data Track Editorのトラックエリアには以下の要素があります：

	このボタンを押すと、新規Track Editor Exportウィンドウが開きます。
	クリックするとポップアップが開き、トラック表示の設定ができます。
	クリックするとポップアップが開き、タイムラインとタイムトラックの設定ができます。
	トラックの上にある青い部分は、どのデータがチャートに表示されるかを示しています。また、選択したエリアは、ツールバーの機能に使用されます。赤と緑のエリアを動かすことで、選択範囲を広げたり、狭めたりできます。赤と緑のエリアの間をクリックすると、選択した部分を移動できます。 左右の端では、選択範囲をスナップできます。これは、マーカやトラックエンド付近で起こります。スナップ機能を無視するには、移動中にAltキーを押します。 選択値は10を下回ることはできません。
	マーカはチャートだけでなく、トラックにも表示されます。トラックでは、テキストをクリックしてマーカの名前を変更できます。また、右クリックでマーカを削除できます。文字がなくハサミがあるマーカは、消去に使える特別なマーカです。
	タイムラインでは、左端に開始時間が記されます。その後、マーカが等間隔で続き、経過する時間を示します。
	タイムラインの右端には終了時間のマーカがあります。終了時間は、開始時間に最もサンプル数の少ないトラックのサンプル数にサイクルタイムを掛けた値を加えたものと等しくなります。 トラックのサンプル数が異なる場合、終了時間マーカの後ろに赤い破線エリアが表示されます。全てのトラックデータがないため、このエリアはエクスポートされません。
Tracks Container（トラックコンテナ）	下部にはトラックが表示されています。トラックのデータ表示は開始時間から始まり、その前には新しいシンボルが表示されます。

Data Tracks（データトラック）

現在、データを格納できるトラックは2種類あります。1つ目はSingle Track（シングルトラック）です。これはシンボルが1つであることを示しています。2つ目はMulti Track（マルチトラック）です。これは、複数のシンボルをグループ化し、シンプルな表示を維持できます。ただし、マルチトラックでは、データ型や含まれるシンボルの名前をカスタマイズできません。

データを含むトラックも同様に表示されます。

左側には以下の機能があります：

	ここでは、トラックをグラフとしてチャートに表示するかどうか設定できます。
トラック名	トラックの名前は、グラフではチャートに、エクスポートではシンボル名として使用されます。名前はコンテナ内で一意にする必要があり、スペースを含むことはできません。
	クリックするとポップアップが開きます。ここにトラックの設定が表示されます（詳しくは設定の項を参照）。
	このボタンは、トラックとその下の全てのトラックを削除します。

右側にはトラックのデータ要素が表示されます。データトラックは、トラックのタイプに合った複数のデータ要素を含むことができます（シングルトラックは配列を含むことができません）。

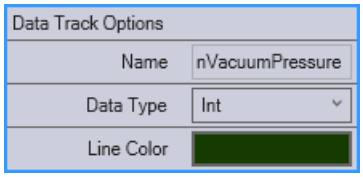
コンテナトラック

コンテナトラックはデータトラックと同様の構造になっています。ただし、コンテナトラックにデータ要素を入れることはできませんが、トラックを入れることはできます。格納されたトラックはコンテナトラックの下に表示され、 ボタンで展開または折りたたむことができます。コンテナトラックを表示または非表示にすると、全てのサブトラックも表示または非表示になります。

設定

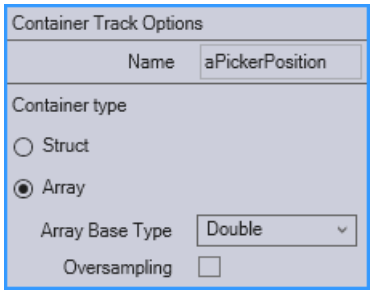
全タイプのトラックについて、 ボタンのポップアップから異なる設定ができます。

シングルトラックの場合、以下のようなものがあります：



データ型	ここでシングルトラックのデータ型を定義します。表示データとエクスポートデータは、ここで指定した型に変換されます。 文字列データ型の場合は、文字列の長さを設定するため、別のテキストフィールドが表示されます。 トラックの1つまたは複数のデータ要素がより大きなデータ型を持っており、変換の問題が発生する可能性がある場合は、警告が通知されますが、値は変換されます。
線の色	グラフの色を設定します。クリックするとカラーピッカーが表示され、色を設定できます。

コンテナトラックには以下の設定があります：

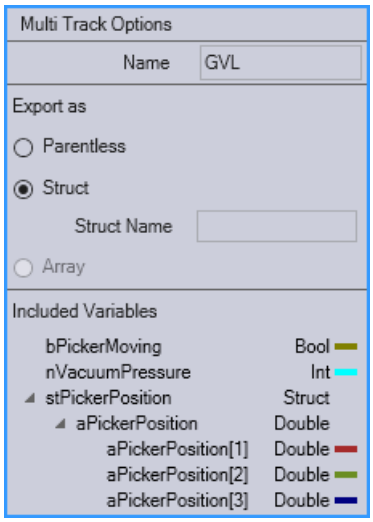


コンテナのタイプ	コンテナトラックは構造体も配列も可能です。ただし、配列として作成するためには、全てのサブトラックが配列に対して互換性を持っている必要があります。シングルトラックしかない場合、コンテナトラックは基本的なデータ型の配列にできます。同じ構造を持つコンテナトラックしかない場合、基本的な配列、あるいは構造体を基本データ型とする配列にできます。

各コンテナタイプはそれぞれ独自のプロパティを持っています：

構造	構造名	エクスポートするデータ型の名前を指定します。
配列	基本のデータ型	ここでは、配列要素のデータ型を設定できます。ここでデータ型を設定すると、サブコンテナトラックの配列を基本としたデータ型と、サブシングルトラックのデータ型が変更されます。 文字列データ型の場合、文字列の長さを設定できます。
	オーバーサンプリング	このオプションは、コンテナトラックがサブトラックとして、シングルトラックのみ持っている場合に有効です。有効時には、サブトラックは個別のグラフとしてチャートに表示されず、コンテナトラックが1つのグラフとして表示されます。このため、サブトラックの値がロードされ、同時に全ての値が順に配置されます。

マルチトラックには以下の設定があります：



エクスポート設定	コンテナトラックのコンテナタイプと同様に、マルチトラックのエクスポート方法を設定できます。 Structure（構造体）の場合、マルチトラックは含まれる変数を要素とする構造体データ型としてエクスポートされます。希望する構造体の名前を指定できます。 配列としてエクスポートするオプションは、含まれる全ての変数のデータ型が同じ場合にのみ使用できます。データ型も変更はできません。マルチトラックは、エクスポート時に配列となり、含まれる変数が配列要素となります。 Parentless（親なし）の場合、マルチトラックはエクスポートに対して考慮されず、含まれる変数は、マルチトラックの階層にあるかのようになります。
含まれる変数	ここにはマルチトラックに存在するシンボルが表示されます。マルチトラックの全てのデータ要素は同じシンボルである必要があります。

データ要素

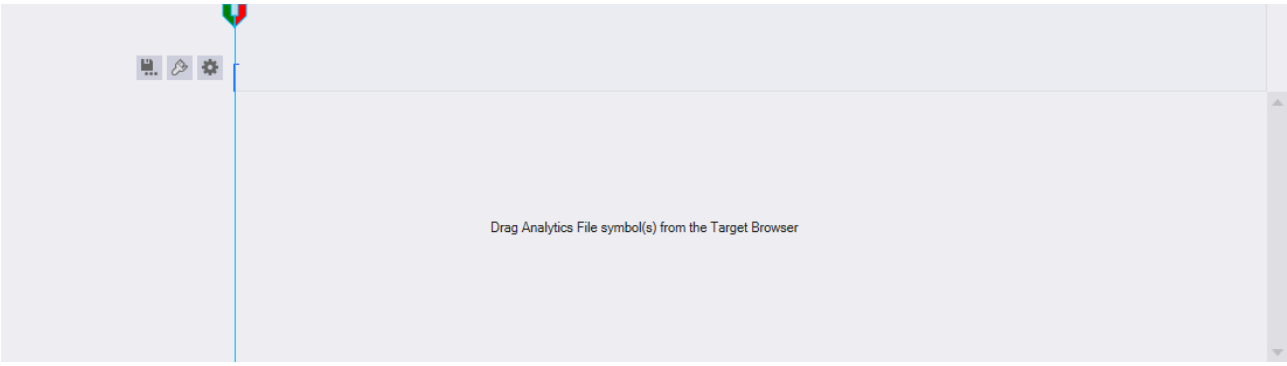
データ要素はトラックに配置できます。これらの変数には、ソースから1つ以上の変数と、時間範囲が含まれます。時間範囲とソースのサイクルタイムを使用して、データ要素内のサンプル数が計算されます。データアイテムの長さは、どのトラックが最も多くのサンプルを含み、利用可能な合計スペースによって決定されます。長さは、サンプル数と、1つのサンプルに対応するピクセル数に依存します。

データアイテムはブロックとして表示されます。データアイテムの色はソースによって異なるため、同じソースのデータアイテムは全て同じ色になります。データ要素の最後には区切り線があります。ソースからのシンボルの元の名前がデータ要素に表示されます。記号が複数ある場合は、記号の数のみが表示されます。

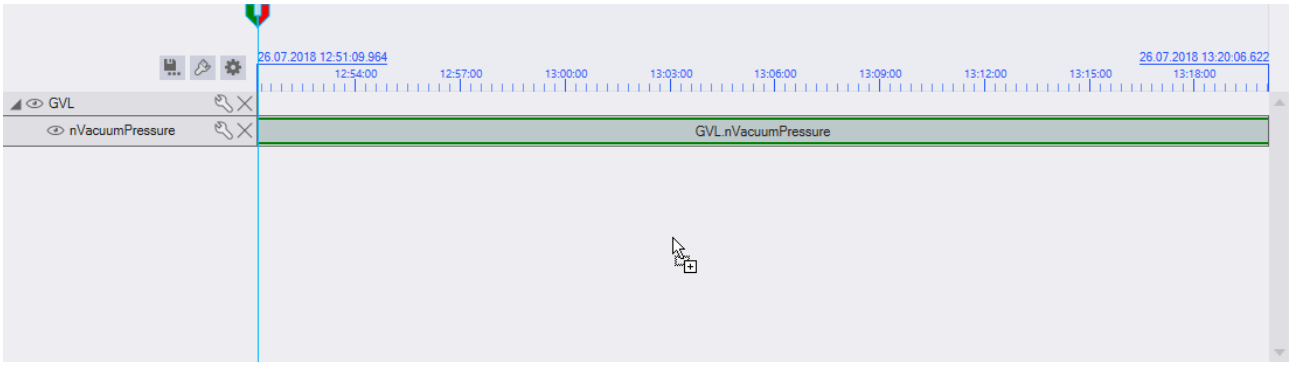
データ要素を右クリックすると、コンテキストメニューが開きます。ツールバーの機能を使わずに、データ要素を完全に削除することもできます。

ドラッグ&ドロップで追加

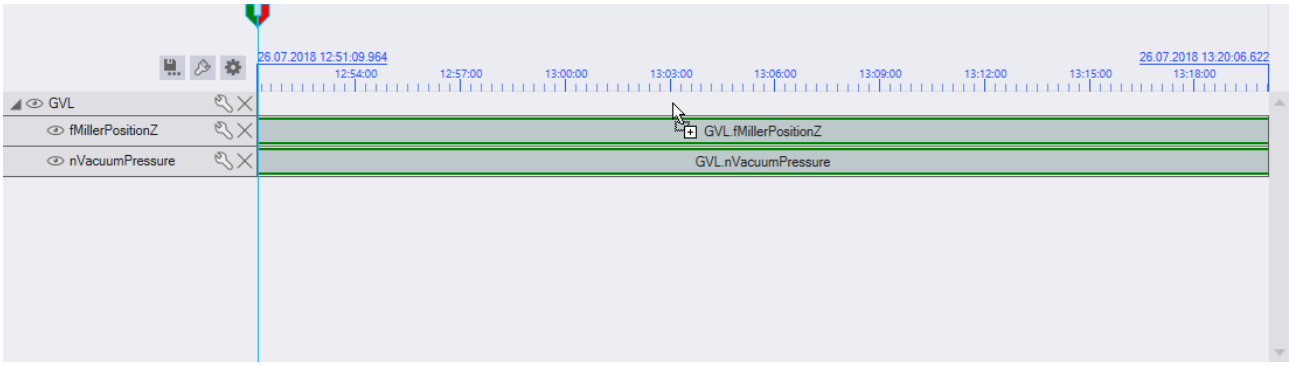
Data Track Editorのトラックは別のエリアに表示されます。これは最初は空ですが、Target Browserを使って入力できます。



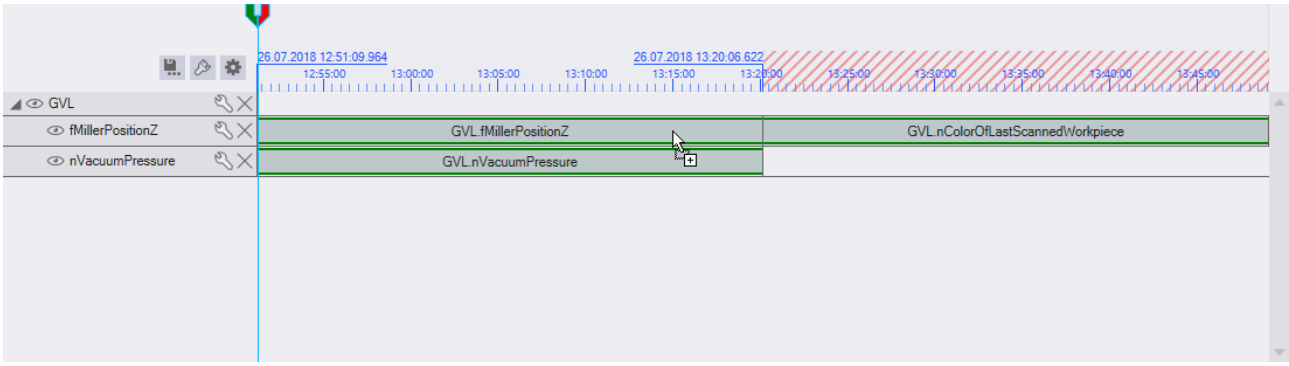
Target Browserから1つ以上のシンボルを選択し、Track Editorの空エリアにドラッグすると、新しいコンテナトラックが作成されます。新しいコンテナトラックに、選択した全てのシンボルがトラックとして追加されます。



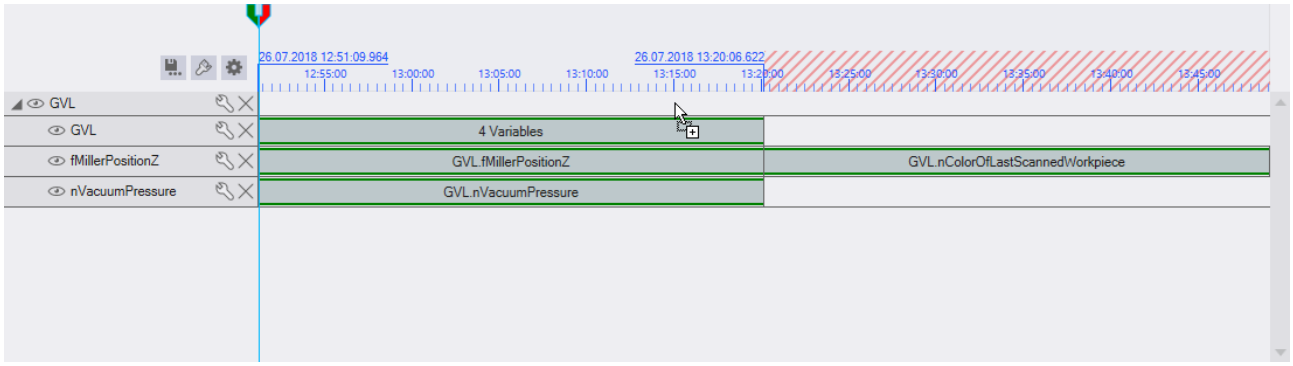
コンテナトラックのデータエリアにシンボルがドラッグされると、コンテナの下に新しいトラックが作成されます。



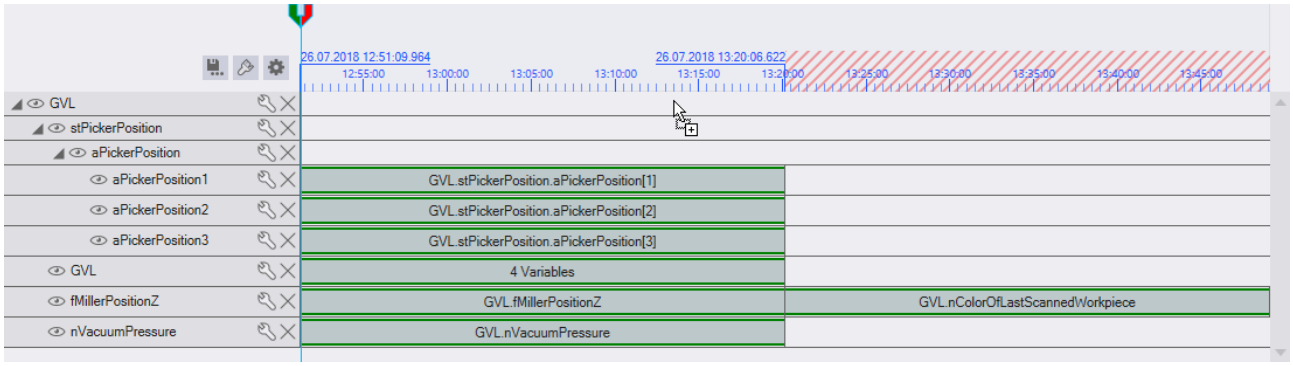
データトラックにドラッグすると、新しいデータ要素を追加できます。ただし、これは追加するシンボルがトラックと互換性がある場合にのみ可能です。基本的なデータ型を持つトラックには、構造体を記述できません。



新しいマルチトラックを作成するには、[Ctrl] キーを押しながらコンテナトラックにドラッグするか、空のエリアにドラッグします。

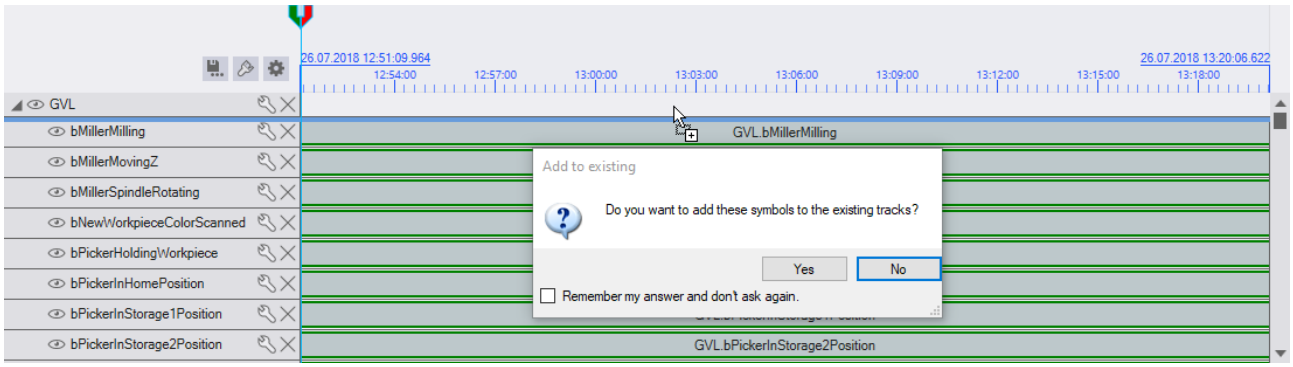


構造体や配列が追加される場合は、基本的なデータ型を持つシンボルと同様のことが発生する可能性があります。ただし、コンテナトラックは、構造体や配列の上位要素として作成されます。

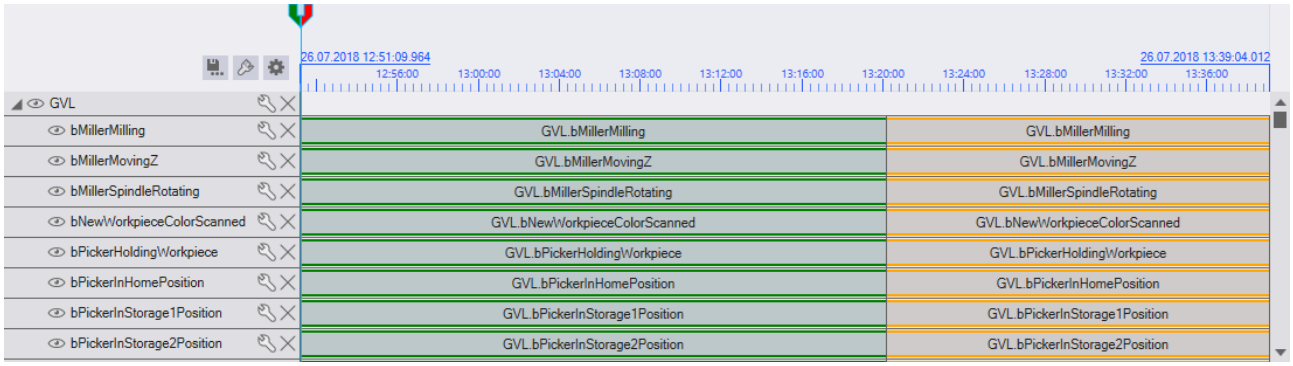


自動マージ

複数の記録を持つAnalyticsファイルなど、同じ構造を持つシンボルをより簡単にマージするため、複数のシンボルをコンテナやトラックの空きエリアにドラッグすると、シンボルが同じかどうかシステムがチェックします。検出後すぐに、すでに存在するトラックにシンボルを追加できる旨のメッセージが表示されます。



このクエリにYesと答えた場合、追加したシンボルはデータ要素として、一致する既存トラックに追加されます。Noと答えた場合、新しいトラックが追加されます。



各シンボルを個別に既存のトラックにドラッグしても、同じ結果が得られます。

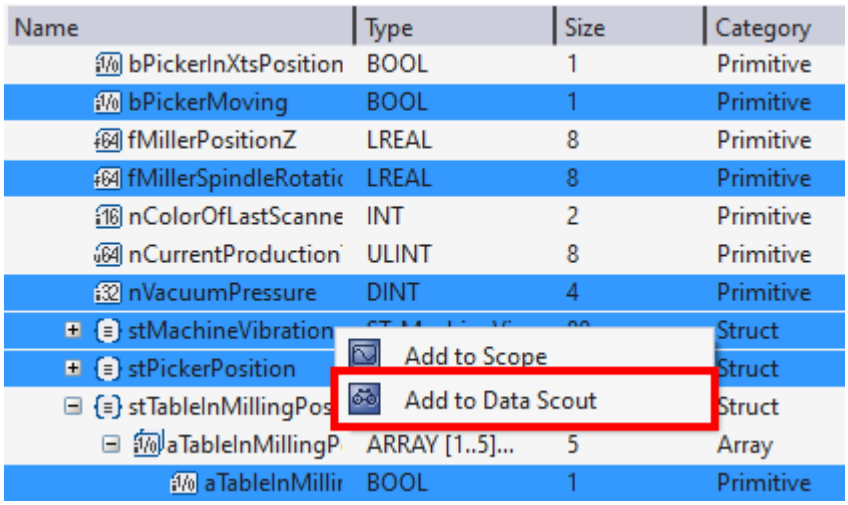
順序の変更

既存のトラックやデータ要素の順序を変更する場合は、ドラッグ＆ドロップで行うことができます。データ要素は全面で移動でき、左の境界線 のトラックも移動できます。これを行うには、対象のエリアをマウスで左クリックし、マウスを押したまま移動させる必要があります。

データ要素は同じトラック内で並べ替えたり、別のトラックに移動させることができます。データ要素をコンテナトラックまたはトラックエディタにドラッグすると、新しいトラックが作成されます。また、データ要素は移動先のトラックと互換性が必要です。例えば、マルチトラックから複数のシンボルを持つデータ要素を、シングルトラックに移動することはできません。

トラックは、そのトラックが入っているコンテナの下で並べ替えたり、別のコンテナに移動できます。移動しようとしているトラックの内部にあるコンテナにそのトラックを移動することはできません。また、トラック自体の設定や、上位のコンテナの設定によって変更が許可されていないトラック（例：構造対配列など）を移動することもできません

Target Browserコマンドによる追加

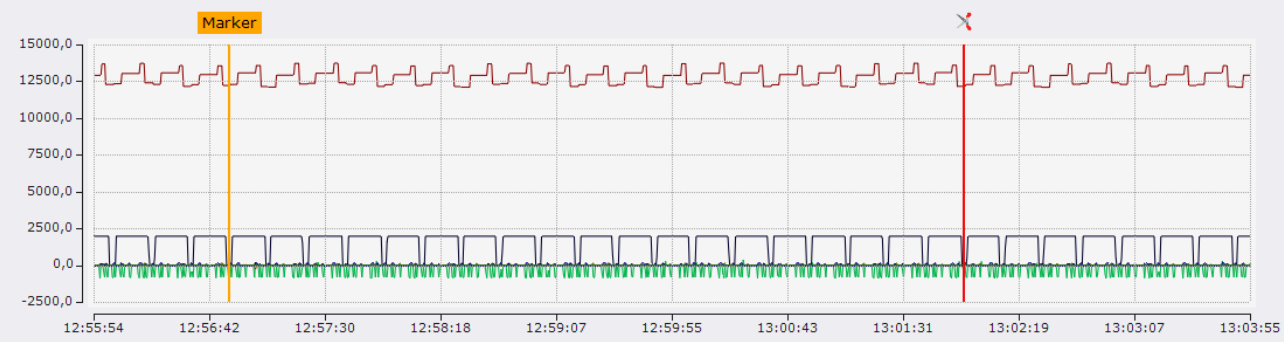


コンテキストメニューの 「Add to Data Scout」コマンドを呼び出すことで、Target BrowserからTrack Editorーにシンボルを追加できます。このコマンドは、Track Editorに対応したシンボルをTarget Browserで選択し、右クリックすると表示されます。次に各シンボルと、配列や構造体のサブシンボルに対してトラックが作成されます。

Data Scoutプロジェクトが存在しない場合、新規プロジェクト作成が試みられます。既にData Scoutプロジェクトがある場合、シンボルはそのプロジェクトに追加されます。Solution ExplorerでTrack Editorが選択されていれば、そこにシンボルが挿入され、そうでなければ新規作成されます。

6.3.1.2 チャート

Data Track Editorは、スコープビューのYTチャートに類似したチャートを使用して、トラックのデータを表示します。



チャートは左側にY軸、下側にX軸があり、その上に時間が表示されます。データを含む表示トラックは、これらの軸上で、それぞれグラフとして表示されます。

チャート用に時間値が再割り当てされるため、元のソースからの時間値は、1つのチャートで複数のソースを表現する際には使用されません。開始時間とサイクルタイムは、Track Editorのタイムラインオプションで設定できます。

Track Editorでは、タイムラインとトラックのデータ要素の上に青いエリアが表示されます。このエリアは、どのエリアがチャートに表示されているかを示しています。ツールバーの機能またはマウスホイールを使ってチャートを拡大すると、トラック内の選択エリアも変化します。その逆も同様で、トラック内で選択範囲を調整すると、チャートに表示される範囲を変更できます。

データのロード

Data Track Editorにデータのあるトラックが入るとすぐ、このトラックのデータがロードされます。ロードはバックグラウンドで自動的に行われます。ロードの際、データ要素の構成が同じ、つまり同じタイムトラックに属するトラックは同時にロードされます。選択したエリアが変更されたり、新しいトラックが挿入されたりするたびに、選択したエリアのデータが更新されます。

表示幅を拡大し、チャートに表示されるデータが多すぎる場合には、データは部分的にしか表示されない。

Analyticsファイルのロード

Analyticsファイルは高速表示のために準備する必要があります。Target BrowserのAnalyticsファイル拡張子で、準備されていないAnalyticsファイルのフォルダに緑色の矢印が表示されます。この矢印をクリックすると新しいウィンドウが開き、Analyticsファイルが高速表示用に準備されます。

マーカ

チャートには、マーカを表示できます。これらは特定の位置に留まりますが、X軸をクリックしてドラッグすることで移動させることができます。

マーカには2種類あります。標準マーカには、マーカ線の上にテキストが表示されます。テキストはマーカ線と同じ色でハイライトされます。2つ目のタイプのマーカは、データを切り取るためのものです。Data Track Editorには、これらのマーカは最大2つまでしか配置できません。3つ目のマーカを挿入しようとすると、新しく挿入されたマーカに最も近いマーカが削除されます。切り取り用のマーカは線の上に文字がありません。ハサミと線は常に赤色で表示されます。

マーカをマウスで左クリックすると、ツールヒントが開き、マーカ位置の時間と値が表示されます。マーカをマウスで右クリックすると、コンテキストメニューが開きます。この中に、Data Track Editorからマーカを削除するオプションがあります。

6.3.1.3 ツールバー

Data Track Editorツールバーは、チャートの表示を操作したり、トラック内のデータを編集するための様々な機能を提供します。

これらの機能は、X軸およびY軸をリセットするために使用されます。

	チャートの表示幅を最大幅に設定し、Y軸のスケーリングをリセットします。
	Y軸のスケーリングをリセットします。

これらの機能のうち、一度に有効になるのは1つの機能だけです。これらはチャートを左クリックしたときの動作に影響します。

	この機能を有効にすると、チャート内をクリックしてX軸の範囲を選択できます。このためには、チャートをクリックし、マウスボタンを押したままマウスを左右に動かす必要があります。これにより選択プレビューが表示されます。マウスボタンを離すと、表示幅は選択エリアに縮小されます。
	この機能を有効にすると、チャート内をクリックしてY軸の範囲を選択できます。マウスを押すと、選択エリアのプレビューが表示されます。マウスを離すと、Y軸のスケーリングは選択した範囲に縮小されます。
	この機能を有効にすると、チャート内をクリックしてX軸とY軸の表示エリアを縮小できます。チャート上でマウスを押すと、表示されるエリアのプレビューが表示されます。マウスを離すと、Y軸のスケーリングと表示幅が選択したエリアに縮小されます。
	この機能を有効にすると、チャートをクリックして新しいマーカを挿入できます。マーカはトラックにも表示され、そこで名前を変更できます。
	この機能を有効にすると、チャートをクリックして特殊マーカを挿入し、エリアを切り取ることができます。

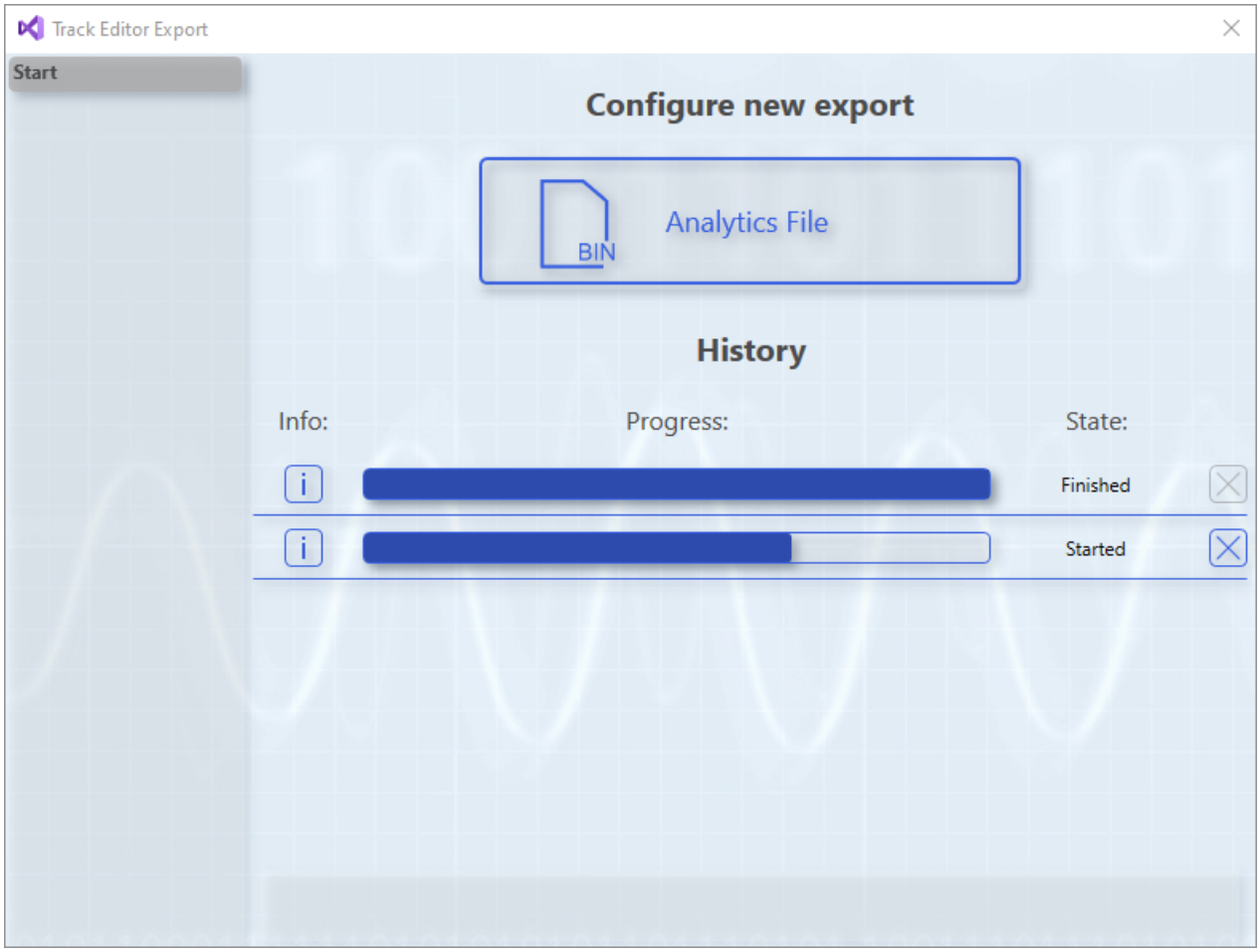
これらの機能は、トラックのデータを編集するために使用できます。

	データを削除できる以下の機能を使用して、表示トラックのみを処理するか、非表示トラックも処理するかを決定します。
	この機能は、切り取り用に2つのマーカが挿入されている場合にのみ使用できます。 を使用するか、グラフ上で右クリックして挿入できます。実行すると、2つのマーカ間のエリアが削除されます。
	この機能は、トラックで選択エリアを削除するために使用できます。それまでチャート上に表示されていたエリアが削除されるため、選択エリアのサイズが縮小されます。
	この機能は、選択範囲以外の全てのエリアを削除します。
	この機能は、全てのトラックのデータ量が同じではない場合にのみ使用できます。この差は、トラックのタイムラインで赤の破線で表示されます。実行すると、より多くのデータを含むトラックから、最後のデータが削除され、最もデータ数が少ないトラックと同じ数になるように調整されます。
	ここでは、Curve Creator（曲線クリエーター）を新しいウィンドウで開くことができます。これには、表示トラックの選択したエリアのデータが含まれます。Curve Creatorを終了すると、グラフはAnalyticsファイルに保存され、トラックに置き換えられます。Curve Creatorで、始点または終点を変更する機能が使用されている場合、データの元となる範囲が置き換えられます。そうでない場合は、以前に選択されていた範囲が置き換えられます。 この機能は、選択した範囲の値が100,000より小さい場合にのみ使用できます。

6.3.1.4 エクスポート

Data Track Editorは、Track Editorエクスポートウィザードを使用してエクスポートできます。ウィザードを呼び出すには、トラックのタイムラインの隣にあるボタン をクリックします。

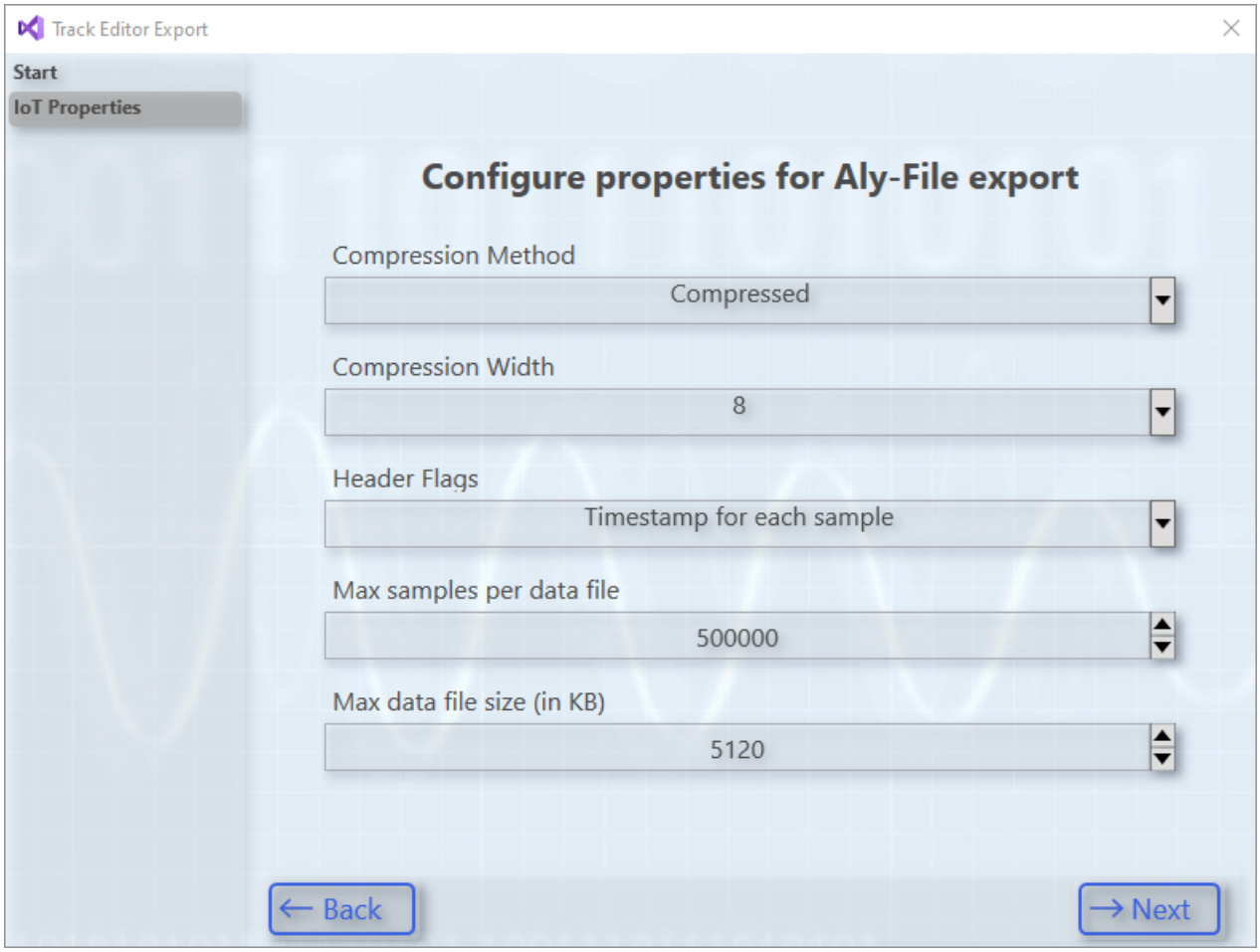
スタートページ



ウィザードのスタートページ上部から、新規エクスポートを開始できます。利用可能な全てのエクスポート形式は、個別のボタンとして表示されます。下部には、プロジェクトが開いている間に開始されたエクスポートの履歴があります。履歴の各エントリーの左側には、エクスポートに関する情報を表示するためのボタンがあります。表示情報には、エクスポートの開始時間、エクスポート形式ごとの重要なプロパティ、エラーが発生した場合のエラーメッセージが含まれます。右側には進捗バーが表示され、エクスポートするサンプルのうち、何件が既にエクスポートされたかを示します。その後、エクスポートのステータスが出力されます。右端のボタンは、実行中のエクスポートをキャンセルするために使用できます。

新規エクスポートの開始ボタンをクリックすると、ウィザードに新しいページが表示され、形式の設定ができます。

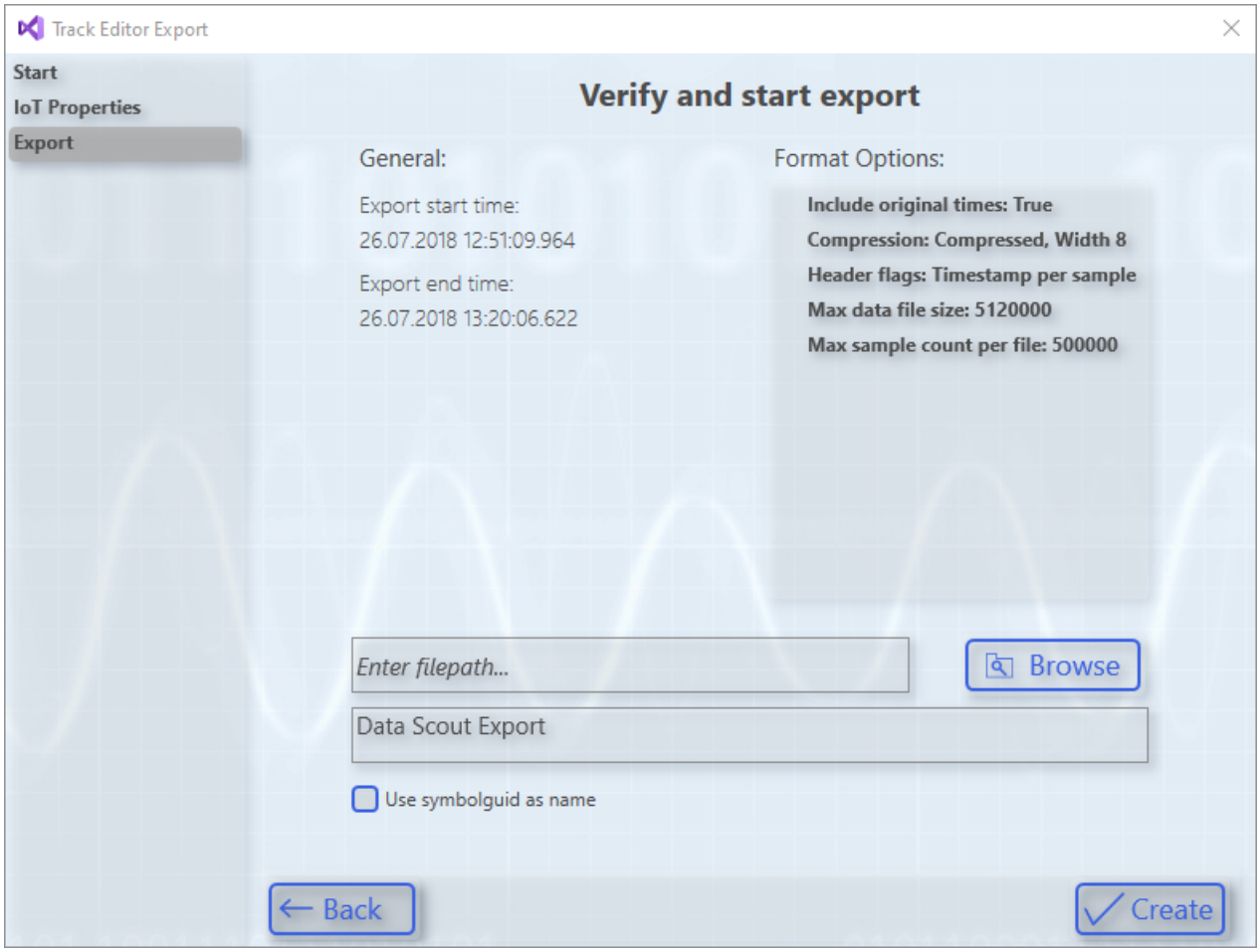
Analyticsファイル設定ページ



Analytics File 形式の設定ページでは、次の設定をします：

圧縮方式	Analyticsファイルは、圧縮または非圧縮で保存できます。()
圧縮幅	このオプションは、Analyticsファイルを圧縮形式で保存する場合にのみ使用できます。これは、圧縮時に比較対象とする最大幅を指定します。
ヘッダーフラグ	フラグ中では、.tayファイルに1つの時間値だけを持たせるか、.tayファイル内の各サンプルにそれぞれ異なる時間値を持たせるかを指定できます。
最大サンプル数とファイルサイズ	この2つのプロパティで、Analyticsファイルの.tayファイルのサイズを設定できます。また、ファイルに保存するサンプル数の上限を設定できるほか、ファイルシステム上のファイルサイズをキロバイト単位で設定できます。これらの値のいずれかを超過すると、新規.tayファイルが作成されます。

エクスポートページ



エクスポートページは、ファイル形式ページの設定後に表示されます。ここで全ての設定が再びリストアップされ、必要に応じて保存場所を指定できます。上のテキストフィールドはフォルダ、下のテキストフィールドはファイル名を示しています。

Analyticsファイル形式では、ファイル名を指定する代わりにチェックボックスをオンにすると、シンボルIDがフォルダ名として使用されます。

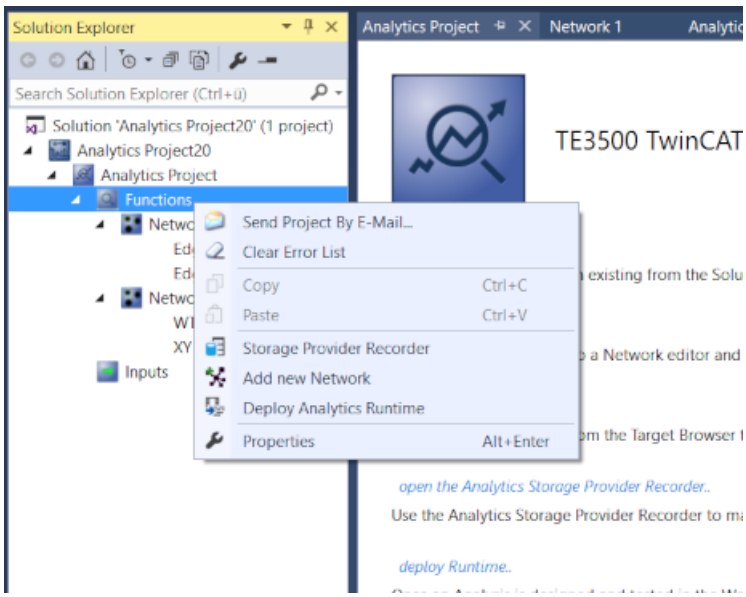
Createボタンをクリックすると、バックグラウンドでエクスポートが開始されます。その後、このページは閉じ、ウィザードはスタートページに戻ります。新規エクスポートは履歴にも表示され、進捗状況を追跡できます。

6.4 エディタの基本

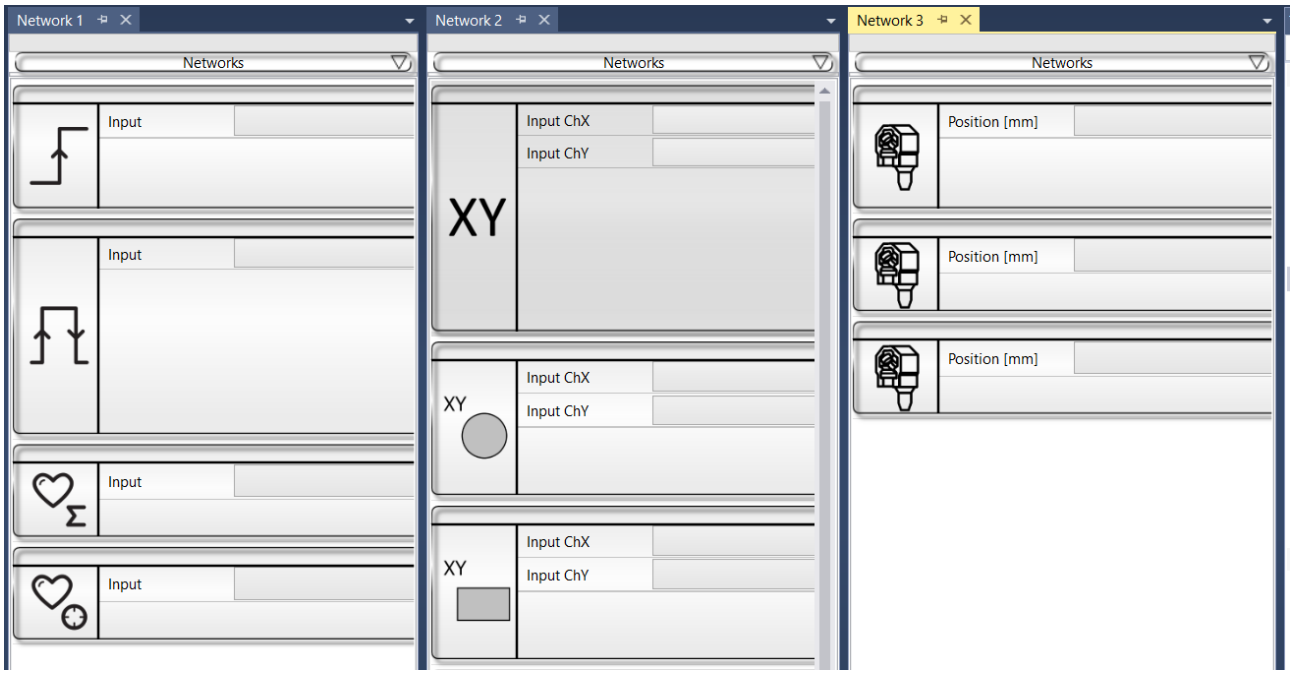
6.4.1 ネットワーク

ネットワークは解析情報を整理し、構造化するのに適しています。これにより、解析の透明度が大幅に向上します。さらに、テンプレートとして保存するための、アルゴリズムのコンテナとしても機能します。

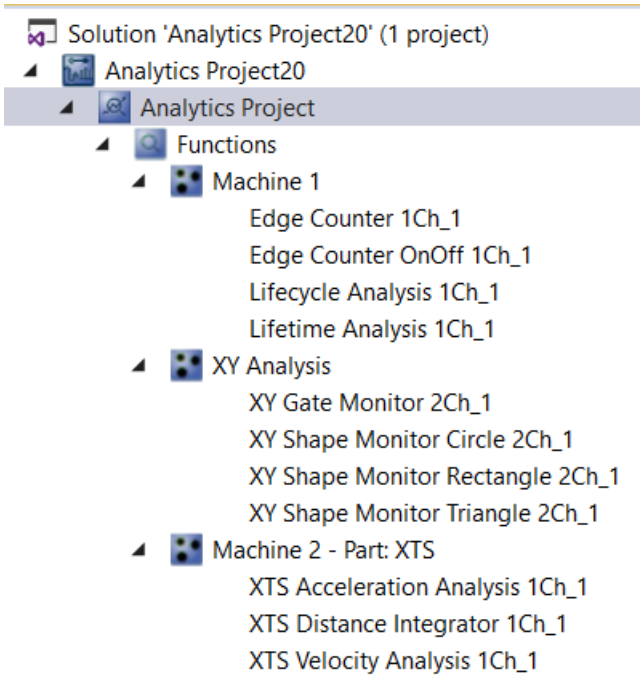
ネットワークは、Analyticsプロジェクトのホーム画面から、またはコンテキストメニューから追加できます。



各ネットワークは個別のタブで表示されます。これにより、ネットワークを個別に表示できます。横並びに表示したり、積み上げて表示したりできます。



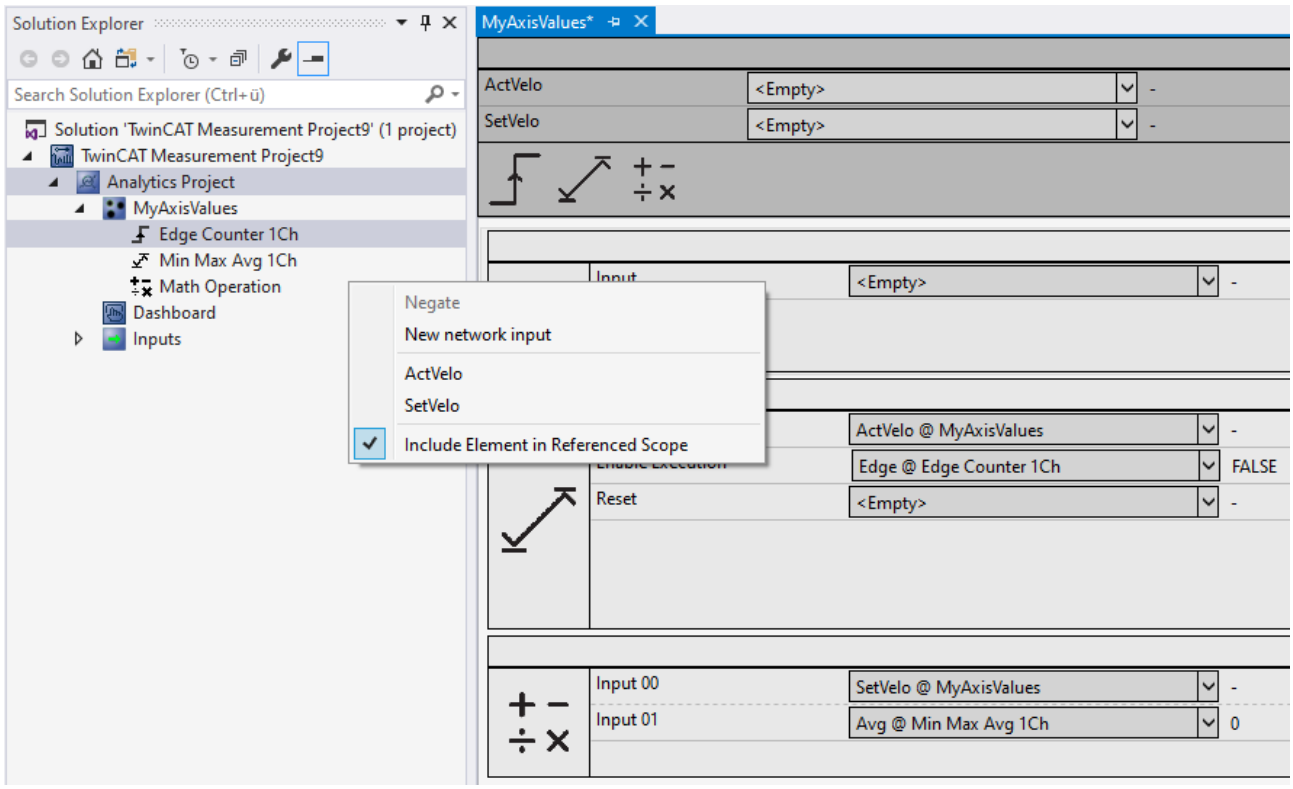
さらに、ネットワーク名を変更して（Solution Explorerで選択したネットワーク要素上でF2）、異なる機械、機械部品、その他の関連する接続などを指定してネットワークを作成するオプションもあります。



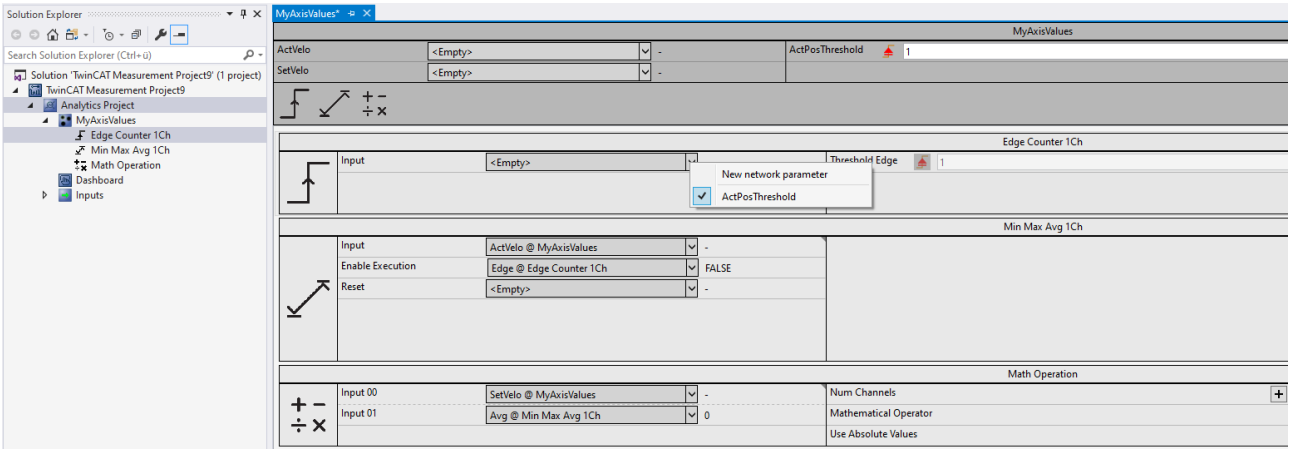
6.4.1.1 テンプレートとしてのネットワーク

ネットワーク内のアルゴリズムの入力、パラメータ、出力を、ネットワークの外部に直接ピン留めできます。よって、ネットワーク自体に入力、パラメータ、出力があります。これにより、繰り返し行う解析をテンプレートとして保存し、何度もインスタンス化できます。

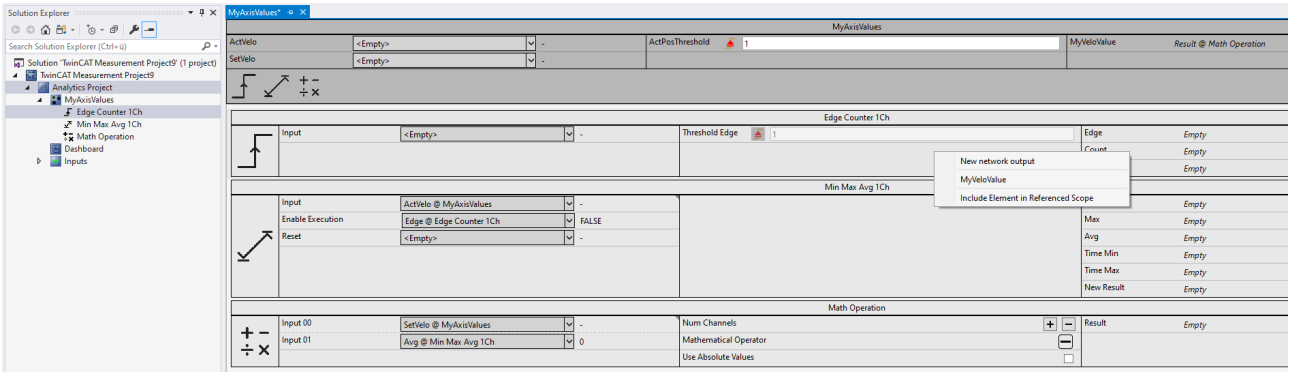
解析を作成したら、まず利用可能な入力のうち、どの入力をネットワークの外から見えるようにするかを選択する必要があります。これを行うには、目的の入力を選択し、右クリックでコンテキストメニューに切り替えます。そこで、入力を既存ネットワークの入力にリンクするか、新規ネットワーク入力を定義できます。



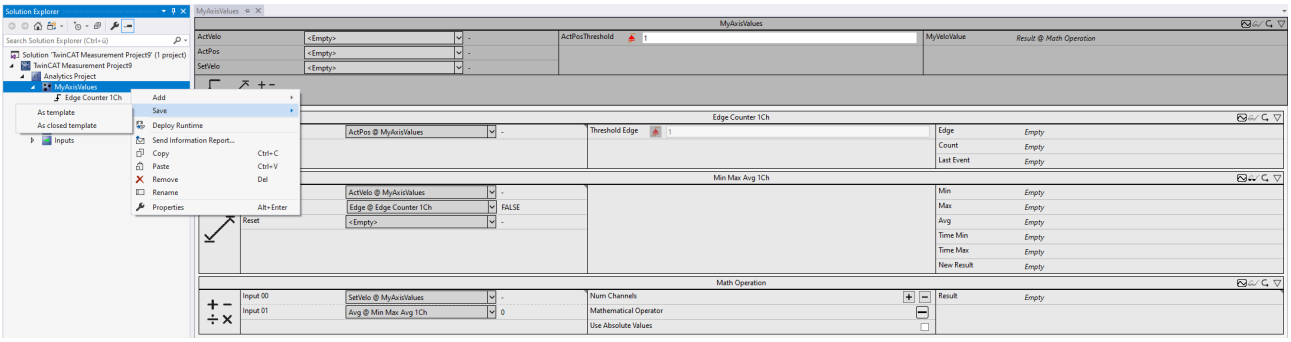
パラメータと出力についても同様の手順が可能です。



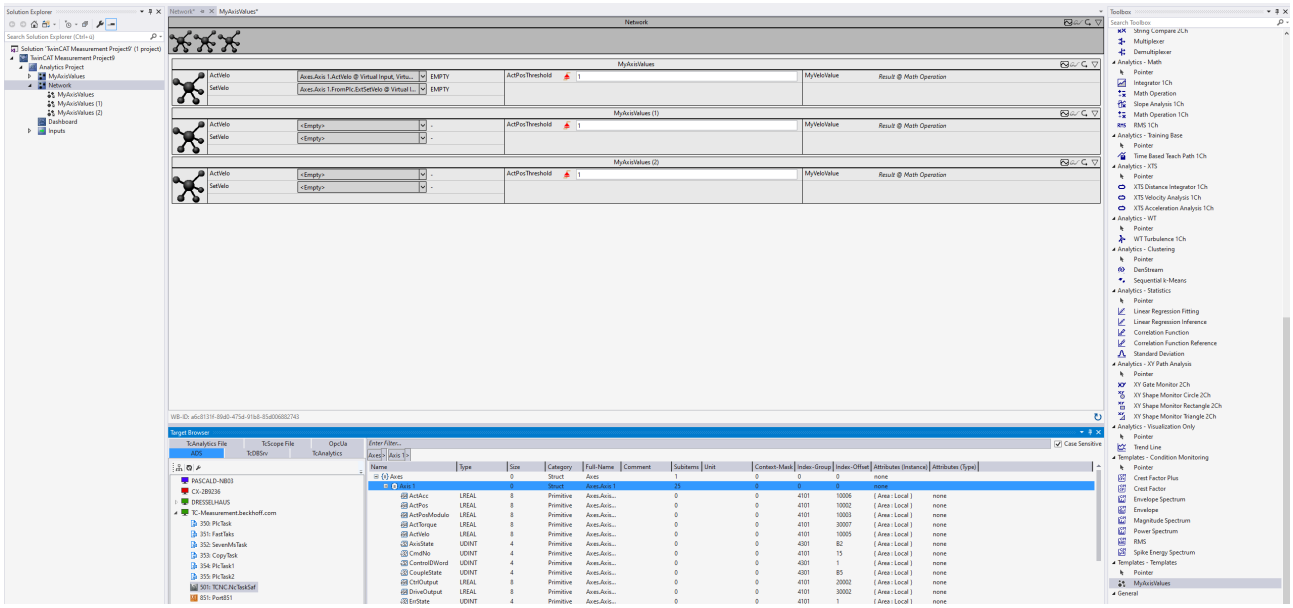
入出力に加えて、動的な入出力を指定することも可能です。この機能は、選択したアルゴリズムが機能をサポートしている場合に自動的に提供されます。背景には、演算アルゴリズムの入力のように、パラメータを介してアルゴリズムの入出力の数を増やすオプションがあります。



テンプレートとして使用するネットワークの定義が完了したら、保存します。保存するには、Solution Explorerでネットワークを右クリックします。コンテキストメニューのSaveオプションを使用します。Templete（テンプレート）か、Closed Templete（クローズドテンプレート）から選択できます。シンプルなテンプレートであれば、インスタンス化した後に内部を確認し、基本的なアルゴリズムの相互接続を確認したり、変更したりできます。このオプションはクローズドネットワークでは利用できません。ただし、知的財産の保護の機能は提供されていません。内部ロジックは、Analytics Workbenchで生成の可能性があるPLCコードにも表示されます。



保存後、テンプレートはツールボックスで選択でき、繰り返し解析に使用できます。



Target Browserには特別な機能があります。
ネットワーク名が、構造体の変数と正確に対応するよう設定されている場合、軸構造体のような完全な構造体を、ネットワークの x 入力のいずれかに直接ドラッグすることができます。全てのネットワーク入力と構造変数の名前が一致する場合は、自動的にマッピングされます。

ネットワークテンプレートに保存されたスコープの設定

作成したスコープの設定は、関連するネットワークと共にnetwork templete（ネットワークテンプレート） [▶ 352] に保存できます。ネットワークが再度使用される際に自動的に同じスコープの設定を取得できます。

6.4.2 有効化と無効化

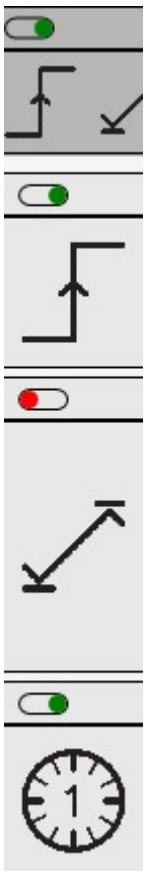
TwinCAT Analytics Workbenchは、データ解析において、個々のアルゴリズムや複雑なネットワークのオン/オフを切り替える機能を提供します。本書では、この機能の使い方と利点を説明します。

この機能は、特に解析の一部分を最適化したり、無効化したりすることにより、全体の動作に影響を与えることができます。

解析におけるモジュール無効化のメリットと適用例：

- 精度の向上：
  データ解析で、アルゴリズムやネットワークのオン・オフ切り替えができるため、解析精度を向上できます。アルゴリズムが適切に動作していない場合や、特定のネットワークが関連しない場合は、それらを無効化して解析全体の精度を向上できます。
- リソース使用の最適化：
  アルゴリズムやネットワークのオン・オフ切り替えも、リソース使用の最適化に役立ちます。例えば、あるアルゴリズムが高い計算能力を必要とする場合、他の解析分野にリソースを集中させるために、必要に応じてそのアルゴリズムをオフにできます。
- 柔軟性：
  データ解析においてアルゴリズムやネットワークのオン・オフ切り替えが可能なため、柔軟性をもって解析をコントロールできます。ユーザは、特定のニーズに合わせてデータ解析をカスタマイズできます。
- デバッグ：
  この機能により、解析中の問題をデバッグできます。解析の特定部分に不具合がある場合、問題の切り分けと修正のために、影響を受けるアルゴリズムやネットワークを無効にできます。

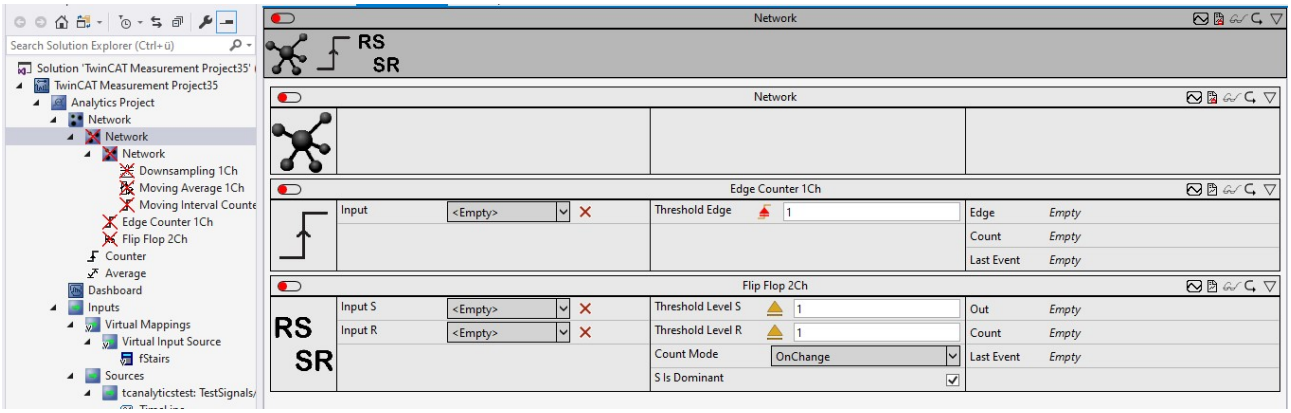
モジュールのオン・オフ切り替えは、各モジュールの左上隅にあるコントロール要素で制御できます。



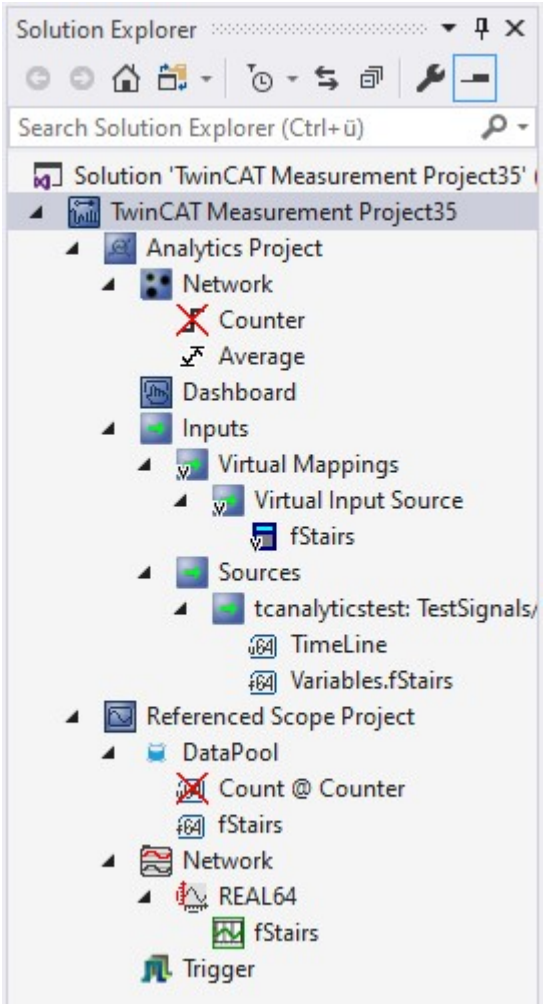
向き	色	ステータス
右	緑	有効
左	赤	無効

リンクされたモジュールの場合、リンクのチェーンはそのまま通過します。無効化されたモジュールに接続された入力には、赤い×印が表示されます。
2つの解析アルゴリズム間の無効化されたリンク：

ネットワークが無効化されると、そのネットワークとそのサブネットワーク（再帰型）以下の全てのアルゴリズムの処理も無効化されます。
無効化されたネットワークの要素とサブネットワークの要素：



参照スコープ（スコープとの相互作用 [▶ 348]を参照）を使用している場合、スコープにリンクされた要素も無効化の影響を受けます。
アルゴリズムを無効化すると、関連スコープの取得も無効化されます：

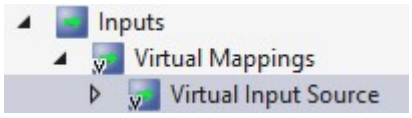


6.4.3 固定値

固定値は、アルゴリズムの実行中に変更しない定数を格納するために、解析アルゴリズムでよく使用されます。この固定値は、例えば、アルゴリズムで使用される閾値、スケーリング係数、その他の数学的な定数として機能します。

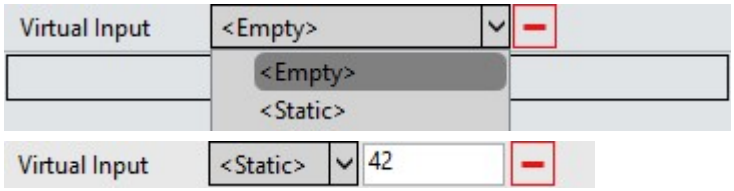
解析アルゴリズムに固定値を使用することには、様々な利点があります。第一に、各アルゴリズムの実行時に毎回行う必要のある計算を回避することで、アルゴリズムの性能を向上できます。代わりに、アルゴリズムは初期化時に設定された固定値にアクセスします。また、入力値をより明確に構造化し、定数に意味のある呼称を与えることで、アルゴリズムの可読性と理解しやすさを向上できます。

固定値は、TwinCAT Analytics WorkbenchのVirtual Inputでソートされます。後で中央システムでデータを作成、マージするためです。

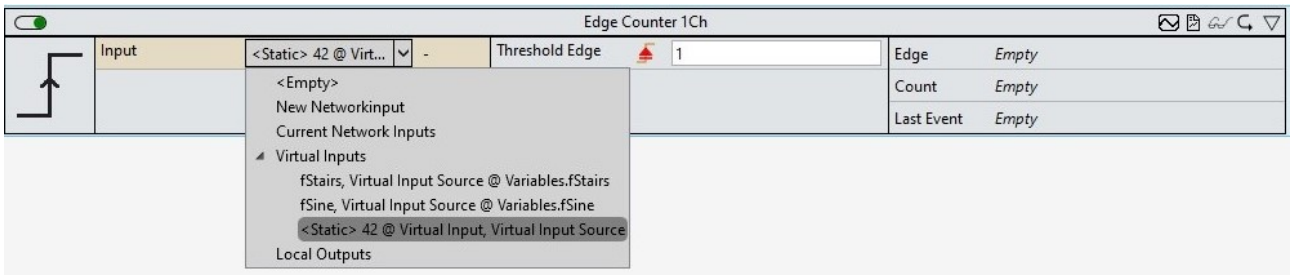


Virtual Input Sourceエディタでは、+を使用して、新規仮想入力を作成したり、既存の仮想入力を編集できます。

その後、ドロップダウンメニューでVirtual Inputを固定値（ここでは42）に設定できます。



この固定値は、入力モジュールを通じて全体の設定で選択できるようになりました。



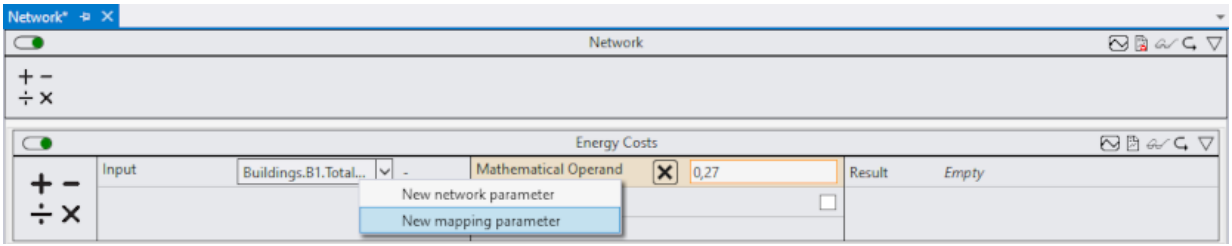
固定値は、ネットワーク入力でも使用できます。ここでは、各モジュールの従属入力とのリンクは、入力データに対する動作とまったく同じように実行されます。

解析アルゴリズムにおける固定値の使用には注意が必要です。アルゴリズムが適用されるデータまたはパラメータが変更された場合、アルゴリズムの最適なパフォーマンスを確保するために固定値を調整する必要がある場合があります。したがって、固定値は定期的にチェックし、アルゴリズムが正しく効果的に機能していることを確認し、必要に応じて調整します。

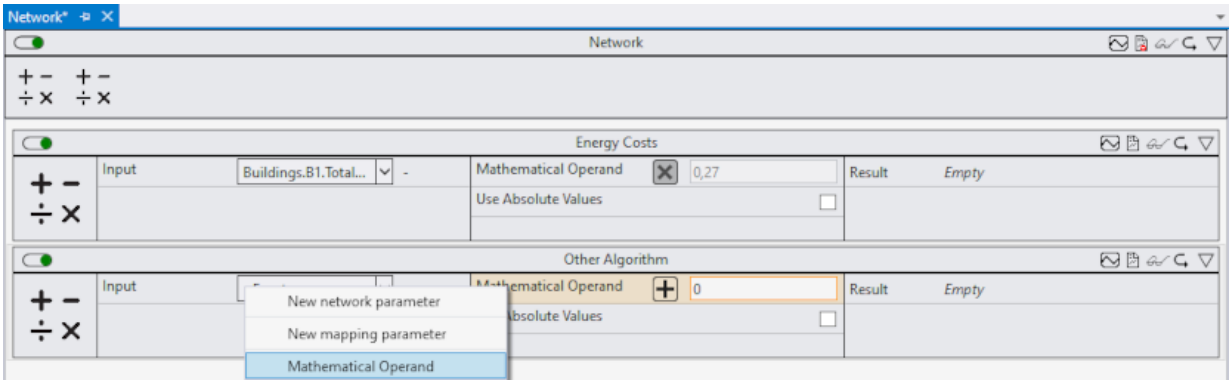
6.4.4 グローバルパラメータ

グローバルパラメータのオプションによって、エネルギー管理のためのコスト係数などの固定値を定義できます。これは異なるアルゴリズムで複数回、使用できます。

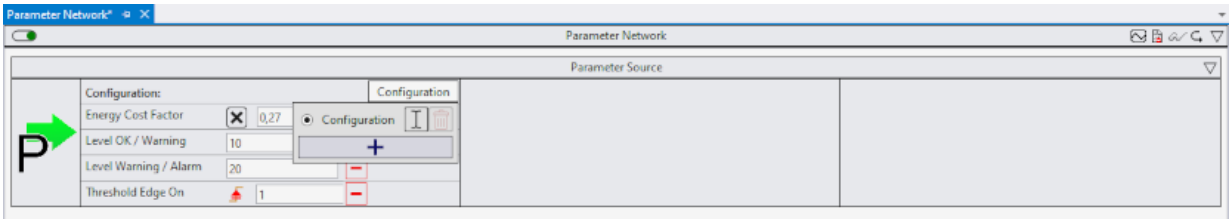
1. 新規グローバルパラメータを作成するには、既存のパラメータを右クリックし、**New mapping parameter**を選択します。

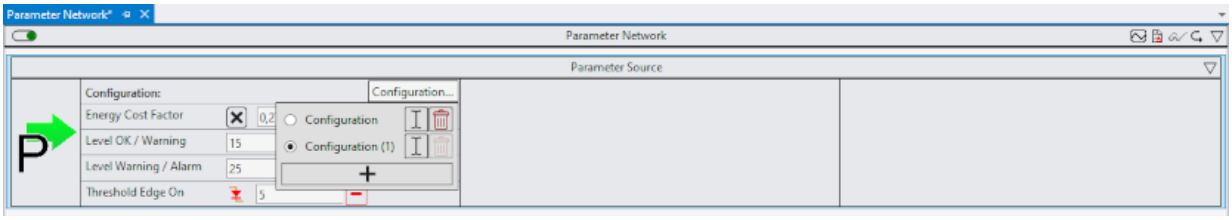


2. 既存のグローバルパラメータは、他のアルゴリズムでも同じコンテキストメニューから選択できます。

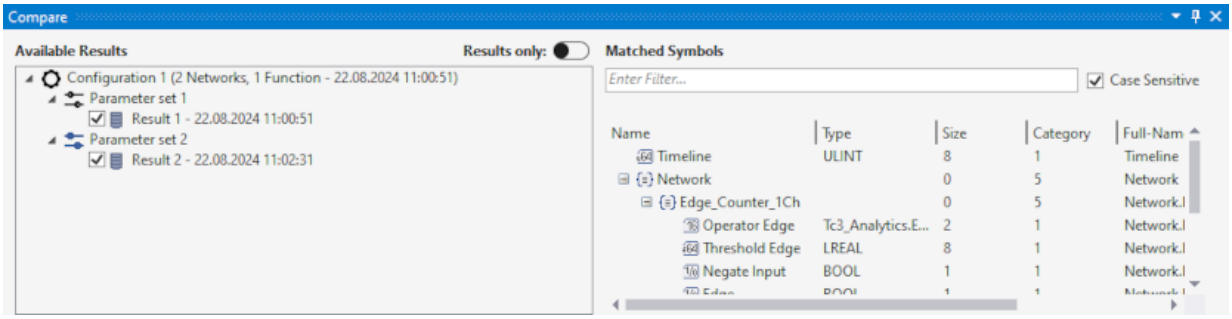


3. 最初のグローバルパラメータが作成されると同時に、新たに**Parameter Network**が作成されます。これにより、パラメータを一元的に設定できます。また、切り替え可能なパラメータ設定を複数定義することも可能です。



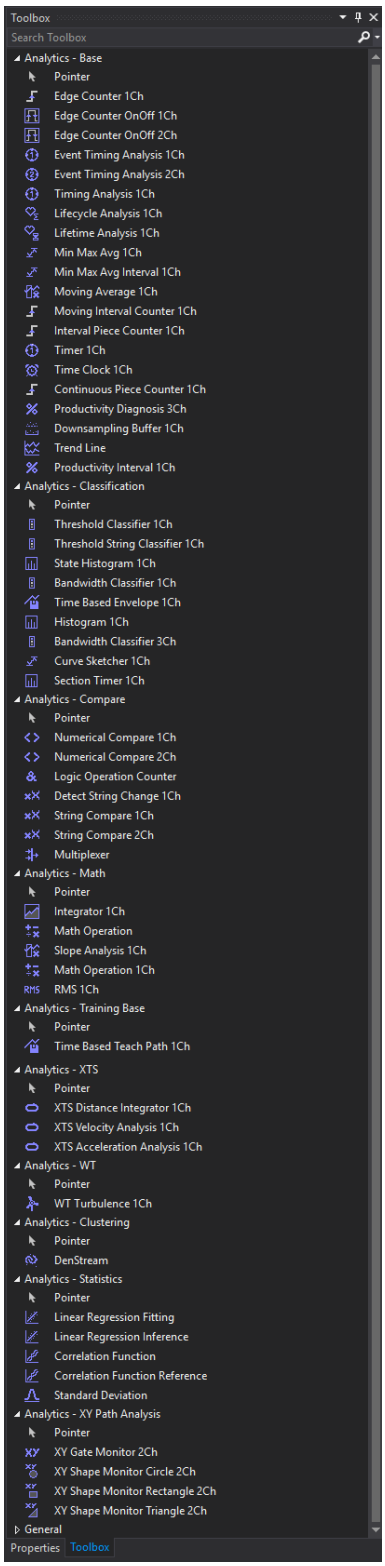


4. 異なるパラメータ設定でプロジェクトを連続して実行すると、これに対応する結果のデータセットを持つ新しい**Parameter Set**が、既存プロジェクト下のAnalytics Diary [▶ 531]に作成されます。以降、プロジェクトに変更を加えることはできません。変更する場合は、新規のプロジェクト設定が作成されます。



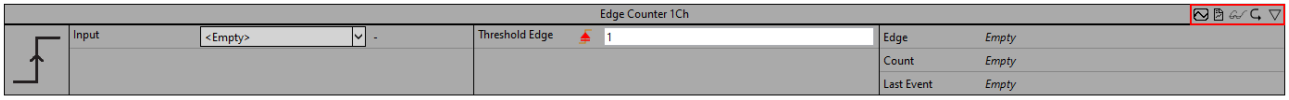
6.5 アルゴリズム

TwinCAT Analytics WorkbenchのコンフィグレータとTwinCAT Analytics Service Toolには、様々な解析アルゴリズムが含まれています。これら50以上のアルゴリズムが要件を満たさない場合は、Analyticsラムダ関数を使ってユーザ固有のアルゴリズムを開発できます。ツールボックスが空の場合は、Analyticsプロジェクトを選択してアルゴリズムを表示します。



現在、アルゴリズムには次の11グループがあります：Analytics-Base、Analytics-Classification、Analytics-Compare、Analytics-Math、Analytics-Training Base、Analytics-XTS、Analytics-WT、Analytics-XY Path Analysis、Analytics-Clustering、Analytics-Statistics、Analytics-C++ Lambda Functions ユーザ独自のアルゴリズムは、C++のラムダ関数を使用して開発できます（C++ラムダ関数 [▶ 272] を参照）。さらに、Analytics Toolboxに実装されている他のライブラリのアルゴリズムを使用することも可能です（その他のTwinCATライブラリのアルゴリズム [▶ 303] を参照）。テンプレートやC++ラムダ関数など、ユーザ固有の要素に関して、ツールボックスをクリーンアップするためにAnalytics Custom Toolbox Cleaner [▶ 302]を使用できます。

各アルゴリズムの右上には5つのアイコンがあります：



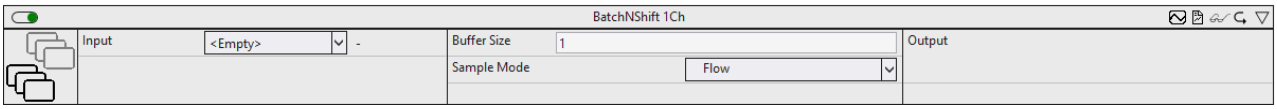
- **参照スコープに含める：** *Included in Referenced Scope*のアイコンをクリックすると、アルゴリズム出力が参照スコーププロジェクトに追加されます。
- **レポートに含める：***Include in Report*アイコンをクリックすると、設定された可視化アルゴリズムが手動レポートに統合されます。
- **メガネ：**メガネのアイコンをクリックすると、オプションのパラメータ*Enable Execution*が表示されます。このパラメータにはブール（Boolean）信号を選択でき、選択した信号の値が*TRUE*の場合にのみ、アルゴリズムが有効になります。
- **リセット矢印：**この矢印をクリックすると、特定のアルゴリズムの出力値がリセットされます。
- **最小化矢印：**右側の最小化矢印をクリックすると、アルゴリズムが折りたたまれます。

異なるアルゴリズムグループについて以下に説明します。

6.5.1 Analytics - Base（解析 - 基本）

*Analytics-Base*カテゴリのアルゴリズムは、プロセスおよびアプリケーションデータを解析するための基本的な機能を提供します。例えば、閾値検出、タイミング解析、最小値、最大値、平均値の計算などです。

6.5.1.1 BatchNShift 1Ch（バッチシフト 1Ch）



*BatchNShift 1Ch*は、バッファサイズとサンプルモードにしたがって、入力信号の値を一時的に保持（バッファリング）します。バッファリングされた入力値が格納される出力チャンネルの数は、バッファサイズに対応しています。サンプルモードにより、アルゴリズムの2つの異なる動作モードを区別できます。サンプルモードに*Flow*が選択されている場合、リングバッファまたはシフトレジスタ（Shift）が実現されます。値は次々にバッファに書き込まれ、各サイクルでバッファの位置が1つずつシフトされます。バッファがいっぱいになると、最も古い値が削除されます。*Wait*モードでは、バッファがいっぱいになるたびに内容がすべてクリアされ、新しい値で満たされます。このようにして、値はバッチ（batch）単位で処理されます。また、解析の開始時には、バッファがいっぱいになるまで待ってから、値をバッファに書き込みます。したがって、ファンクションブロックは、サイクル（*BufferSize + 1*）から有効な値のみを供給します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Buffer Size:**バッファのサイズ、つまり格納される値の数を指定します。出力チャンネル数はバッファサイズに等しくなります。
- **Sample Mode:**2つの異なるモードでバッファからの値を出力チャンネルに渡すことができます：
  - *Flow*：バッファはリングバッファのように満たされます。解析開始時、全ての出力値はゼロに設定されます。リングバッファに変更があるたびに、即座に出力チャンネルに転送されます。New Result（新規結果）フラグは、全ての出力チャンネルに値が割り当てられるとTRUEに設定され、新しい値がバッファに保存されると常にTRUEになります。
  - *Wait*：解析開始時またはリセット後、全ての出力チャンネルはゼロに設定されます。内部バッファが一杯になったときのみ、これらの値が出力チャンネルに転送され、New Resultフラグが TRUEに設定されます。これらの値は、内部バッファの値が全て更新されるまで出力値として残ります。そうなって初めて、出力チャンネルに転送されます。

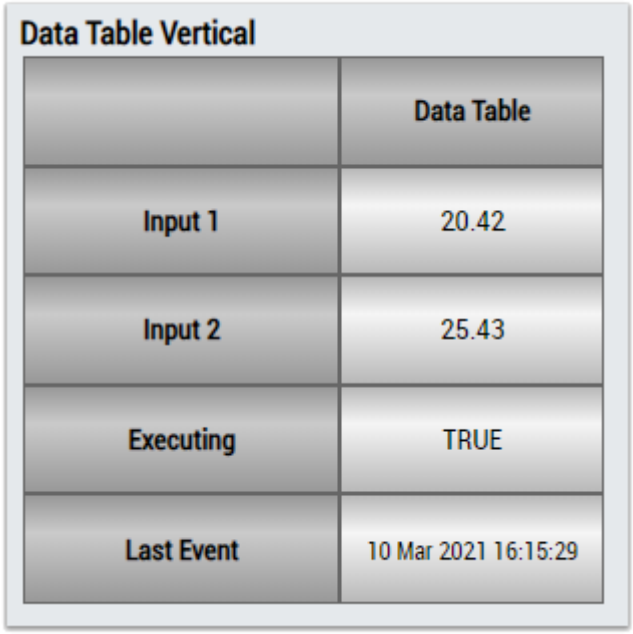
出力値

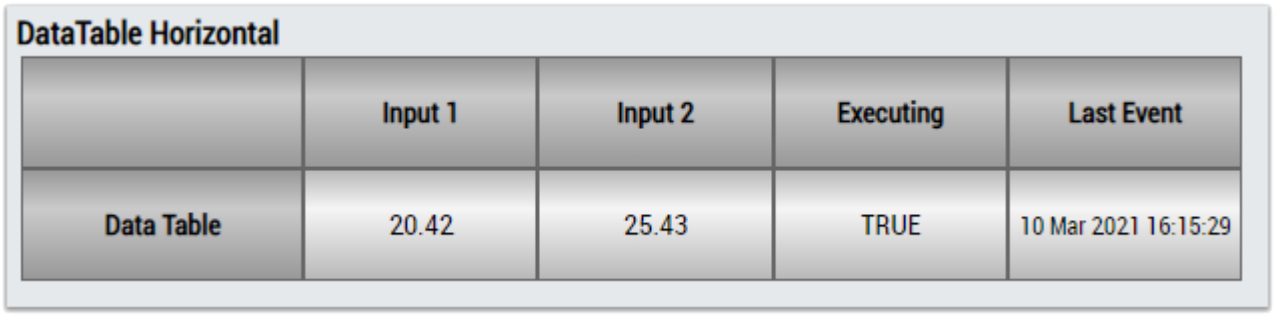
- **Output Value 00..n**: サンプルモードによるBatchNShiftバッファの結果。各出力チャンネルは、それぞれ1つのバッファ領域を表します。

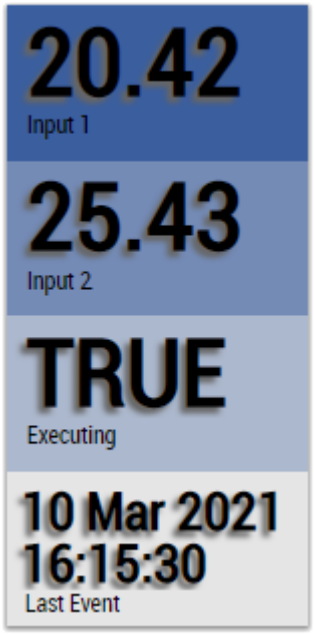
HMIコントロール

*BatchNShift 1Ch*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.Tableコントロール、または Multivalueコントロールは、出力値 *Output Value 00..n*を表示します。

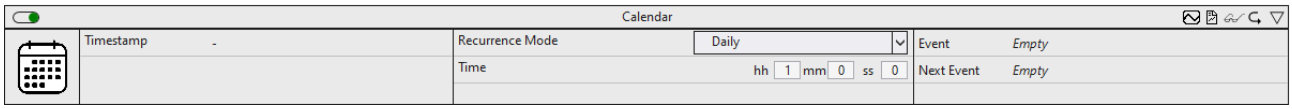






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*BatchNShift 1Ch*アルゴリズムにマッピングすることもできます。

6.5.1.2 Calendar



*Calendar*は特定の時間に、定期的なイベントを作成します。繰り返しモードは個別に設定できます。毎日、毎週、毎月、毎年の繰り返しのモードが利用可能です。
アルゴリズムが動作するための時間コンテキストが必要であるため、参照値としてのタイムスタンプが必要です。この参照タイムスタンプは、コンフィギュレーションに別のアルゴリズムがある場合、自動的に設定されます。したがって、カレンダーを個別に使用することはできません。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Recurrence Mode:**
  Daily：毎日繰り返し。イベントの時間を設定できます。
  Weekly：1週間ごとに繰り返し。イベントの時間と曜日を設定できます。
  Monthly：毎月繰り返し。イベントの時間と繰り返しタイプを設定できます。
  Yearly：毎年繰り返し。イベントの時間、繰り返しタイプ、イベント月を設定できます。
- **Time:** イベントの時間。
- **Day Of Week Mask:** イベントの曜日。
- **Recurrence Specification:**
  Specific Day：特定の日にイベントを設定します。これは*Day*パラメータで設定します。
  First：その月の最初の指定曜日にイベントを設定します。*Day Of Week*パラメータで設定します。
  Last:その月の最後の指定曜日にイベントを設定します。*Day Of Week*パラメータで設定します。
- **Day:** イベントを設定する日。
- **Day Of Week:** イベントの曜日。
  *Unspecified:*イベントは、*Recurrence Specification*パラメータによって、月の最初または最後の日に設定されます。
  Monday..Sunday：イベントは、*Recurrence Specification*パラメータによって、月の最初または最後の平日に設定されます。
- **Month Of Year Mask:** イベントの月。

出力値

- **Event:**イベント時に*TRUE* を示し、そうでない場合は*FALSE* を示します。
- **Next Event:** 次のイベントまでの残り時間を表示します。

HMIコントロール

カレンダーアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Event, Next Event。

Data Table Vertical

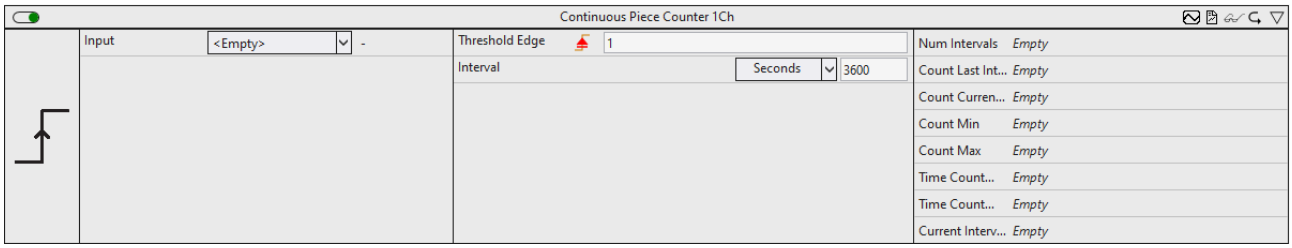
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Calendar*アルゴリズムにマッピングすることもできます。

6.5.1.3 Continuous Piece Counter 1Ch



*Continuous Piece Counter 1Ch*は、設定した区間内の個数をカウントします。入力チャンネルの信号が、設定されたエッジで特定の閾値を超過するとカウンタが増加します。時間区間が経過すると計算が再開されます。このアルゴリズムは、個数、最小個数、最大個数、および最小個数時と最大個数時の時間値を提供します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:** エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:** それぞれのエッジにおける信号の閾値。信号がこの閾値を超過すると、カウンタがインクリメントします。
- **Interval:** 値を計算する時間区間。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

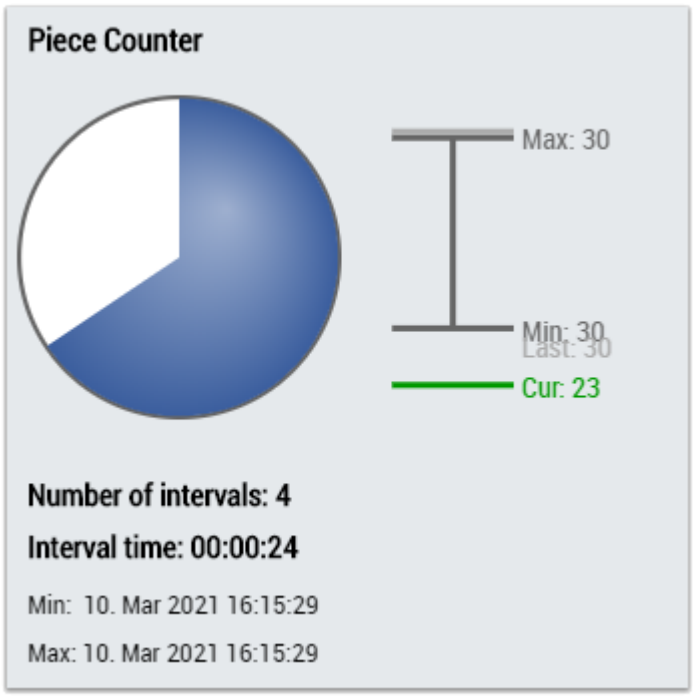
出力値

- **Num Interval:** 区間の数を示します。
- **Count Last Interval:**直近の区間に含まれる個数を示します。
- **Count Current Interval:**現在の区間に含まれる個数を示します。
- **Count Min:** 区間内の最小個数を示します。
- **Count Max:** 区間内の最大個数を示します。
- **Time Count Min：** 最小の時間値を示します。
- **Time Count Max：** 最大の時間値を示します。
- **Current Interval Time:**現在の区間の時間を示します。

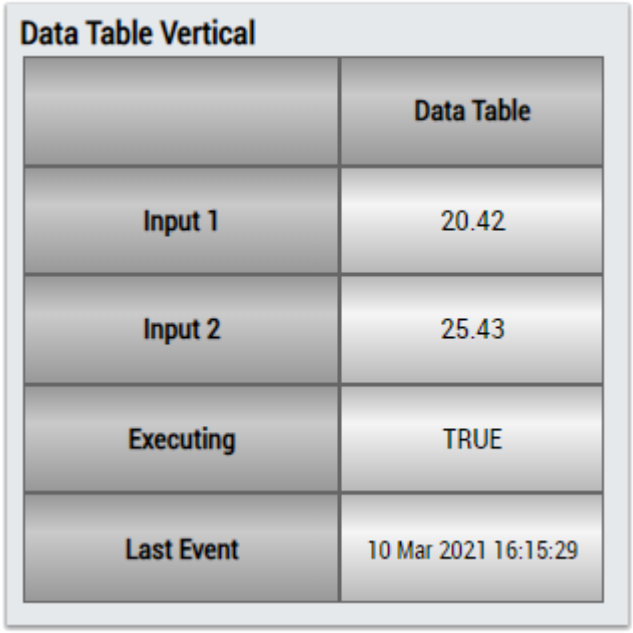
HMIコントロール

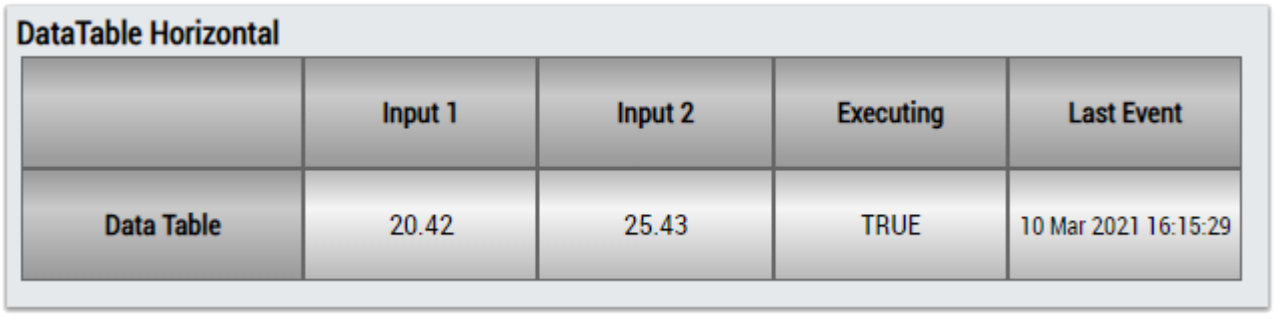
Continuous Piece Counter 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

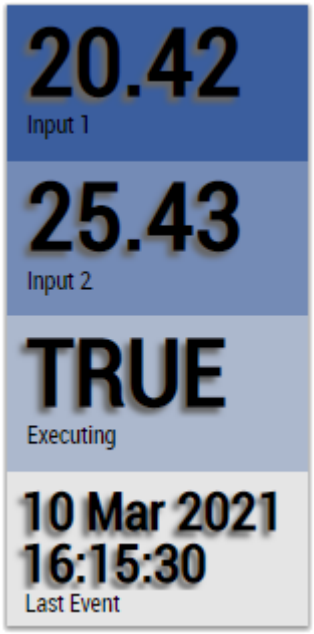
1.PieceCounter コントロールは、出力値 Num Intervals、Count Last Interval、Count Current Interval、Count Min、Count Max、Time Count Min、Time Count Max、Time Current Interval を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Num Intervals, Count Last Interval, Count Current Interval, Count Min, Count Max, Time Count Min, Time Count Max, Time Current Interval。

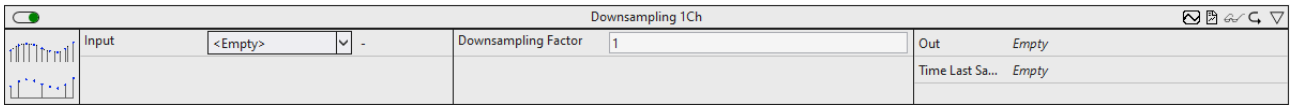






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをContinuous Piece Counter 1Chアルゴリズムにマッピングすることもできます。

6.5.1.4 Downsampling 1Ch



Downsampling 1Ch は、ダウンサンプリング係数を設定して、入力チャンネルの値を処理します。これにより、出力信号は低いサンプリングレートで入力信号を表現するダウンサンプリング処理の結果となります。これは、傾向を特定したり、解析において低いサンプリングレートのみ必要な場合に、高いサンプリングレートで取得した信号を圧縮するために有用です。これは、解析のパフォーマンスを上げる簡単な手法です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Downsampling Factor:**ダウンサンプリングに使用する係数。例えば、ダウンサンプリング係数が100の場合、100番ごとの値のみが保存されます。したがって、サンプル時間は元のサイクルタイムの100倍となり、これはサンプリングされた2つのデータポイントの時間区間に相当します。ダウンサンプリング係数が1に設定されている場合、全ての値がバッファリングされます。

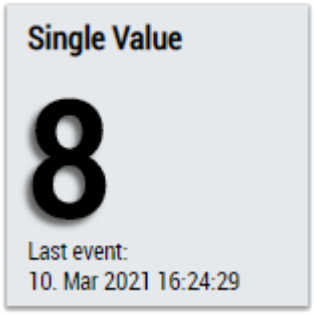
出力値

- **Output Value** : ダウンサンプリング係数でサンプリングレートを下げた出力信号。

HMIコントロール

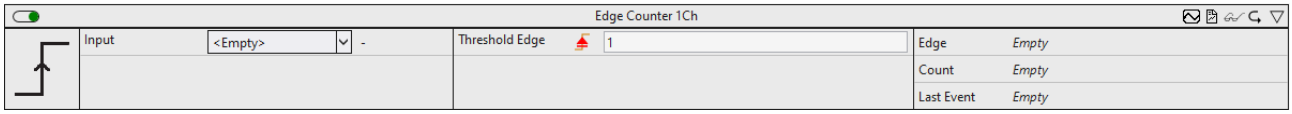
*Downsampling 1Ch*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.SingleValue コントロールは、出力値*Output Value*を表示します。



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Downsampling 1Ch*アルゴリズムにマッピングすることもできます。

6.5.1.5 Edge Counter 1Ch



*Edge Counter 1Ch*は、発生したイベント数をカウントします。入力チャンネルの信号が、設定されたエッジで特定の閾値を超過すると、イベントが発生します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

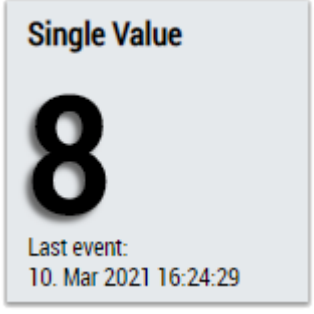
出力値

- **Edge:**イベントがトリガーされた時に*TRUE* を示し、そうでなければ*FALSE* を示します。
- **Count:** トリガーされたイベントの数をカウントします。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

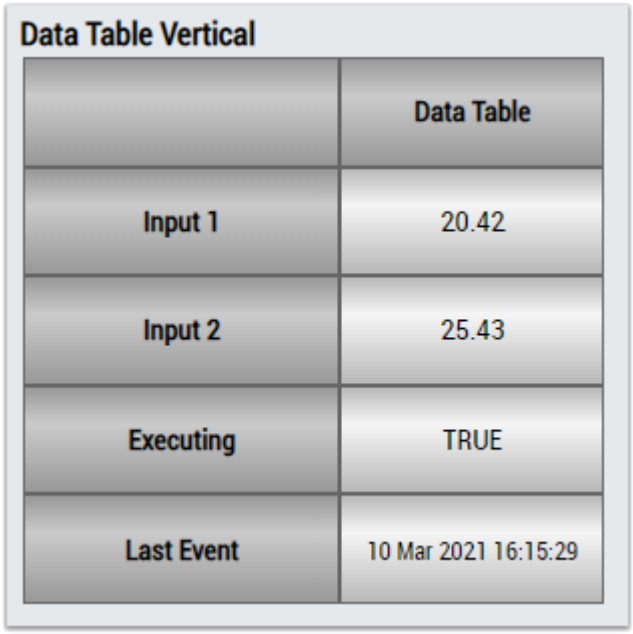
HMIコントロール

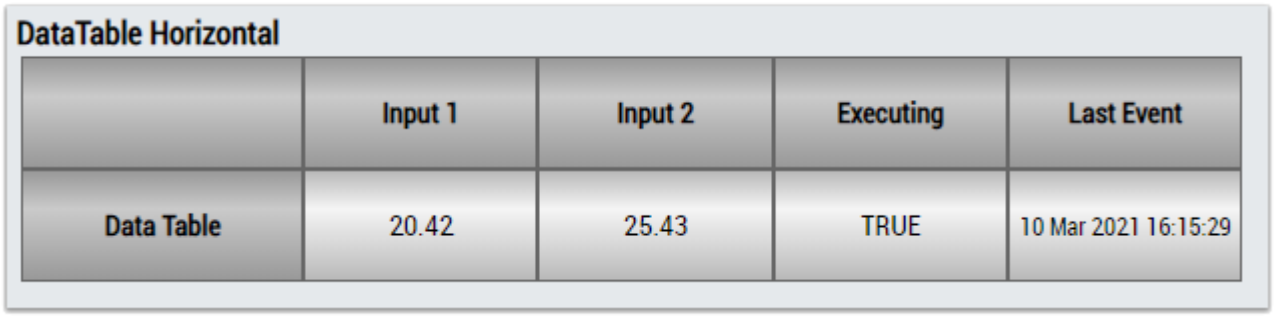
Edge Counter 1Chのアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

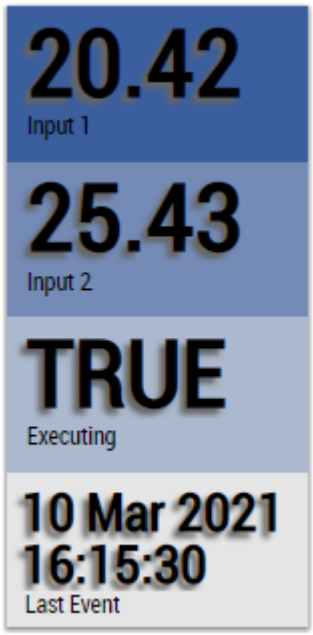
1.SingleValue コントロールは、出力値 Count と Last Event を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Edge, Count, Last Event。

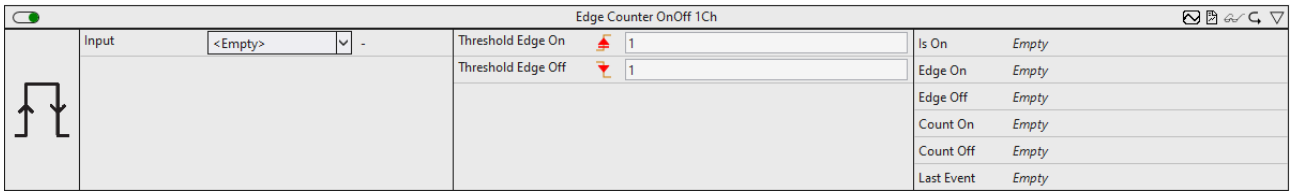






また、Mappingウィザード [▶ 438]を使用して、カスタムのHMIコントロールをEdge Counter 1Chアルゴリズムにマッピングすることもできます。

6.5.1.6 Edge Counter On Off 1Ch



Edge Counter On Off 1Ch は、オンイベントおよびオフイベントの発生回数をカウントします。オンイベントは、入力チャンネルの信号が設定されたエッジにおいてオン閾値を超過したときに発生します。オフイベントは、同じ信号がオフ閾値を超過したときに発生します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold On:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過すると、オンイベントがトリガーされます。
- **Threshold Off:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超えると、オフイベントがトリガーされます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

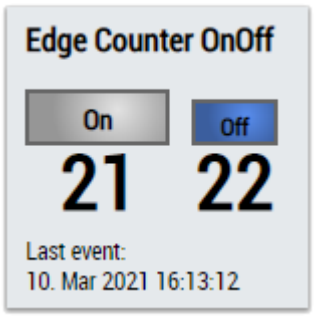
出力値

- **Is On:** オンイベントとオフ結果の間の時間内に*TRUE*を示します。それ以外は*FALSE*となります。
- **Edge On:**立ち上がりエッジで TRUE を示します。
- **Edge Off:**立ち下がりエッジで TRUE を示します。
- **Count On :** トリガーされたオンイベントの数をカウントします。
- **Count Off :** トリガーされたオフイベントの数をカウントします。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

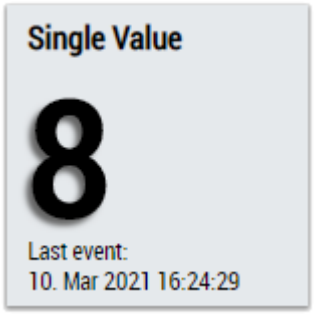
HMIコントロール

Edge Counter On Off 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

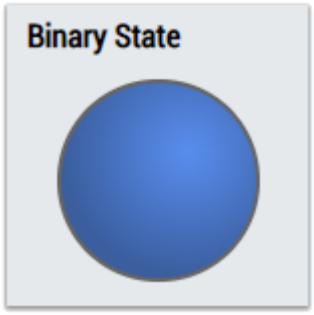
1.EdgeCounterOnOff コントロールは、出力値 Is On、Count On、Count Off、Last Event を表示します。



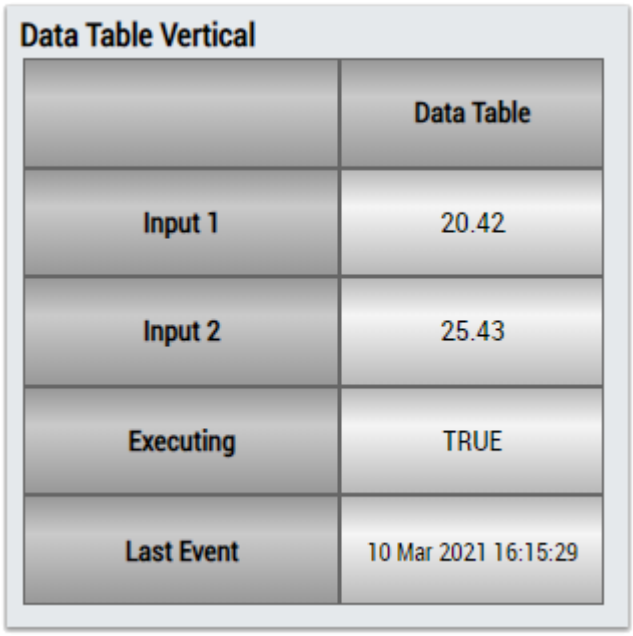
2.SingleValueコントロールは、出力値 Count OnとLast Eventを表示します。

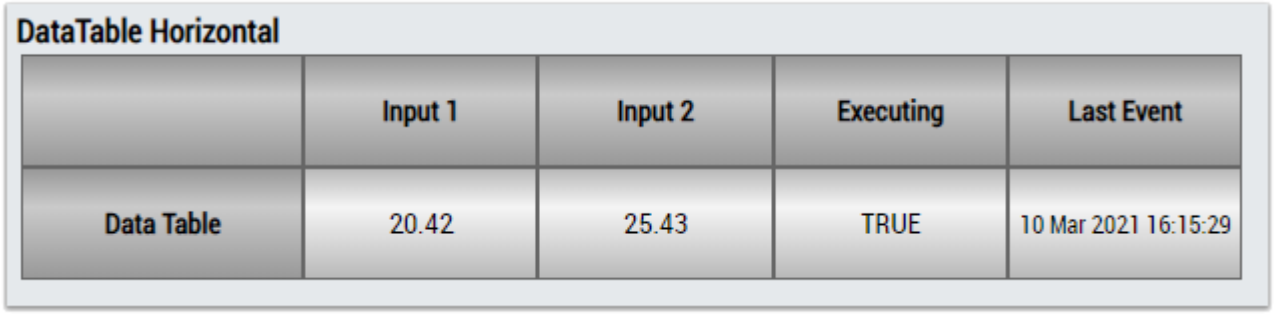


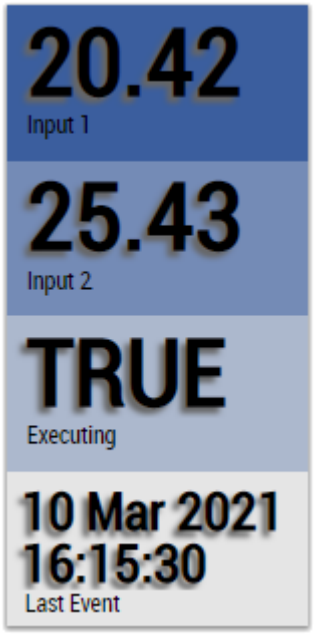
3.BinaryState コントロールは、出力値 Is On を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Flanks (Is On, Edge On, Edge Off), Count On, Count Off, Last Event。

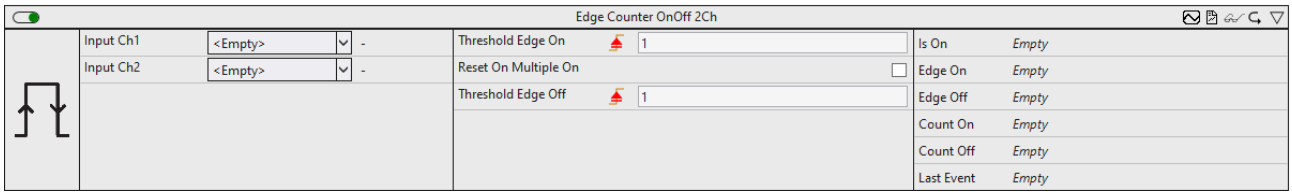






また、Mappingウィザード [▶ 438]を使用して、カスタムのHMIコントロールをEdge Counter On Off 1Chアルゴリズムにマッピングすることもできます。

6.5.1.7 Edge Counter On Off 2Ch



Edge Counter On Off 2Ch は、オンイベントおよびオフイベントの発生回数をカウントします。オンイベントは、第1の入力チャンネルの信号が設定されたエッジでオン閾値を超過したときに発生します。オフイベントは、第2のチャンネルの信号がオフ閾値を超過したときに発生します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold On:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過すると、オンイベントがトリガーされます。
- **Reset On Multiple On :** チェックボックスをオンにすると、オンイベントごとに 「Count On」のカウンタがインクリメントされます。そうでない場合、オンイベントはカウンタリセット（オフイベント）の後にのみカウントされます。
- **Threshold Off:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超えると、オフイベントがトリガーされます。
- **Tolerance（オプション） :** Equal / NotEqual 比較時に使用される許容値。

出力値

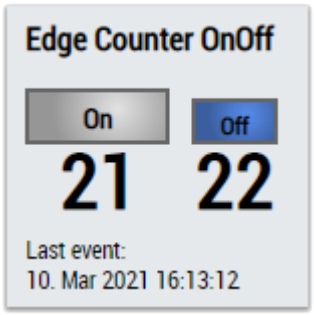
- **Is On:** オンイベントとオフ結果の間の時間内に*TRUE*を示します。それ以外は*FALSE*となります。
- **Edge On:**立ち上がりエッジで TRUE を示します。
- **Edge Off:**立ち下がりエッジで TRUE を示します。
- **Count On :** トリガーされたオンイベントの数をカウントします。
- **Count Off :** トリガーされたオフイベントの数をカウントします。

- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

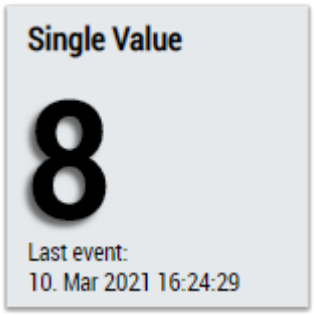
HMIコントロール

Edge Counter On Off 2Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

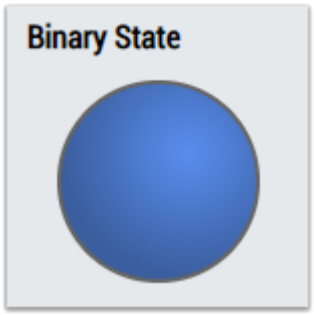
1.EdgeCounterOnOff コントロールは、出力値 Is On、Count On、Count Off、Last Event を表示します。



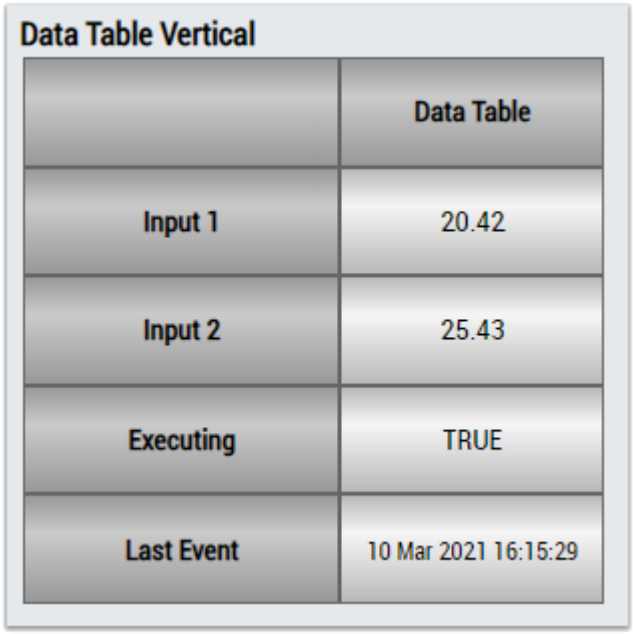
2.SingleValueコントロールは、出力値 Count OnとLast Eventを表示します。

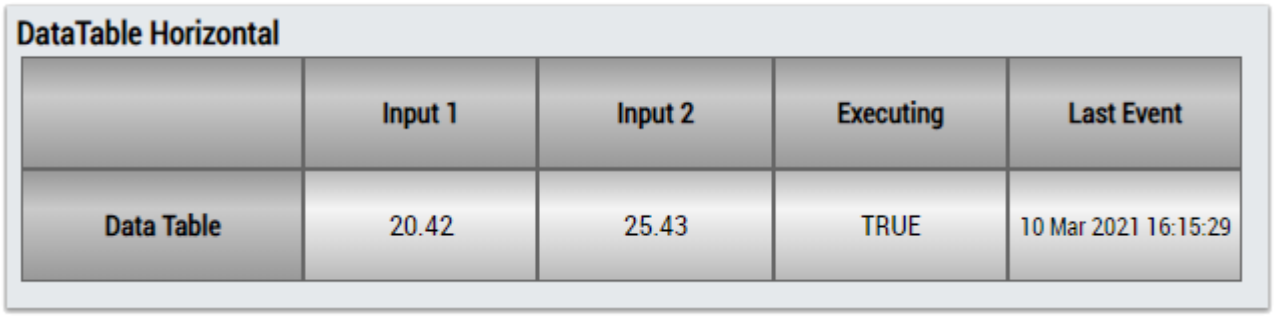


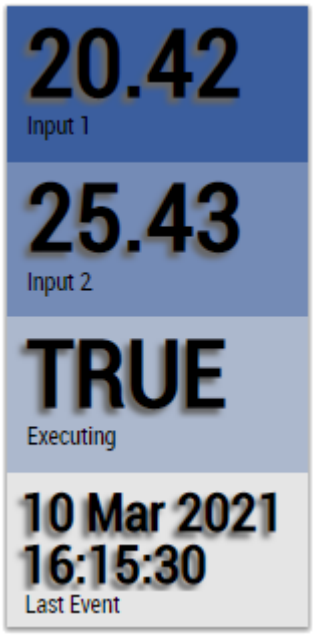
3.BinaryState コントロールは、出力値 Is On を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Flanks (Is On, Edge On, Edge Off), Count On, Count Off, Last Event。

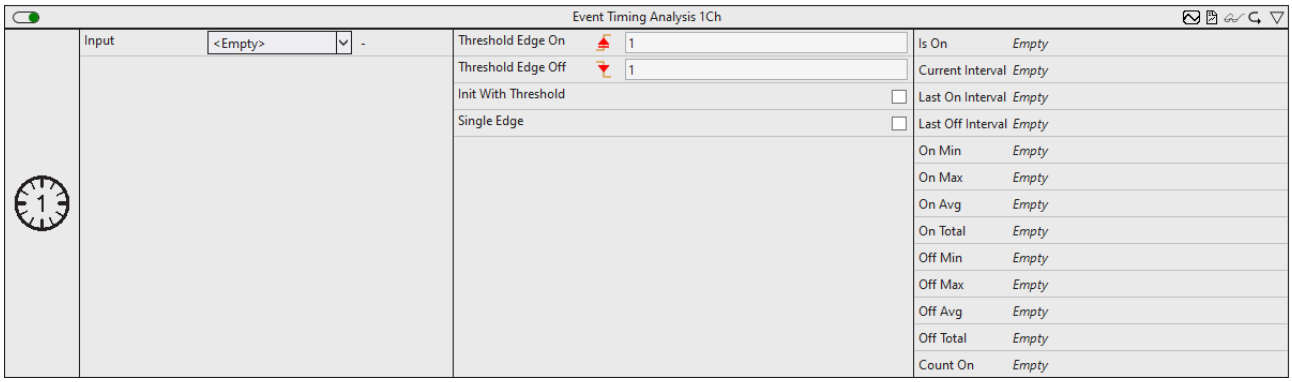






また、Mappingウィザード [▶ 438]を使用して、カスタムのHMIコントロールをEdge Counter On Off 2Chアルゴリズムにマッピングすることもできます。

6.5.1.8 Event Timing Analysis 1Ch



*Event Timing Analysis 1Ch*は、オンイベントとオフイベントの時間差を計測し、イベントの回数をカウントします。オンイベントは、入力チャンネルの信号が設定されたエッジにおいてオン閾値を超過したときに発生します。オフイベントは、同じ信号がオフ閾値を超過したときに発生します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを示します。
- **Threshold On:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過すると、オンイベントがトリガーされます。
- **Threshold Off:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超えると、オフイベントがトリガーされます。
- **Init With Threshold :** 値がTRUEの場合、アルゴリズムはエッジを待つ代わりに閾値を使用して内部状態を初期化します。
- **Single Edge :** 値がTRUEの場合、アルゴリズムはオンイベントのエッジのみに基づいて実行されます。
- **Tolerance (オプション):**Equal / NotEqual比較時に使用される許容値。

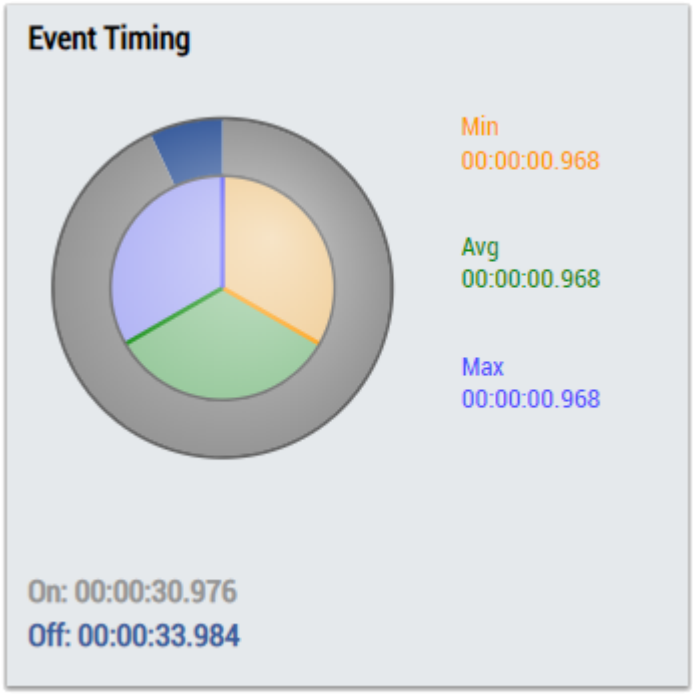
出力値

- **Is On:** オンイベントとオフ結果の間の時間内に*TRUE*を示します。それ以外は*FALSE*となります。
- **Current Interval:** 現在の区間の時間を示します。
- **Last On Interval:**直近に完了したオンイベントからオフイベントの間の時間区間。
- **Last Off Interval:** 直近に完了したオフイベントからオンイベントの間の時間区間。
- **On Min:** オンイベントからオフイベントまでの最小時間。
- **On Max:**オンイベントからオフイベントまでの最大時間。
- **On Avg :** オンイベントからオフイベントまでの平均時間。
- **On Total :** オンイベントからオフイベントまでの合計時間。
- **Off Min:**オフイベントからオンイベントまでの最小時間。
- **Off Max:**オフイベントからオンイベントまでの最大時間。
- **Off Avg :** オフイベントからオンイベントまでの平均時間。
- **Off Total :** オフイベントからオンイベントの間の合計時間。
- **Count On :** トリガーされたオンイベントの数をカウントします。

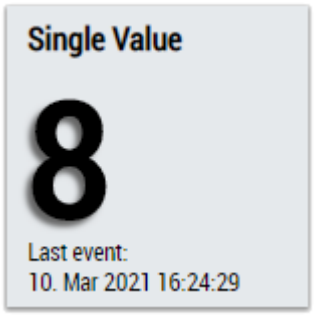
HMIコントロール

Event Timing Analysis 1Chのアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

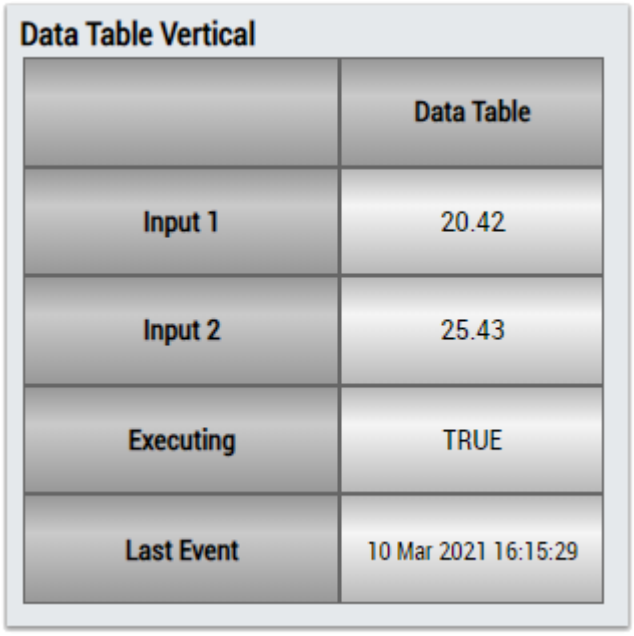
1.EventTiming コントロールは、出力値 Is On、Count On、Current Interval、On Min、On Max、On Avg、On Total、Off Min、Off Max、Off Avg、Off Total を表示します。



2.SingleValue コントロールは、出力値 Count On を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Is On、Count On、Current Interval、On Min、On Max、On Avg、On Total、Off Min、Off Max、Off Avg、Off Total。

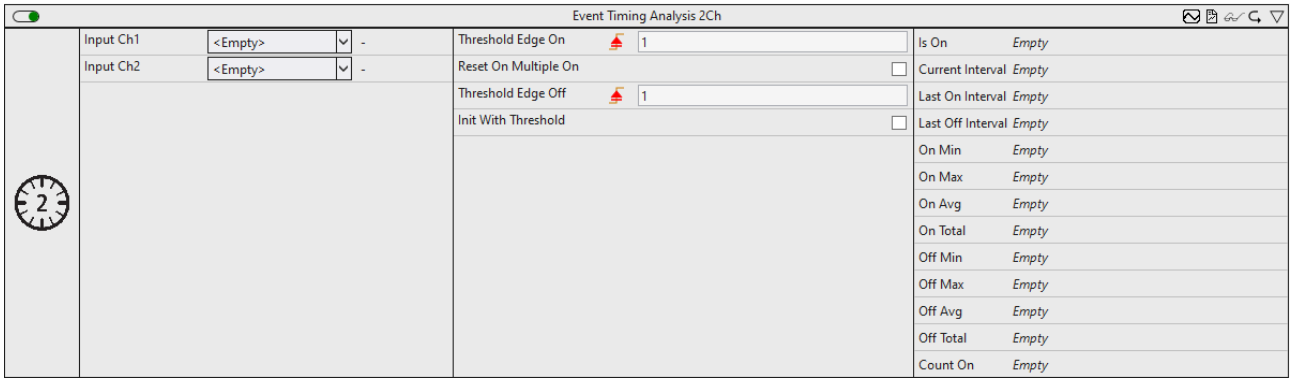


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをEvent Timing Analysis 1Chアルゴリズムにマッピングすることもできます。

6.5.1.9 Event Timing Analysis 2Ch



*Event Timing Analysis 2Ch*は、オンイベントとオフイベントの時間差を計測し、イベントの回数をカウントします。オンイベントは、第1の入力チャンネルの信号が設定されたエッジでオン閾値を超過したときに発生します。オフイベントは、第2のチャンネルの信号がオフ閾値を超過したときに発生します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold On:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過すると、オンイベントがトリガーされます。

- **Reset On Multiple On :** チェックボックスをオンにすると、オンイベントごとに 「Count On」のカウンタがインクリメントされます。そうでない場合、オンイベントはカウンタリセット（オフイベント）の後にのみカウントされます。
- **Threshold Off:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超えると、オフイベントがトリガーされます。
- **Init With Threshold:** 値がTRUEの場合、アルゴリズムはエッジを待つ代わりに、閾値を使用して内部状態を初期化します。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

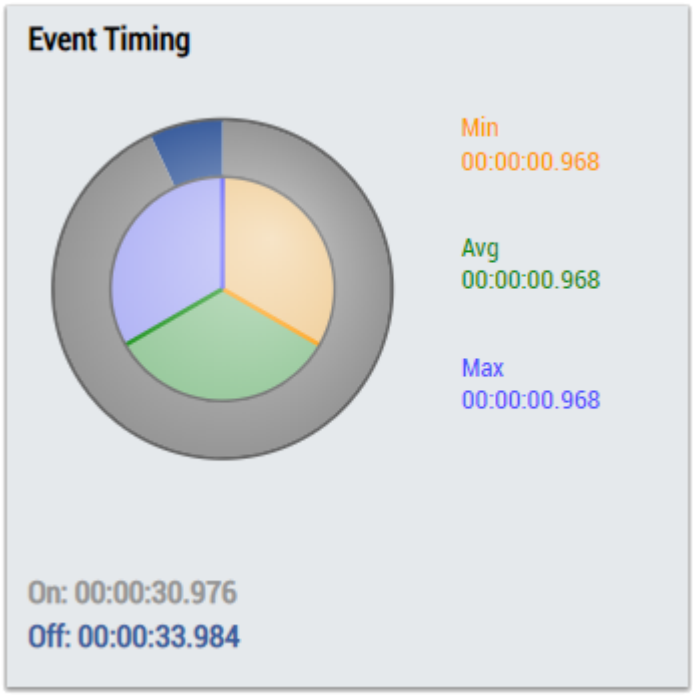
出力値

- **Is On:** オンイベントとオフ結果の間の時間内に*TRUE*を示します。それ以外は*FALSE*となります。
- **Current Interval:** 現在の区間の時間を示します。
- **Last On Interval:**直近に完了したオンイベントからオフイベントの間の時間区間。
- **Last Off Interval:** 直近に完了したオフイベントからオンイベントの間の時間区間。
- **On Min:** オンイベントからオフイベントまでの最小時間。
- **On Max:**オンイベントからオフイベントまでの最大時間。
- **On Avg :** オンイベントからオフイベントまでの平均時間。
- **On Total :** オンイベントからオフイベントまでの合計時間。
- **Off Min:**オフイベントからオンイベントまでの最小時間。
- **Off Max:**オフイベントからオンイベントまでの最大時間。
- **Off Avg :** オフイベントからオンイベントまでの平均時間。
- **Off Total :** オフイベントからオンイベントの間の合計時間。
- **Count On :** トリガーされたオンイベントの数をカウントします。

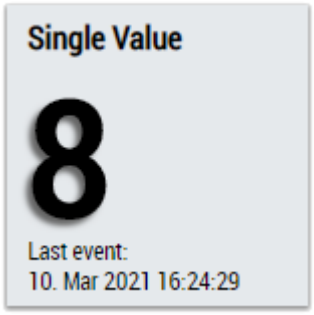
HMIコントロール

Event Timing Analysis 2Chのアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

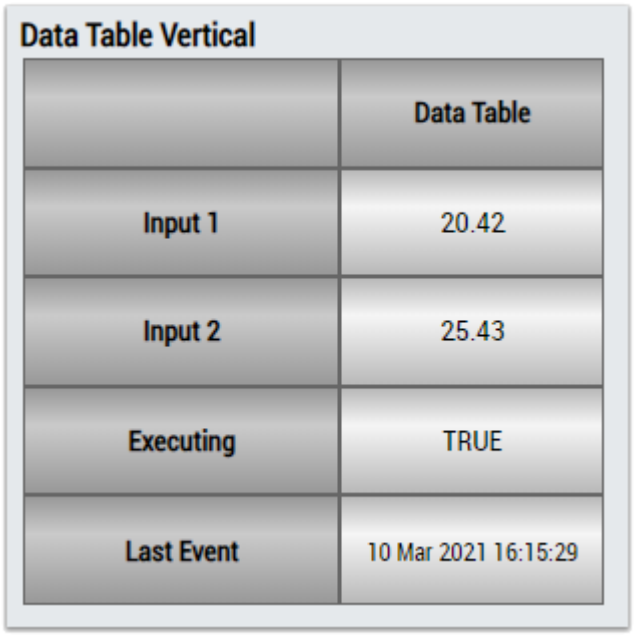
1.EventTiming コントロールは、出力値 Is On、Count On、Current Interval、On Min、On Max、On Avg、On Total、Off Min、Off Max、Off Avg、Off Total を表示します。

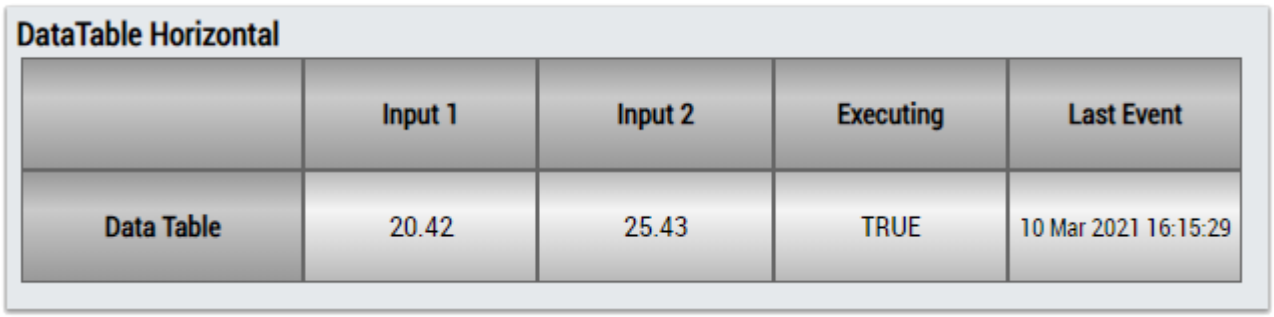


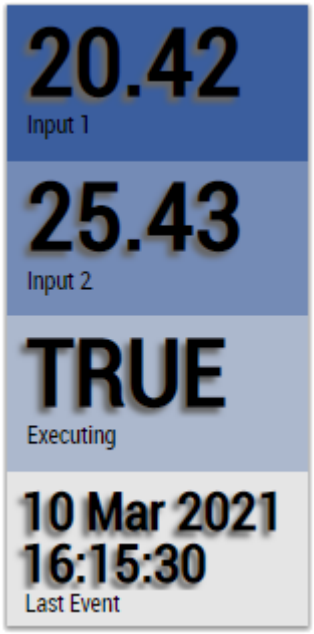
2.SingleValue コントロールは、出力値 Count On を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Is On、Count On、Current Interval、On Min、On Max、On Avg、On Total、Off Min、Off Max、Off Avg、Off Total。

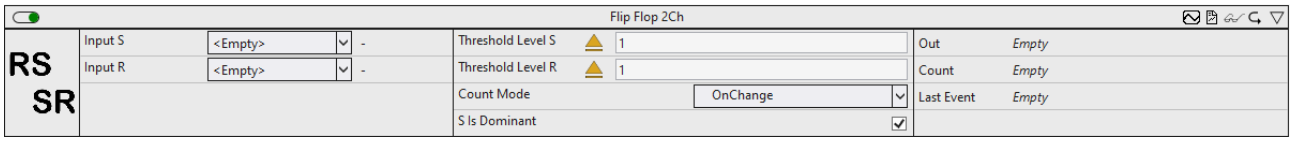






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをEvent Timing Analysis 2Chアルゴリズムにマッピングすることもできます。

6.5.1.10 Flip Flop 2Ch



*Flip Flop 2Ch*は、双安定回路（フリップフロップ）を実装します。出力値の設定（RS）またはリセット（SR）の優先動作を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が閾値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Threshold Level S:**設定する信号の閾値。
- **Threshold Level R:**リセットする信号の閾値。
- **Count Mode:**結果のカウンタモード。
  OnChange：結果がTRUEに変わるたびにカウントします。
  Cyclic：条件が*TRUE*のとき、1サイクルごとにインクリメントします。
- **S Is Dominant:**設定（RS）またはリセット（SR）の優先動作を設定できます。
- **Tolerance (オプション):**Equal / NotEqual比較時に使用される許容値。

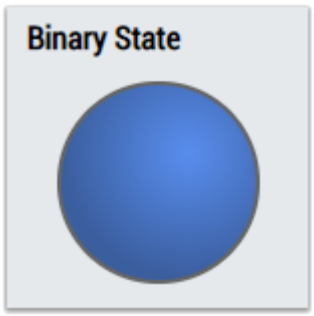
出力値

- **Out:**双安定型フリップフロップの結果。
- **Count:**出力値がTRUEになるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

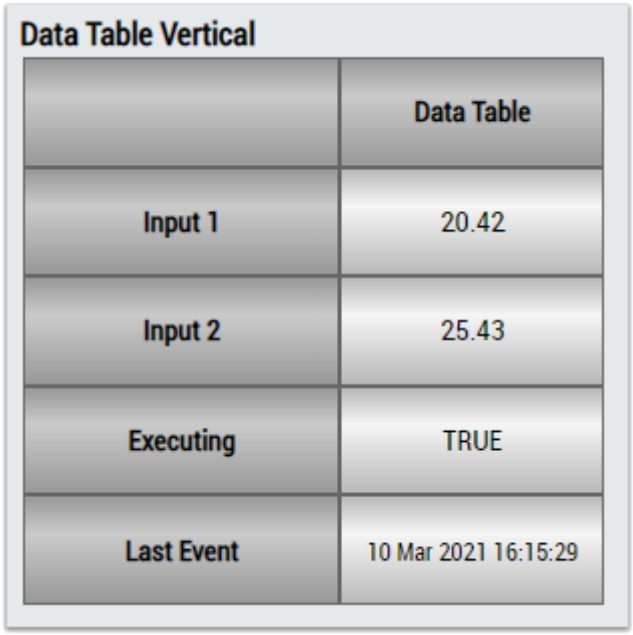
HMIコントロール

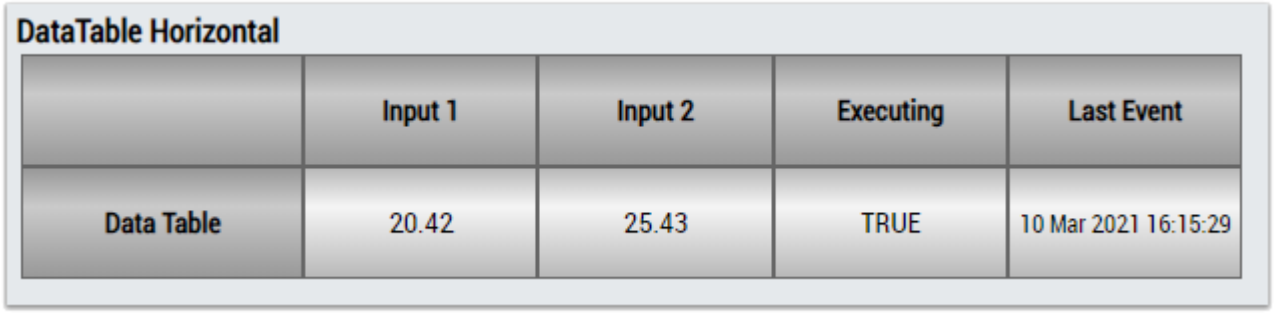
*Flip Flop 2Ch*のアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

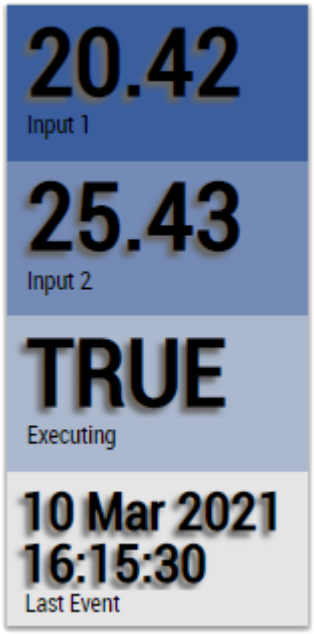
1.BinaryState コントロールは、出力値*Out* を表示します。



2.TableコントロールとMultivalueコントロールは、以下、全ての出力値を表示します：*Out*, *Count*, *Last Event*。

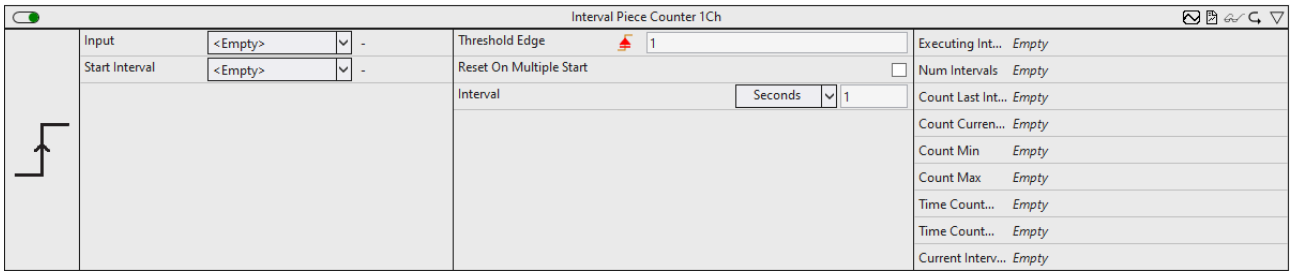






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Flip Flop 2Ch*アルゴリズムにマッピングすることもできます。

6.5.1.11 Interval Piece Counter 1Ch



*Interval Piece Counter 1Ch*は、区間スタートフラグが *TRUE* になってから開始する設定区間内で、発生したイベントの数をカウントします。入力チャンネルの信号が、設定されたエッジで特定の閾値を超過すると、イベントが発生します。時間区間が経過し、区間スタートフラグの値が再び*True*になると、計算が再開されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:** エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:** それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Reset On Multiple Start** : チェックボックスがチェックされている場合、区間スタートフラグが再び*TRUE*になったときに区間が再開されます。そうでない場合は、時間が経過すると自動的に区間が再開されます。
- **Interval:** 値を計算する時間区間。
- **Tolerance（オプション）:** Equal / NotEqual 比較時に使用される許容値。

出力値

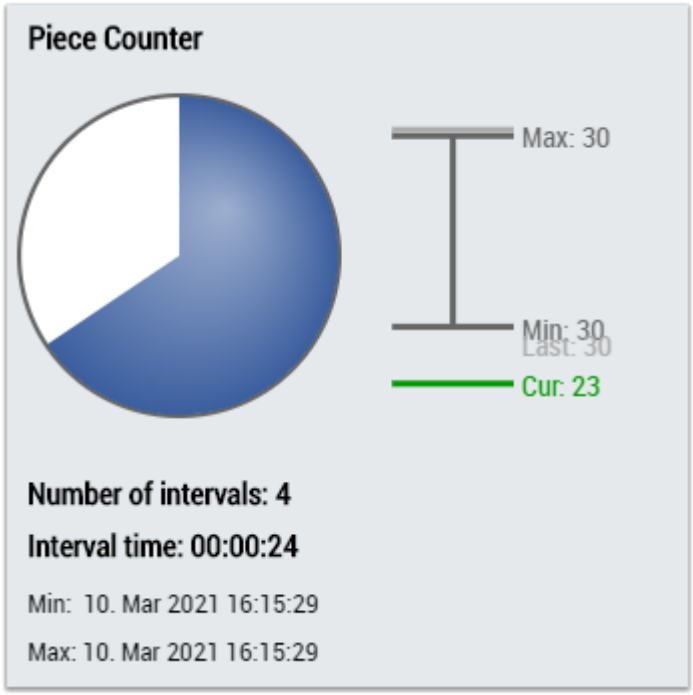
- **Executing Interval:**計算が有効で区間が実行中の場合は*True*、そうでない場合は*False*を示します。
- **Num Intervals:**区間の数を示します。

- **Count Last Interval:**最後の区間でトリガーされたイベントの数を示します。
- **Count Current Interval:**現在の区間でトリガーされたイベントの数を示します。現在、計算が有効でない場合は、最後の区間でトリガーされたイベントの数を示します。
- **Count Min:**区間内でトリガーされたイベントの最小数を示します。
- **Count Max :** 区間内でトリガーされたイベントの最大数を示します。
- **Time Count Min :** 最小の時間値を示します→ イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Time Count Max:**最大の時間値を示します→ イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Current Interval Time:**現在の区間の時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

HMIコントロール

Interval Piece Counter 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.PieceCounter コントロールは、出力値 Num Intervals、Count Last Interval、Count Current Interval、Count Min、Count Max、Time Count Min、Time Count Max、Time Current Interval を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Num Intervals, Count Last Interval, Count Current Interval, Count Min, Count Max, Time Count Min, Time Count Max, Time Current Interval。

Data Table Vertical

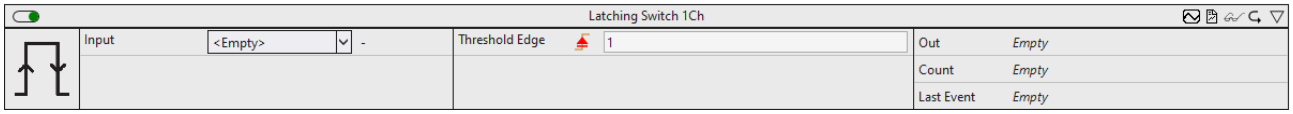
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを Interval Piece Counter 1Chアルゴリズムにマッピングすることもできます。

6.5.1.12 Latching Switch 1Ch



*Latching Switch 1Ch*は、仮想的なインパルス（パルス）スイッチとして動作します。入力でエッジが検出されるたびに、出力はTRUE と FALSE を交互に繰り返します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**入力の立ち上がりエッジに反応するか、立ち下がりエッジに反応するかを指定します。
- **Threshold Edge:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

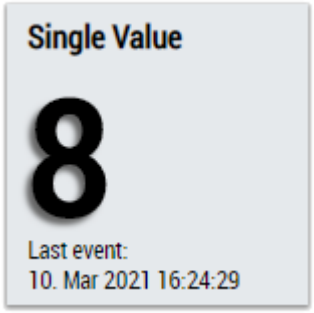
出力値

- **Out:**仮想インパルススイッチの結果。
- **Count:** 出力値が変化するとインクリメントされます。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

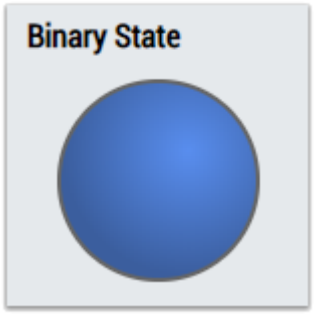
HMIコントロール

*Latching Switch 1Ch*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.SingleValue コントロールは、出力値 Count と Last Event を表示します。



2.BinaryState コントロールは、出力値*Out* を表示します。



3.TableコントロールとMultivalueコントロールは、全ての出力値を表示します。

Data Table Vertical

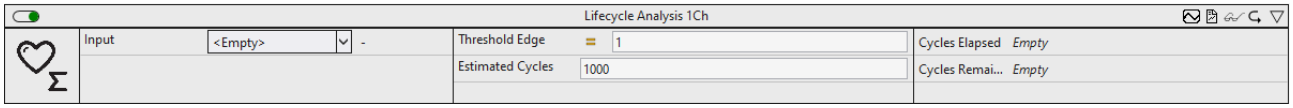
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをマッピングすることもできます。

6.5.1.13 Lifecycle Analysis 1Ch



*Lifecycle Analysis 1Ch*は、デバイスの経過サイクルと推定残存サイクルを計算します。入力チャンネルの信号が設定されたエッジで閾値を超過すると、経過サイクルが増加し、残存サイクルが減少します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超えるとイベントがトリガーされます。
- **Estimated Cycles:**各デバイスの耐用年数中の推定サイクル。
- **Tolerance（オプション）:** Equal / NotEqual 比較時に使用される許容値。

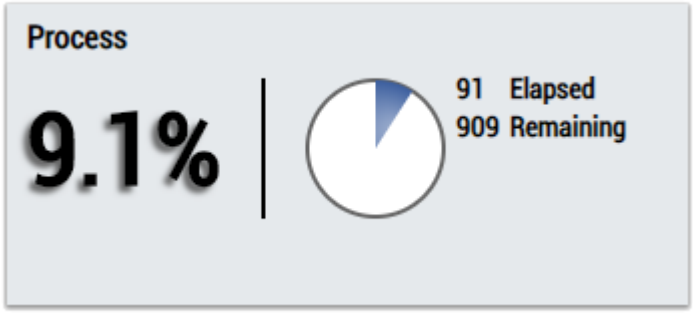
出力値

- **Elapsed Cycles:**既に経過したサイクル数をカウントします。
- **Remaining Cycles:**デバイスの残存サイクルを、推定サイクルと経過サイクルの差として表示します。

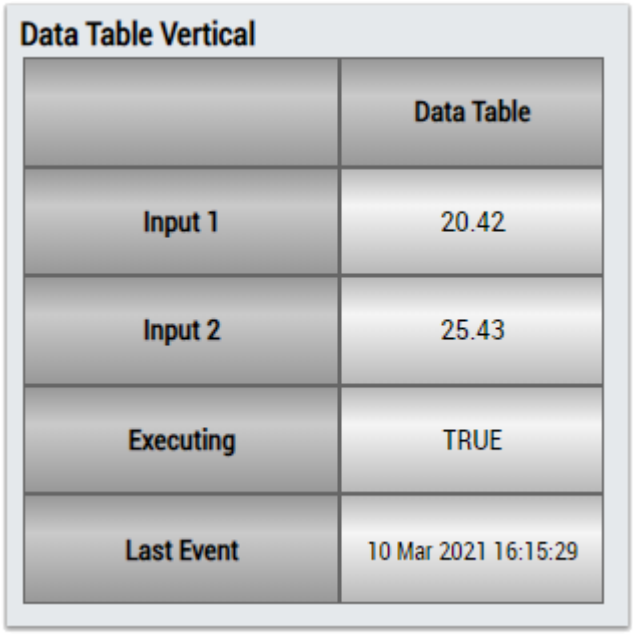
HMIコントロール

Lifecycle Analysis 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.Processコントロールは、出力値 Elapsed Cycles と Remaining Cycles を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Elapsed Cycles, Remaining Cycles。

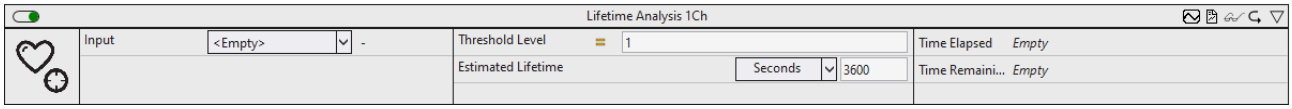


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをLifecycle Analysis 1Chアルゴリズムにマッピングすることもできます。

6.5.1.14 Lifetime Analysis 1Ch



*Lifetime Analysis 1Ch*は、デバイスの経過済み耐用年数と推定残存耐用年数を計算します。入力値が設定条件を満たす場合、耐用年数は短くなります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が閾値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Threshold:**信号の閾値。
- **Estimated Lifetime:**各デバイスの推定耐用年数。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

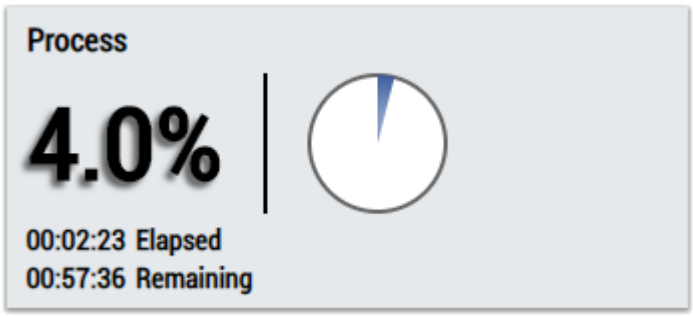
出力値

- **Elapsed Lifetime:**既に経過した耐用年数を表示します。
- **Remaining Lifetime:**デバイスの推定耐用年数と経過済み耐用年数の差として、残存耐用年数を表示します。

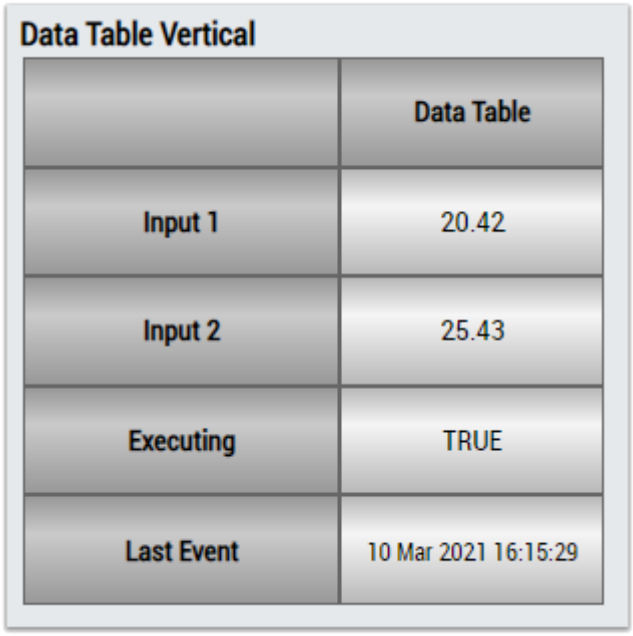
HMIコントロール

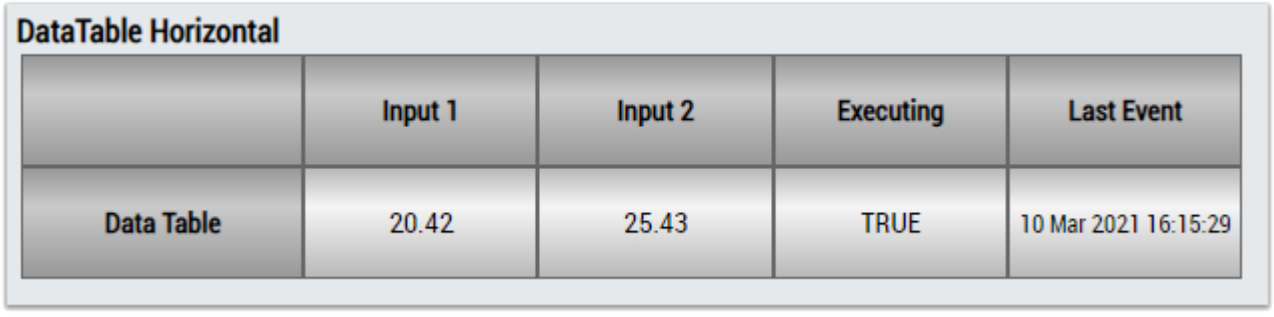
Lifetime Analysis 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

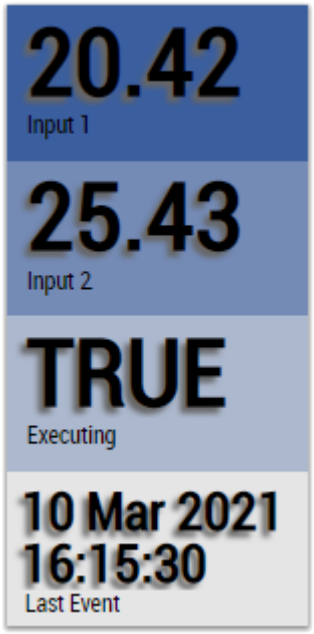
1.Processコントロールは、出力値 Elapsed Lifetime と Remaining Lifetime を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Elapsed Lifetime, Remaining Lifetime。

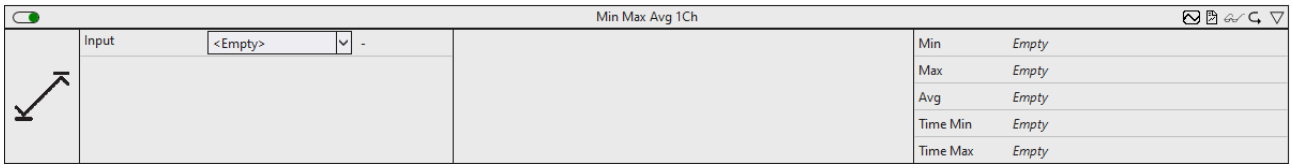






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをLifetime Analysis 1Chアルゴリズムにマッピングすることもできます。

6.5.1.15 Min Max Avg 1Ch



*Min Max Avg 1Ch*は、解析開始から現在までの入力値の最小値、最大値、平均値を計算します。さらに、最小時と最大時の時間値が示されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

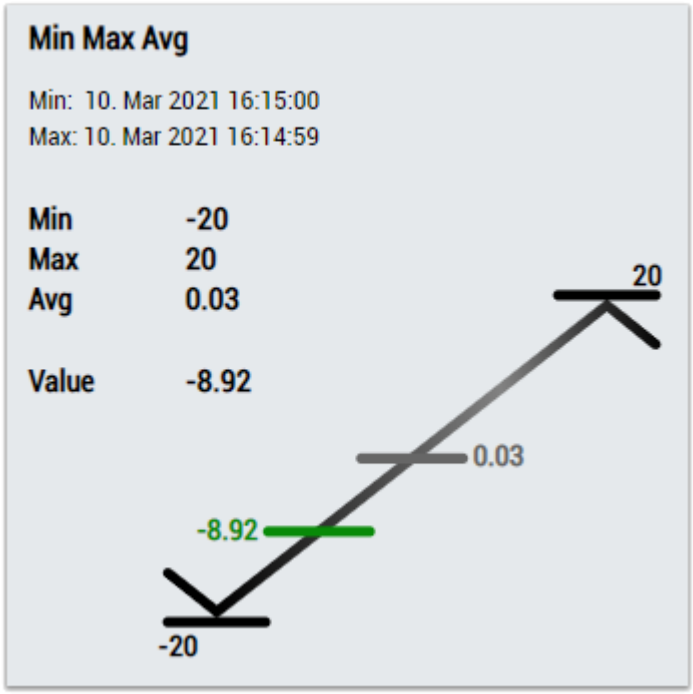
出力値

- **Min:**入力値の最小値を示します。
- **Max :** 入力値の最大値を示します。
- **Avg :** 入力値の平均値を示します。
- **Time Min:**最小時の時間値を示しています → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Time Max:**最大時の時間値を示しています → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

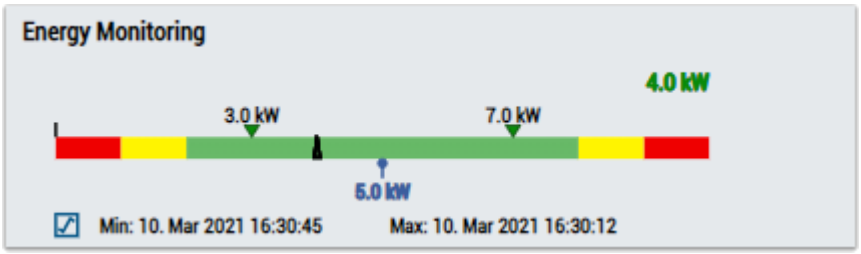
HMIコントロール

Min Max Avg 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

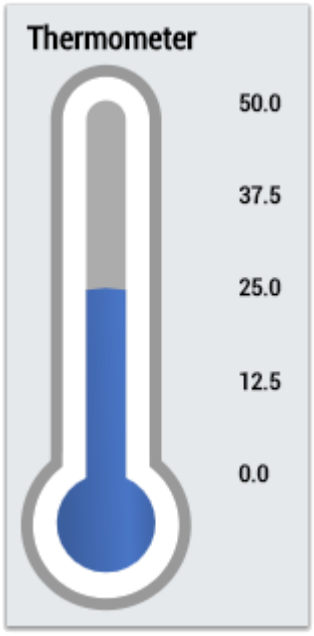
1.MinMaxAvg コントロールは、データの入力値と、出力値 Min、Max、Avg、Time Min、Time Max を表示します。



2.EnergyMonitoring コントロールは、以下、全ての出力値を表示します：Min, Max, Avg, Time Min, Time Max and the current input value。



3.ThermometerコントロールはAvg temperature（平均温度）を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Min, Max, Avg, Time Min, Time Max。

Data Table Vertical

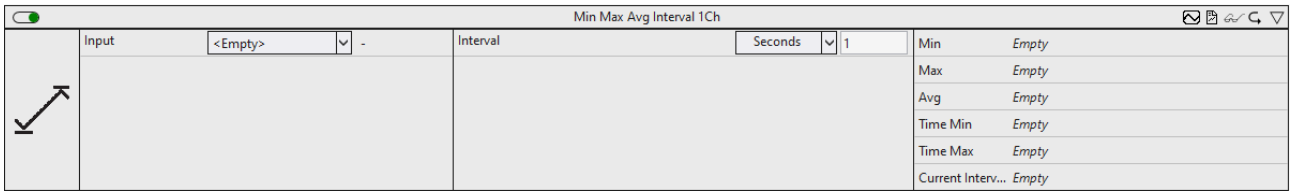
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをMin Max Avg 1Chアルゴリズムにマッピングすることもできます。

6.5.1.16 Min Max Avg Interval 1Ch



*Min Max Avg Interval 1Ch*は、設定した時間周期での、入力値の最小値、最大値、平均値を計算します。さらに、最小時と最大時の時間値が示されます。全て値は直前の相対区間のものであり、区間が終わって初めて更新されることに注意してください。時間区間が経過すると計算が再開されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定した時間内の信号のうち、特定のサンプルのみを出力値の計算に含めたい場合は、オプションで、Boolean入力*Add Sample* を使って実装できます。全ての入力値は、入力*Add Sample* の値が*TRUE* であるものとみなされます。

設定オプション

- **Interval:** 値を計算する時間区間。

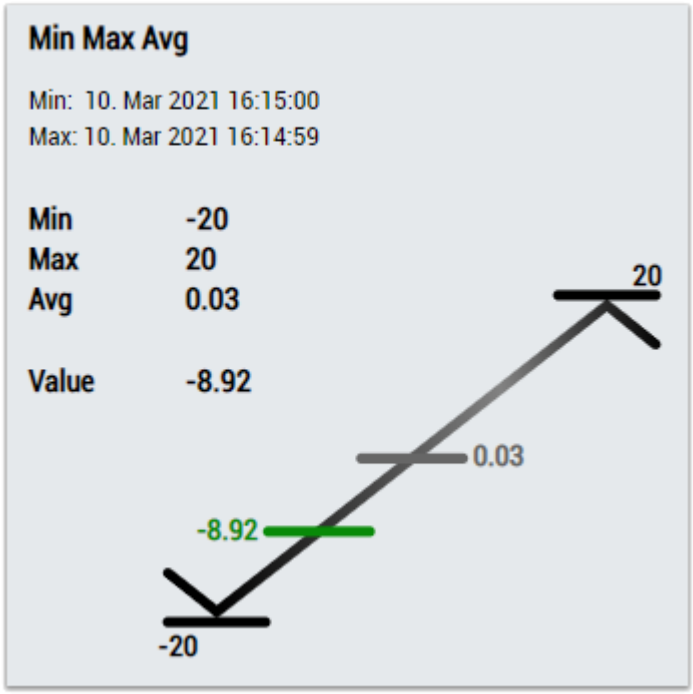
出力値

- **Min:**直近の時間区間における入力値の最小値を示します。
- **Max :** 直近の時間区間における入力値の最大値を示します。
- **Avg :** 直近の時間区間における入力値の平均値を示します。
- **Time Min:**直近の時間区間における最小時の時間値を示します→ イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Time Max:**直近の時間区間における最大時の時間値を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Current Interval Time:**現在の区間の経過時間を示します。

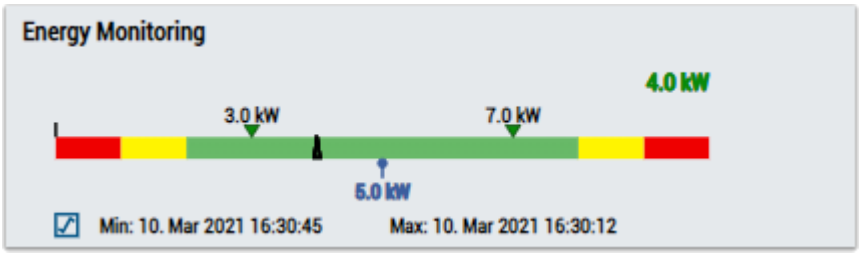
HMIコントロール

Min Max Avg Interval 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

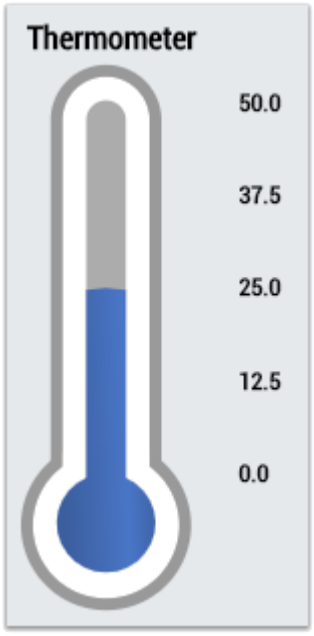
1.MinMaxAvg コントロールは、データの入力値と、出力値 Min、Max、Avg、Time Min、Time Max を表示します。



2.EnergyMonitoring コントロールは、出力値Min、Max、Avg、Time Min、Time Max、および現在の入力値を表示します。



3.ThermometerコントロールはAvg temperature（平均温度）を表示します。



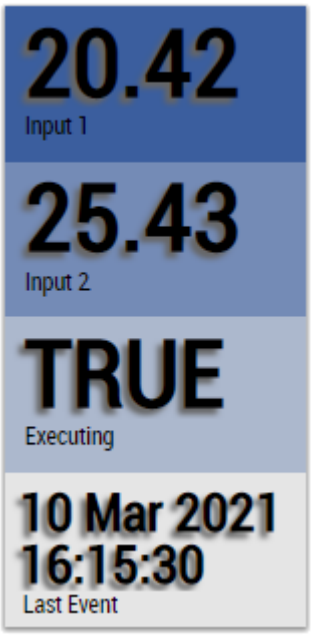
4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Min, Max, Avg, Time Min, Time Max, Current Interval Time。

Data Table Vertical

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

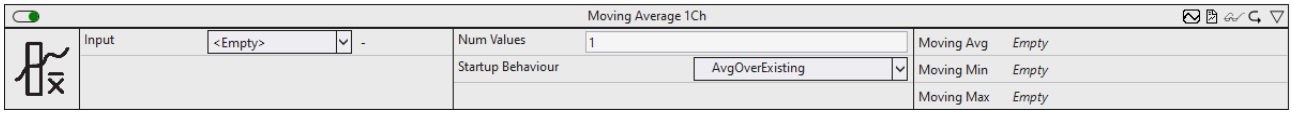
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをMin Max Avg Interval 1Chアルゴリズムにマッピングすることもできます。

6.5.1.17 Moving Average 1Ch



*Moving Average 1Ch*は、指定した長さの区間で、直近の入力値の移動平均、最小値、および最大値を計算します。さらに、最小時と最大時の時間値が示されます。移動平均の計算は、パラメータ設定*Num Values*と*Startup Behaviour*に依存します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Values:** 移動平均、最小値、最大値の計算に含まれる値の数。
- **Startup Behaviour:** *Num Values*分の入力値がそろう前の解析開始時の計算動作。
  ZeroPadding：欠損値をゼロで埋めます。
  UseFirstValue：値が*Num Values*と等しくなるまで、最初の値が使用されます。
  WaitUntilFilled：最初の結果は、値の数が*Num Values*と等しくなると計算されます。
  AvgOverExisting：値の数が*Num Values*と等しくなるまで、既存の値で平均を計算します。

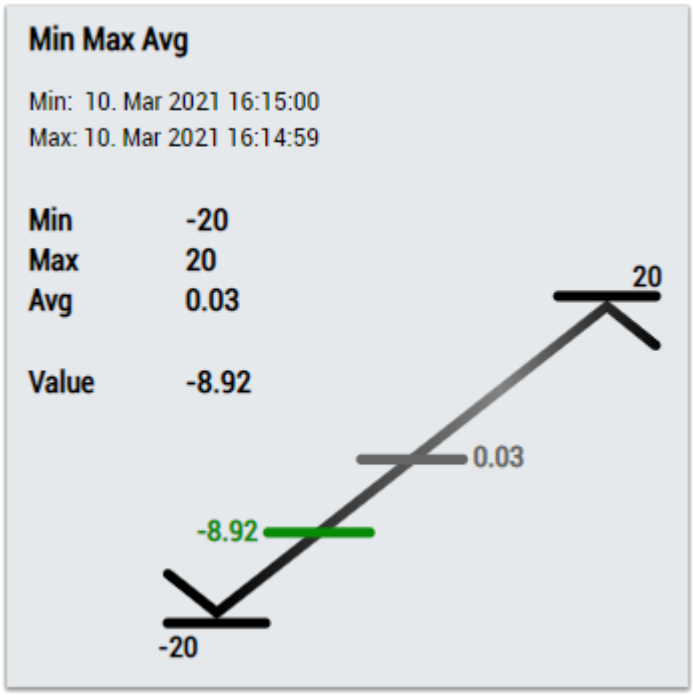
出力値

- **Moving Avg:**現在の平均値を表示します。
- **Moving Min:** 直近のn個の入力値の最小値を示します。
- **Moving Max:** 直近のn個の入力値の最大値を示します。

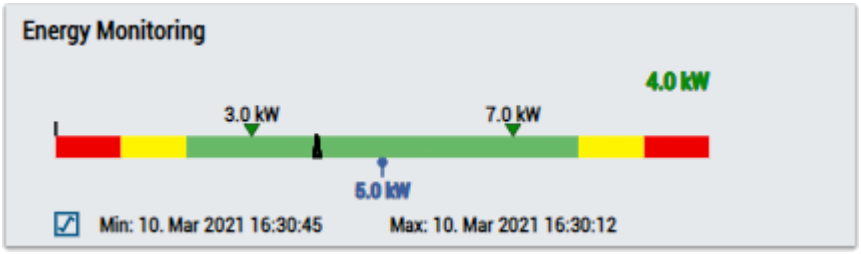
HMIコントロール

Moving Average 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

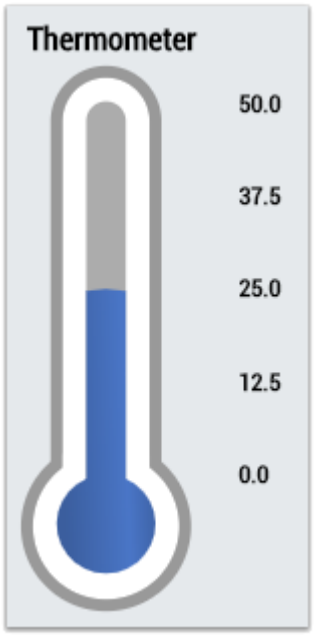
1.MinMaxAvgコントロールは、出力値 Moving Min, Moving Max and Moving Avg とデータの入力値を表示します。



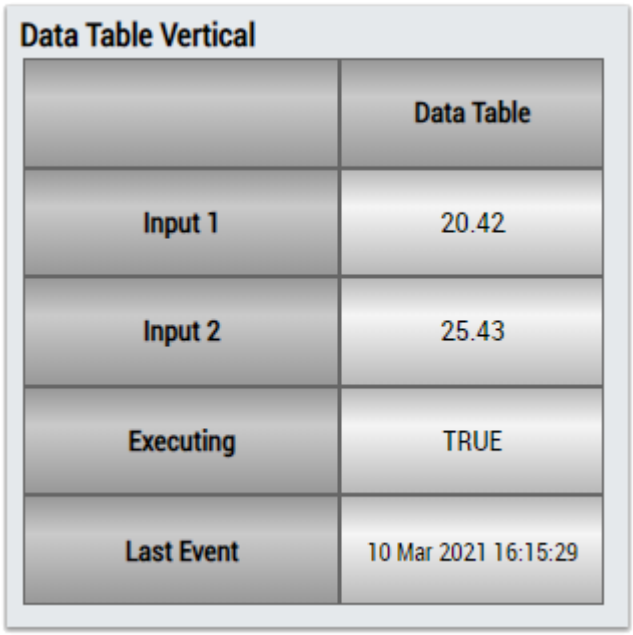
2.EnergyMonitoringコントロールは、出力値Moving Min、Moving Max、Moving Avgと、データの入力値を表示します。



3.ThermometerコントロールはMoving Avg temperature（移動平均温度）を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Moving Min, Moving Max, Moving Avg.。

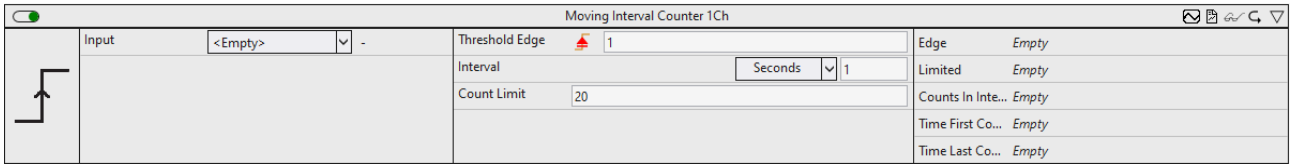


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをMoving Average 1Chアルゴリズムにマッピングすることもできます。

6.5.1.18 Moving Interval Counter 1Ch



*Moving Interval Counter 1Ch*は、設定した区間内で発生したイベントの数をカウントします。入力チャンネルの信号が、設定されたエッジで特定の閾値を超過すると、イベントが発生します。時間区間が経過すると計算が再開されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:** エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:** それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Interval:** 値を計算する時間区間。
- **Count Limit:**区間内でカウントできるエッジの数を制限します。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

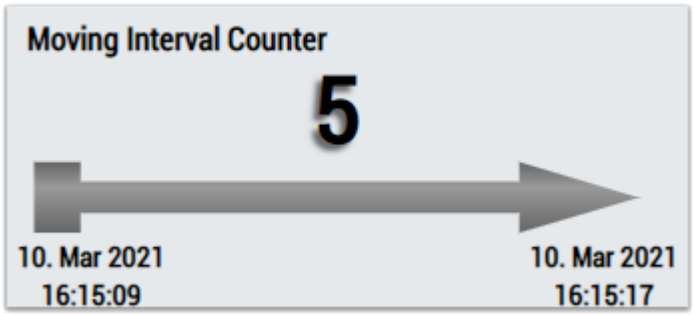
出力値

- **Edge:**イベントがトリガーされた時に*TRUE* を示し、そうでなければ*FALSE* を示します。
- **Limited:** 現在の区間のエッジ数が、設定したCount Limitを超えた場合にTRUEを示します。
- **Counts in Interval:**現在の区間でトリガーされたイベントの数を示します。
- **Time First Count:**最初にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Time Last Count:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

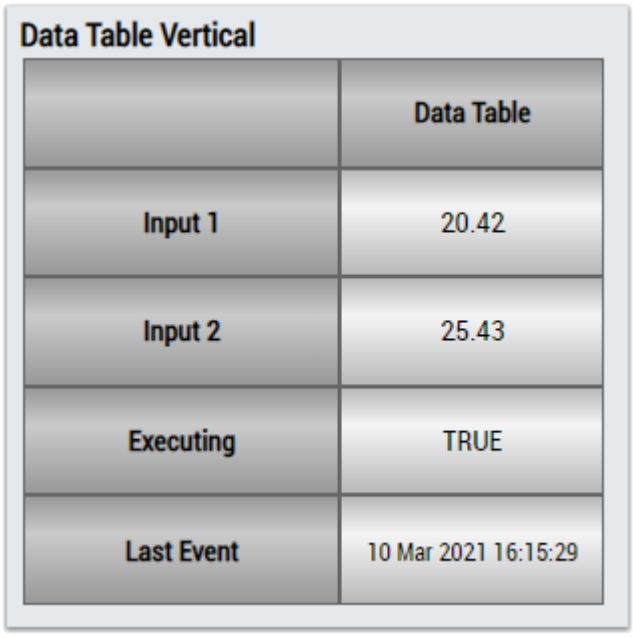
HMIコントロール

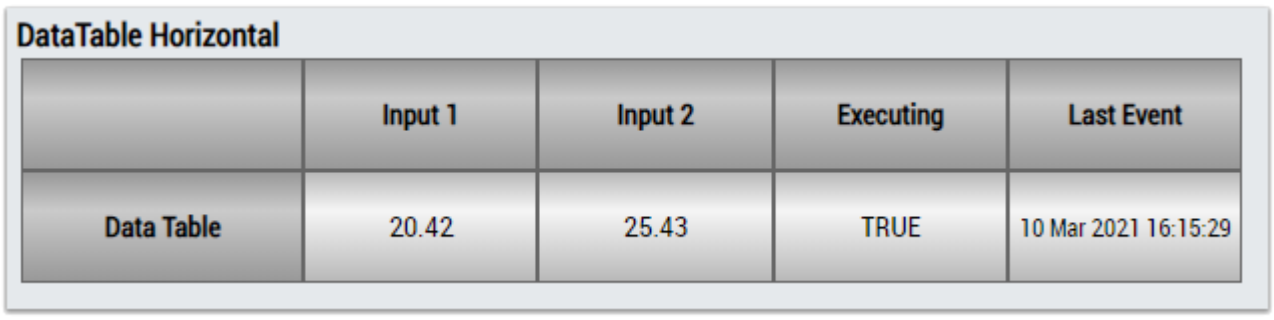
Moving Interval Counter 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

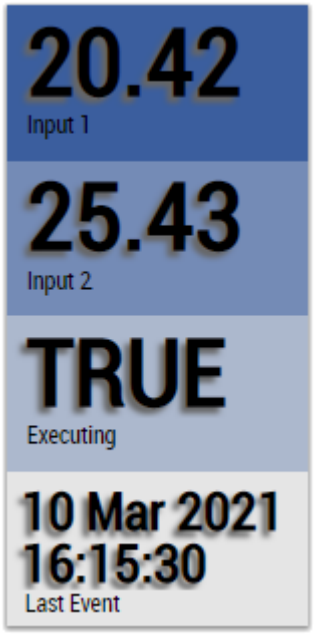
1.MovingIntervalCounterコントロールは、出力値 Counts in Interval, Time First Count and Time Last Countを表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Edge, Counts in Interval, Time First Count, Time Last Count, Limited。

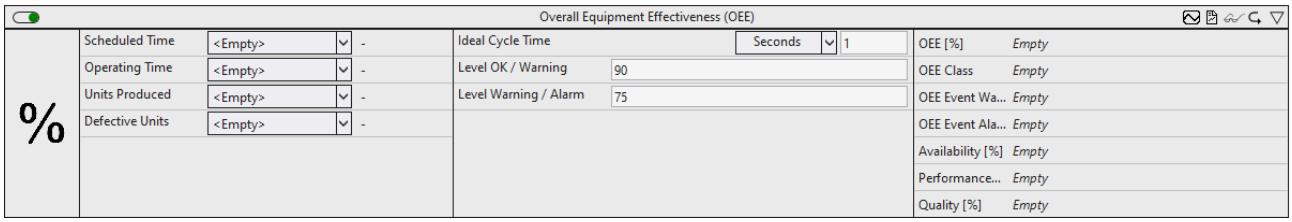






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをMoving Interval Counter 1Chアルゴリズムにマッピングすることもできます。

6.5.1.19 Overall Equipment Effectiveness (設備総合効率、OEE)



*Overall Equipment Effectiveness (*設備総合効率、*OEE*）は、製造工程の現状と、設備が持つ最大限の可能性とを比較するために重要な数値を算出します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Ideal Cycle Time:**1台の生産にかかる理想的なサイクルタイム（基準サイクルタイム）。
- **Level Ok / Warning:** 設備全体の効率が設定した閾値を超える場合は*OK* に分類されます。設備全体の効率が設定した閾値を下回る場合は、*Warning* として分類されます。
- **Level Warning / Alarm:** 設備全体の効率が設定した閾値を超える場合は、*Warning*として分類されます。設備全体の効率が設定した閾値を下回る場合は、*Alarm* として分類されます。

入力値

- **Scheduled Time:**稼働時間はカレンダー時間から非稼働予定時間を差し引いたものです。
- **Operating Time:**実働時間は稼働時間からダウンタイムを差し引いたものです。
- **Units Produced:**不良品を含む生産個数。
- **Defective Units:**不良品の数。

出力値

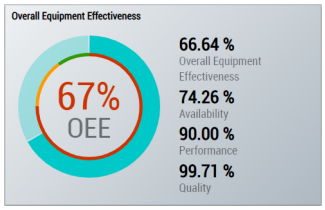
- **OEE :** 設備総合効率（%）。時間稼働率、性能稼働率、良品率を掛け合わせることで算出されます。
- **OEE Class:**設備総合効率の分類。
- **OEE Event Warning:**最後に警告イベントがトリガーされた時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

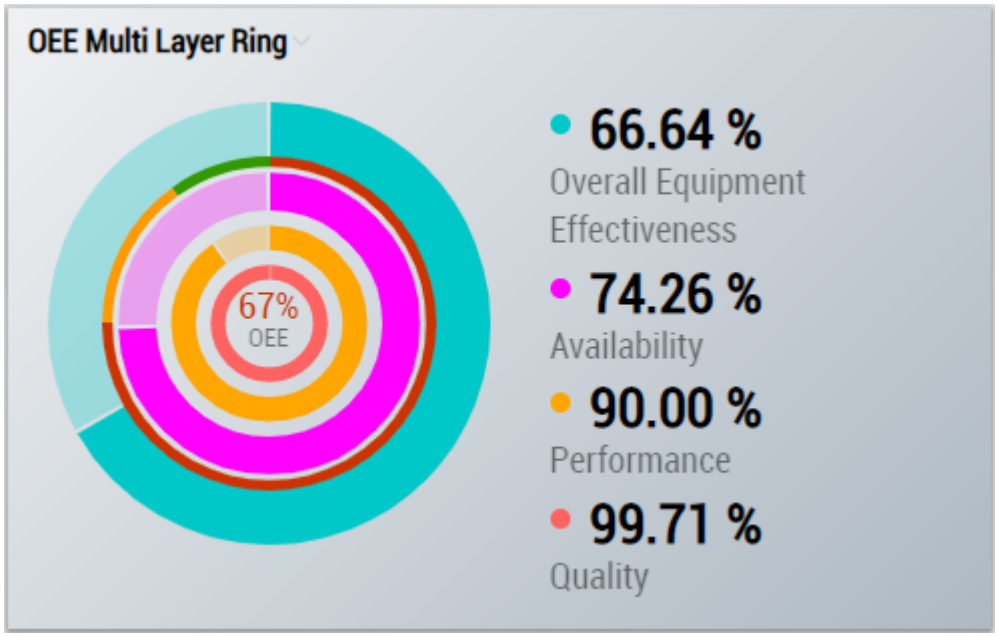
- **OEE Event Alarm:**最後にトリガーされたアラームイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすると、トリガーイベントとして表示できます。
- **Availability:**時間稼働率（%）。実働時間と稼働時間の比率から算出されます。
- **Performance:**性能稼働率（%）。これは、実際の生産数と、理想的なケース（基準サイクルタイム）での生産数の比率から算出されます。
- **Quality:**良品率（%）。これは、生産数に対して良品となった製品の比率から算出されます。

HMIコントロール

*Overall Equipment Effectiveness (OEE)*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.OEEコントロールには3つのデザインがあります。全ての出力値、およびパラメータ *Level OK / Warning* と *Level Warning / Alarm*を表示します。





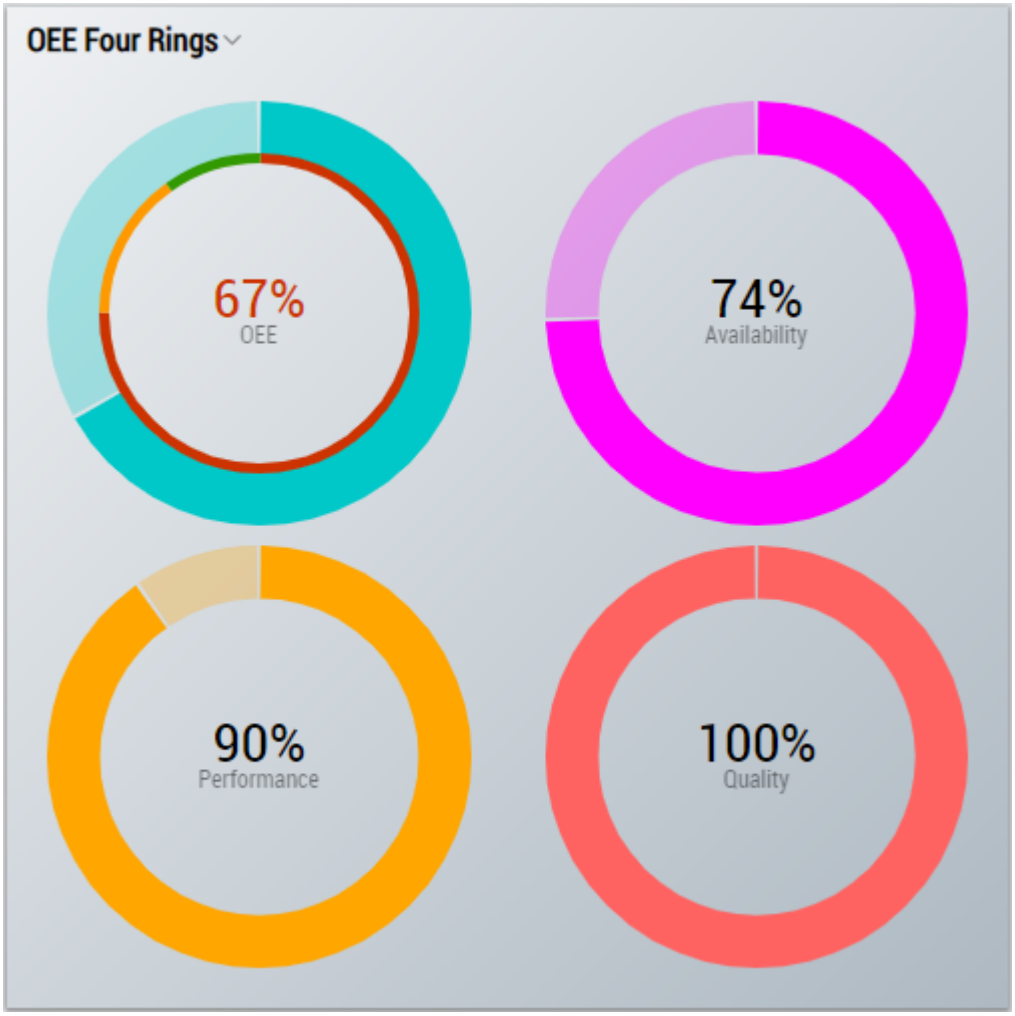


図 1:

2.TableコントロールとMultivalueコントロールは、出力値を表示します：*OEE, Availability, Performance* および *Quality*。

Data Table Vertical

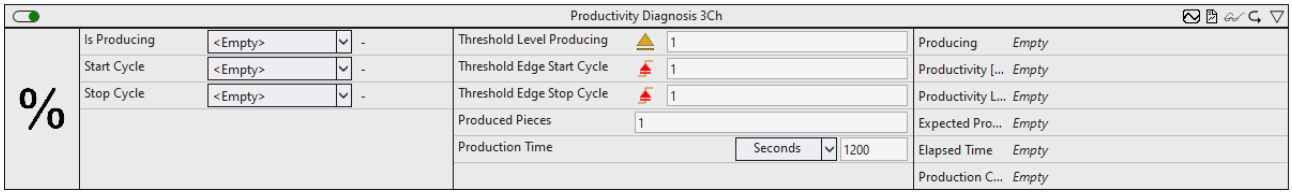
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

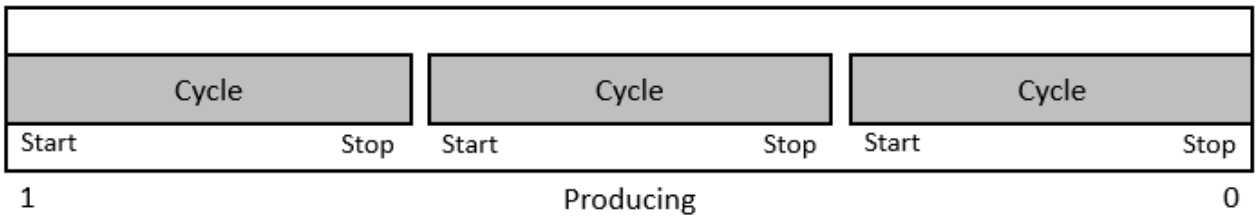
	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29

また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールを*Overall Equipment Effectiveness (OEE)*アルゴリズムにマッピングすることもできます。

6.5.1.20 Productivity Diagnosis 3Ch



*Productivity Diagnosis 3Ch*アルゴリズムは、生産区間中のプロセスの生産性を計算します。下図は、生産工程と各生産サイクルの関係を模式的に示したものです。



生産区間は、*Is Producing* を入力することで開始・停止できます。生産区間の実行中、生産サイクルがカウントされます。 1 つの生産サイクルは1個の製品の生産に相当します。生産サイクルは、*Start Cycle*のエッジで開始し、*Stop Cycle*のエッジで停止します。生産区間全体の生産性（*Productivity*）は、信号*Is Producing*が *Threshold Level Producing*の条件を満たさなくなった時点で、生産区間を停止した後に計算されます。完了した生産サイクル、つまり全ての完成品が考慮されます。生産性は、時間当たりに実際に生産された個数と、一定時間内に生産されるべき個数の目標値との比率として計算されます。出力*Productivity Last Cycle*は、設定された1個あたりの生産時間に対して、最後の生産サイクルに要した時間から計算されます。サイクル間の休止時間は考慮されません。出力*Expected Productivity*は、生産区間中

の総生産性を推定します。このため、過去の生産時間が、生産予定個数に対する総生産性の予測に用いられます。アルゴリズムは、設定された区間内（*Production Time*）の*Produced Pieces*の目標値で設定することができます（例：30秒で1個、1時間で50個）。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the Edge:**アルゴリズムが立ち上がりエッジと立ち下がりエッジのどちらに応答すべきかを指定します。閾値ごとに個別に設定可能。
- **Threshold Level Producing** : それぞれのエッジにおける*Input Producing*の閾値。*Production Time Interval*は、信号がこの閾値を超過した時点で開始されます。
- **Threshold Edge Start Cycle:**それぞれのエッジにおける*Input Start Cycle*の閾値。信号がこの閾値を超過すると、生産サイクルが開始されます。
- **Threshold Edge Stop Cycle:**それぞれのエッジにおける*Input Stop Cycle*の閾値。信号がこの閾値を超えると、生産サイクルは停止します。
- **Produced Pieces:** 設定された時間区間(*Production Time*)内での生産数の目標値。
- **Production Time:** 生産時間の区間。日、時間、分、秒単位で設定可能。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

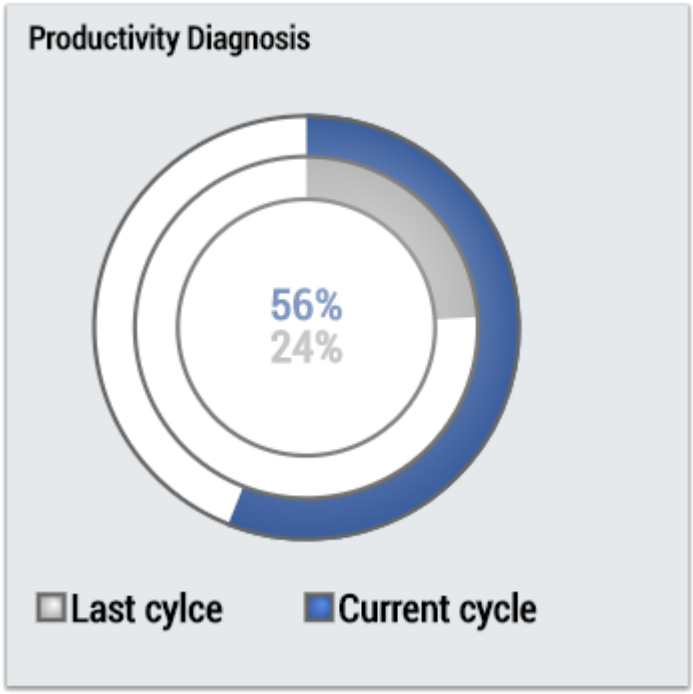
出力値

- **Producing:** 生産区間が有効かどうかを示します。
- **Productivity:**生産区間全体の生産性。
- **Productivity Last Cycle:** 最終生産サイクルの生産性（%）。
- **Expected Productivity:** 生産区間の生産性を予測します。（%）
- **Elapsed Time :** 生産区間の開始から経過した時間。
- **Production Cycles:** 現在の生産区間で完了した生産サイクル数。

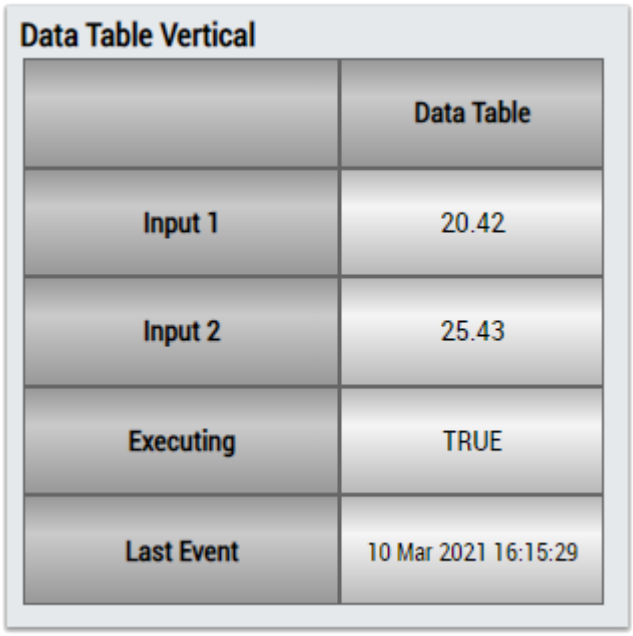
HMIコントロール

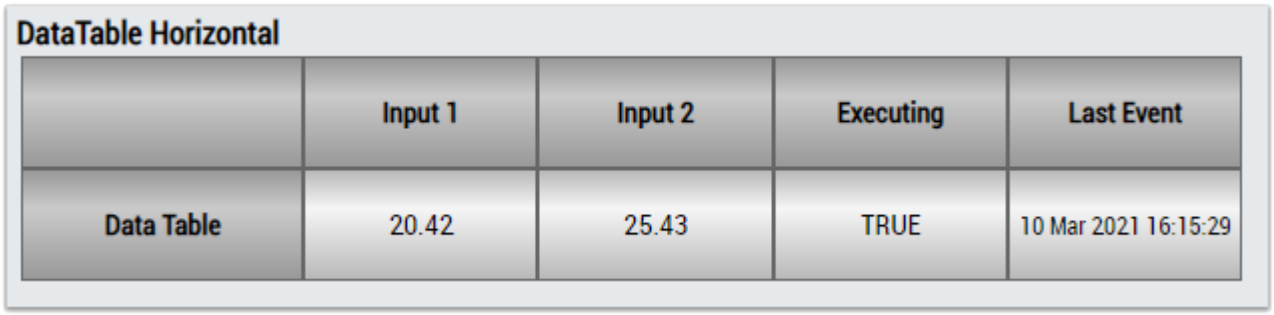
Productivity Diagnosis 3Chでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

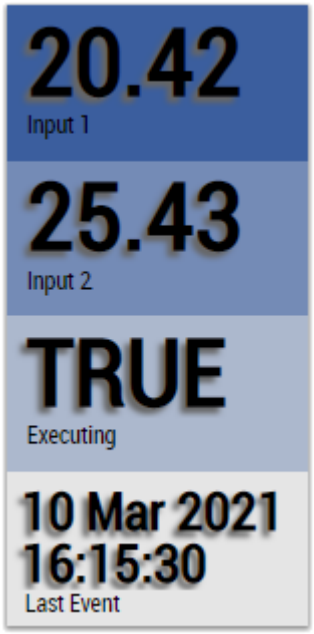
1.ProductivityDiagnosis コントロールは、出力値 Productivity と Productivity Last Cycle を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：New Result, Producing, Cycle Finished, Productivity, Productivity Last Cycle。

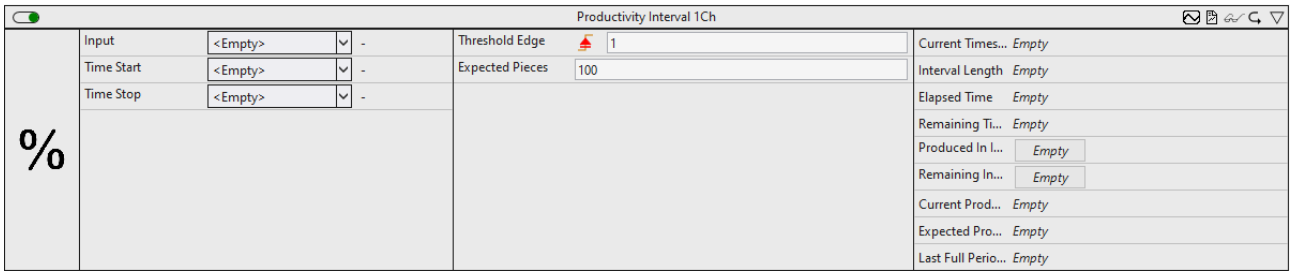






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを生産性診断3Chアルゴリズムにマッピングすることもできます。

6.5.1.21 Productivity Interval 1Ch



アルゴリズム*Productivity Interval 1Ch*は、一定の区間内でプロセスの生産性を計算します。区間は入力*tTimeStart*と *tTimeStop*で定義できます。実行中の生産数が考慮されます。入力にエッジが加わると、生成された要素がカウントされます。現在の区間の推定生産性と、最後に完了した区間の生産性が出力値として提供されます。このアルゴリズムは、所定の区間内における生産数の目標値を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the Edge:**生産数カウンタが立ち上がりエッジと立ち下がりエッジのどちらに応答するかを指定します。
- **Threshold Edge** : 生産数がカウントされる入力 の閾値。
- **Expected Pieces:** 設定した期間内における生産数の指定。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

入力値

- **Input:** 生産数をカウントするための入力値。
- **Time Start:**指定区間の開始時間。プロセスデータから区間を指定できるようにします。1nsの分解能で時間を指定します。
- **Time Stop:** 指定区間の終了時間。プロセスデータから区間を指定できるようにします。1nsの分解能で時間を指定します。

出力値

- **Within Interval:** 現在の時間が区間内にあるかどうかを示します。
- **Current Timestamp:**現在のタイムスタンプ。
- **Interval Length:** 区間の長さ。
- **Elapsed Time:** 区間内の経過時間。
- **Remaining Time:** 区間内の残り時間。
- **Produced In Interval:**区間内の生産済み数。
- **Remaining In Interval:** 区間内の残数。
- **Current Productivity:**区間内の現在の生産性（%）。区間の長さ、経過済みの時間、生産予定数と既に生産済みの数を考慮します。出力はパーセント値で表示されます。
- **Expected Productivity:** 区間で期待される生産性（%）。残り時間で生産できる生産数を見積もるため、最後の生産品の生産時間が使用されます。
- **Last Full Period Productivity:** 最後に完了した区間の生産性（%）。これは、区間内の処理が完了している場合にのみ計算されます。

HMIコントロール

Productivity Interval 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.ProductivityInterval コントロールは、以下の出力値を表示します：time elapsed, time remaining, number of pieces produced in the interval, number of pieces remaining in the interval, productivity。

2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：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

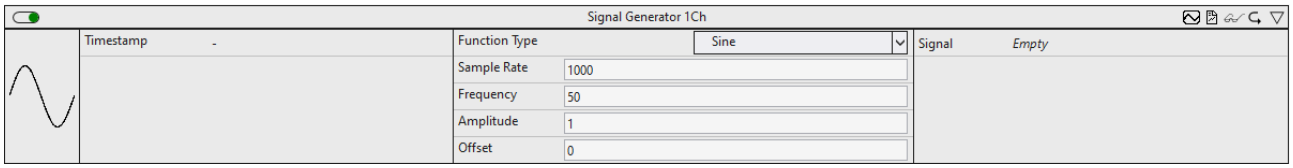
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをProductivity Interval 1Chアルゴリズムにマッピングすることもできます。

6.5.1.22 Signal Generator 1Ch



*Signal Generator 1Ch*は、様々な信号曲線を生成するために使用できます。信号の種類、周波数、振幅、オフセットは個別に設定できます。参照値としてタイムスタンプが必要です。アルゴリズムが動作するために時間コンテキストが必要なためです。この参照用のタイムスタンプは、設定に別のアルゴリズムが含まれている場合は、自動的に設定されます。したがって、Signal Generator 1Chを単独で使用することはできません。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Amplitude:**信号振幅の設定。
- **Frequency:**生成される信号の周波数。
- **Function Type:**生成される信号の関数タイプ。
  Const：定数
  *Rectangle:*矩形波（長方形波）
  *Sawtooth:*ノコギリ波
  *Sine:*正弦波
  *Triangle:*三角波
- **Offset :** 生成される信号の定数オフセット。
- **Sample Rate :** 解析対象システムのサンプルレート。

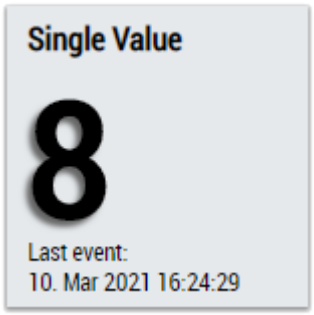
出力値

- **Signal :** 生成された信号を出力します。

HMIコントロール

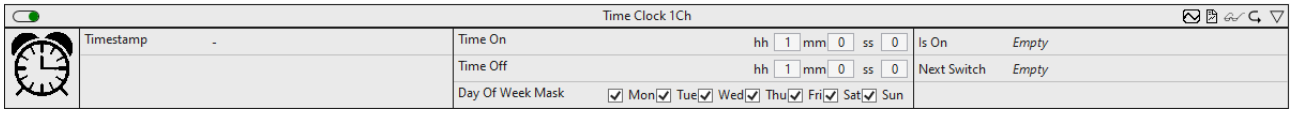
Signal Generator 1Chでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.SingleValueコントロールは、信号出力値を表示します。



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをSignal Generator 1Chアルゴリズムにマッピングすることもできます。

6.5.1.23 Time Clock 1Ch



*Time Clock 1Ch*は、スイッチのオン時間とオフ時間、および動作させる曜日を設定できるタイマー機能を提供します。参照値としてタイムスタンプが必要です。アルゴリズムが動作するために時間コンテキストが必要なためです。この参照用のタイムスタンプは、設定に別のアルゴリズムが含まれている場合は、自動的に設定されます。したがって、Time Clock 1Chを単独で使用することはできません。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Time On :** スイッチオン時間。
- **Time Off :** スイッチオフ時間。
- **Day of Week Mask:** タイマーを有効にする曜日。

出力値

- **Is On:**タイマーがオンの場合は*TRUE*を表示し、そうでない場合は*FALSE*を表示します。
- **Next Switch:** 次のタイマーまでの残り時間を表示します。

HMIコントロール

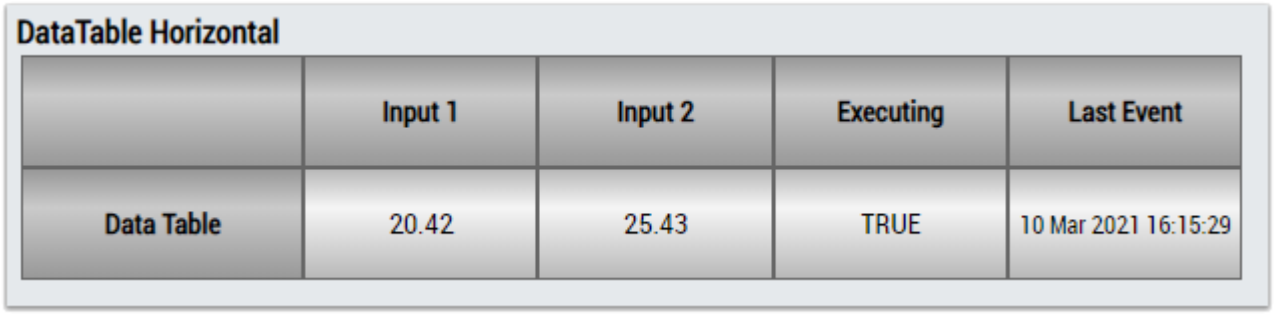
Timer Clock 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

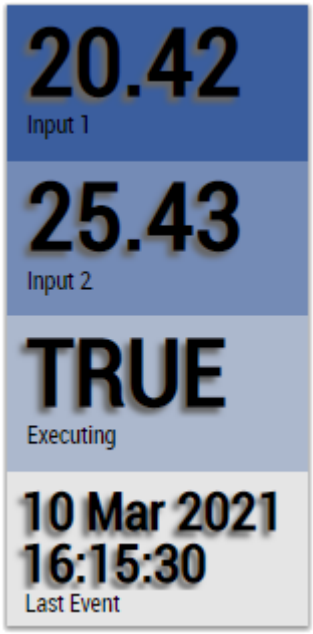
1.Timerコントロールは出力値Next Switchを表示します。

2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Output, Next Switch。

Data Table Vertical

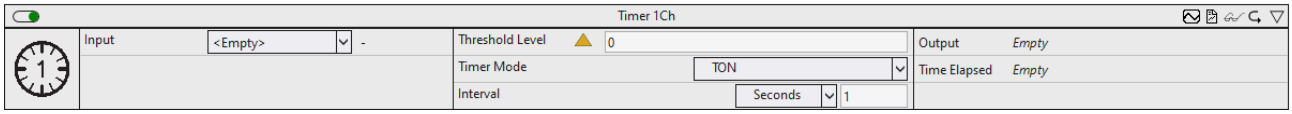
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29





また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをTimer Clock 1Chアルゴリズムにマッピングすることもできます。

6.5.1.24 Timer 1Ch



*Timer 1Ch*は、タイマーモードと区間により設定したタイマーを起動します。設定した条件が*TRUE*（TON、TP）または*FALSE*（TOF）になった場合、特定のタイマモードにしたがってタイマーが開始されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が閾値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Threshold:** 信号の閾値。
- **Timer Mode :** タイマーモード：

  *TON:*TONタイマーは、閾値条件が*TRUE*になり、指定した時間が経過した後に出力を有効にするスイッチオン遅延タイマーです。

  TOF：TOFタイマーは、閾値条件が*FALSE*になり、指定した時間が経過した後に出力を無効にするスイッチオフ遅延タイマーです。

  *TP:*TPタイマーはパルス発生器で、閾値条件が *TRUE* になると、指定された時間の間だけ出力を有効にします。

- **Interval:**設定されたタイマーの時間区間。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

出力値

- **Output:**設定したタイマーの影響を受ける出力値を表示します。
- **Elapsed Time:** タイマーの経過時間を表示します。

HMIコントロール

Timer 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.Timerコントロールは出力値 Elapsed Timeを表示します。

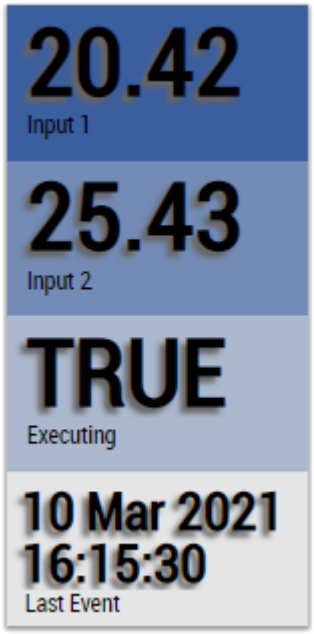
2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Output, Elapsed Time。

Data Table Vertical

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

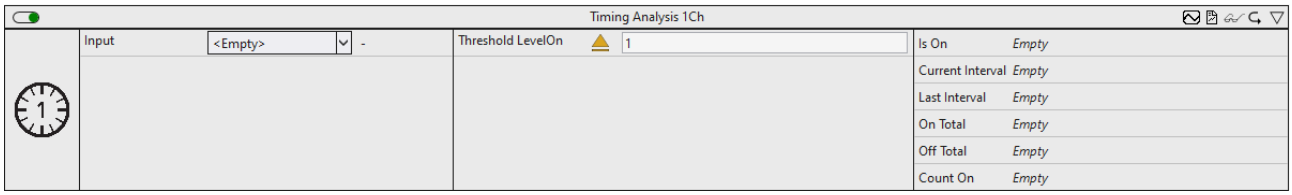
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをTimer1Chアルゴリズムにマッピングすることもできます。

6.5.1.25 Timing Analysis 1Ch



*Timing Analysis 1Ch*は、オン期間とオフ期間の時間差を計測し、オンの期間をカウントします。オン期間は、演算子と閾値の条件が満たされたときに開始されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が閾値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Threshold:**信号の閾値。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

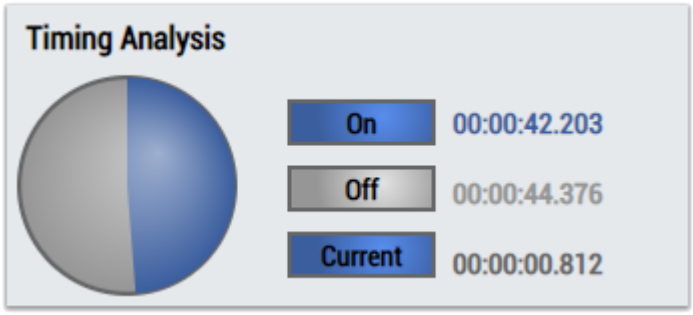
出力値

- **Is On:**オン期間内は*TRUE* を示し、そうでなければ*FALSE* を示します。
- **Current Interval:** 現在の区間の時間を示します。
- **Last Interval:**最後に完了した区間の時間。
- **On Total :** 「Is On」値が*TRUE*の合計時間を示します。
- **Off Total:**「Is On」値が*FALSE*の合計時間を示します。
- **Count On :** トリガーされたオンイベントの数をカウントします。

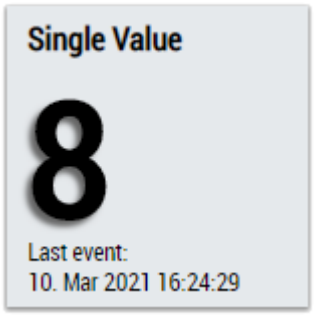
HMIコントロール

Timing Analysis 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

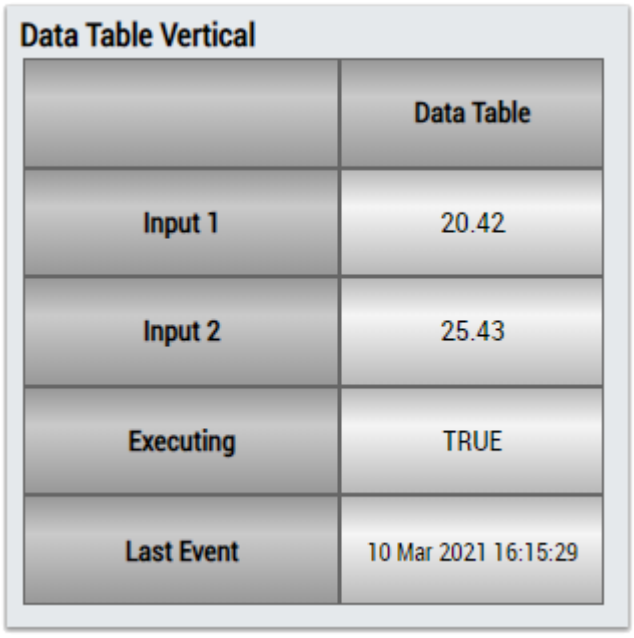
1.TimingAnalysis コントロールは、出力値 Is On、On Total、Off Total、および Current Interval を表示します。

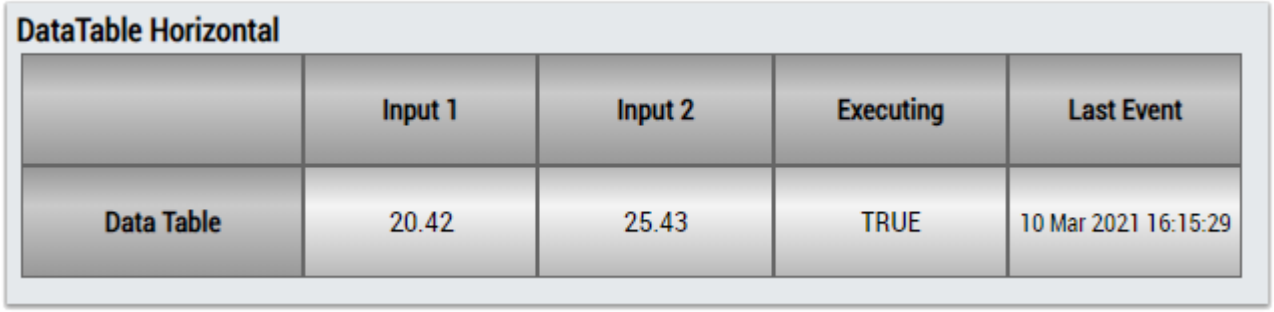


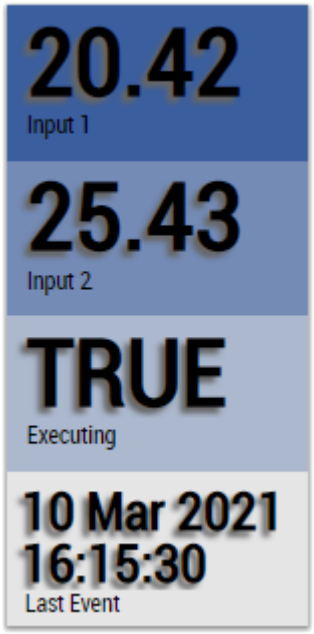
2.SingleValue コントロールは、出力値 Count On を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Is On、Count On、Current Interval、On Total、Off Total。





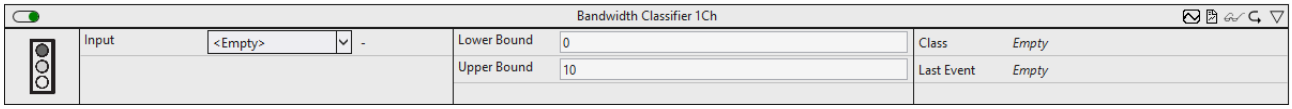


また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをTiming Analysis 1Chアルゴリズムにマッピングすることもできます。

6.5.2 Analytics - Classification（解析 - 分類）

*Analytics-Classification*カテゴリのアルゴリズムは、分類と状態検出の機能を提供します。

6.5.2.1 Bandwidth Classifier 1Ch



*Bandwidth Classifier 1Ch*は、入力信号が設定された制限値内であるか、制限値より小さいか大きいかを判断します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Lower Bound:**比較の下限値。
- **上限：**比較の上限。

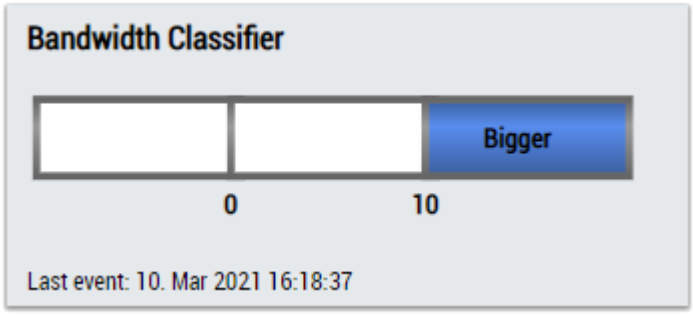
出力値

- **Class:**入力値が属するクラス（WithinBounds/Smaller/Bigger）を示します。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

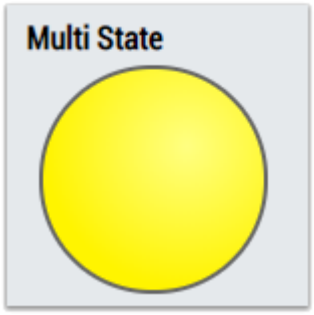
HMIコントロール

Bandwidth Classifier 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.BandwidthClassifierコントロールは、出力値ClassとLast Event、および設定オプションLower BoundとUpper Boundを表示します。



2.MultiState コントロールは、出力値 Class を表示します。



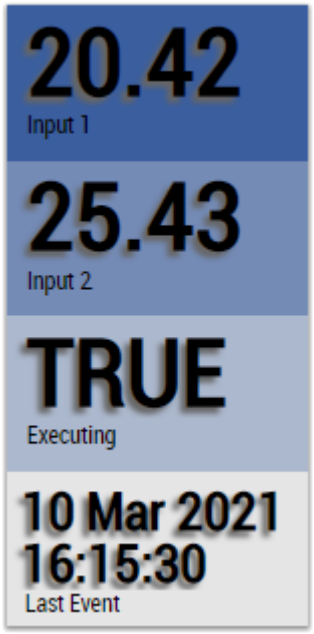
3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Class, Last Event。

Data Table Vertical

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

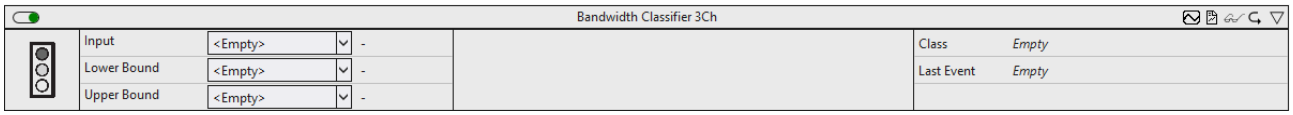
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをBandwidth Classifier 1Chアルゴリズムにマッピングすることもできます。

6.5.2.2 Bandwidth Classifier 3 Ch



*Bandwidth Classifier 3Ch*は、入力信号が制限値内か、制限値未満か、制限値以上かを判断します。制限値は入力信号で設定できるので、曲線を下限帯域および上限帯域として使用することも可能です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

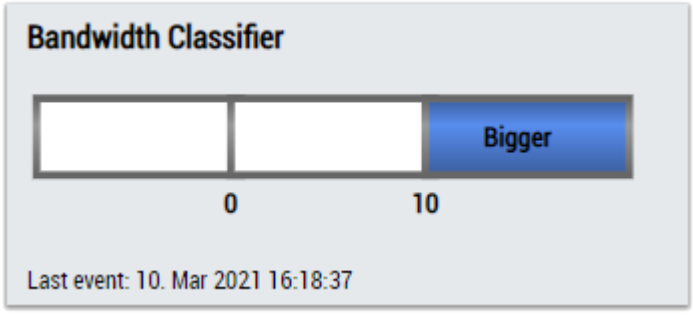
出力値

- **Class:**入力値が属するクラス（WithinBounds/Smaller/Bigger）を示します。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

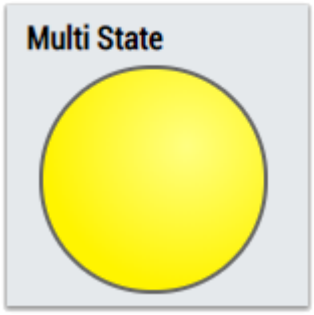
HMIコントロール

Bandwidth Classifier 3Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.BandwidthClassifierコントロールは、出力値ClassとLast Event、Input Lower BoundとInput Upper Boundを表示します。



2.MultiState コントロールは、出力値 Class を表示します。



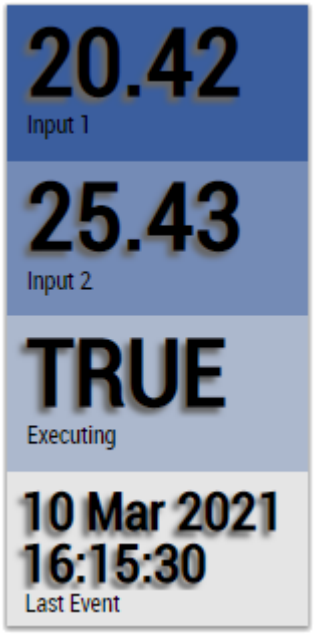
3.Tableコントロールは、以下、全ての出力値を表示します：Class, Last Event。

Data Table Vertical

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

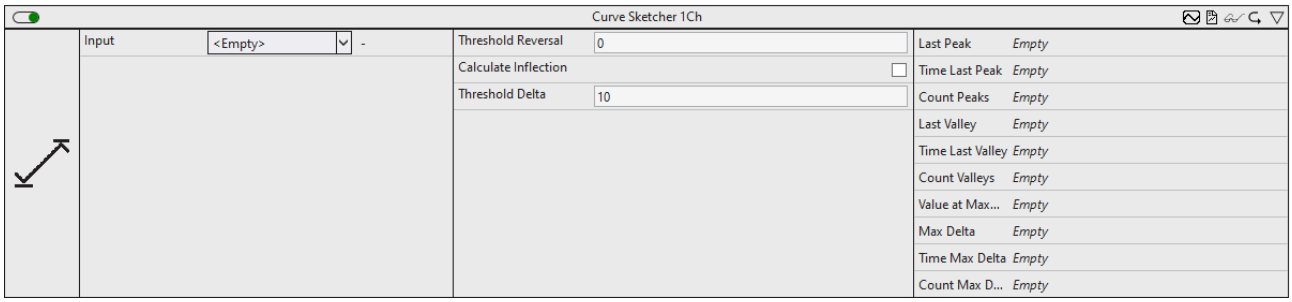
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをBandwidth Classifier 3Chアルゴリズムにマッピングすることもできます。

6.5.2.3 Curve Sketcher 1Ch

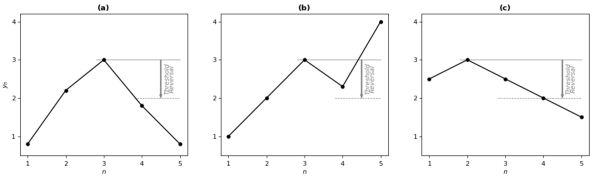


*Curve Sketcher 1Ch*は、入力データストリーム中の反転（*peaks*と*valleys*）を識別します。さらに、連続する2つの値の差の絶対値（*Delta*と呼ぶ）の局所最大値を検出することができます。連続曲線と同様に、検出されたpeakは極大値、valleyは極小値に対応します。Deltaは勾配に対応し、絶対値のピークは変曲点の指標となります。

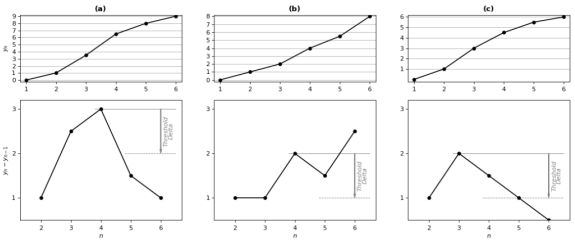
オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Calculate Inflection:**ブールフラグ。変化率の最大値は、このフラグが*True*の場合のみ検出されます。そうでない場合には、*Count Max Delta*、*Max Delta*、*Time Max Delta*、*Value at Max Delta*の値は計算されません。
- **Threshold Reversal :** 反転を識別するための閾値 。Threshold Reversal の値を超える差がなければ反転は検出されません。
  以下に、パラメータ *Threshold Reversal* を用いたピーク識別の3つの例を示します。
  (a) 値 y_3 と y_4 の差が *Threshold Reversal* より大きいため、y_4 の処理直後に y_3 はピークとして識別されます。
  (b) y_3 と y_4 の差が *Threshold Reversal* 未満であり、y_4 の後に曲線が再び上昇し始めるため、y_3 はピークとして識別されません。
  (c) y_2 と y_5 の差が *Threshold Reversal* を超過するため、y_5 を処理した後に y_2 はピークとして識別されます。y_2とy_3（またはy_4）との差が *Threshold Reversal* 以下であり、その後も値が減少し続けるかどうかが不明なため、y_2 を事前にピークとして識別することはできません。



- **Threshold Delta:** Deltaの最大値を識別するための閾値。連続する2つの値の差の絶対値（Delta）の最大値は、連続するDelta同士の差が *Threshold Delta* を超えた場合にのみ検出されます。
  以下に、パラメータ *Threshold Delta* を用いたDelta最大値の識別例を3つ示します。上段の図は元の入力信号を、下段の図はそれに対応するDeltaを示しています。
  (a) 2つのDeltaの差が *Threshold Delta* を超えているため、値 y_5 を処理した後に y_4 が最大値として識別されます。
  (b) Deltaの差が *Threshold Delta* 未満のため、最大値は識別されません。
  (c) 値 y_6 を処理した後、y_3 が最大値として識別されます。



*Threshold Delta*に関係なく、2つの反転のDeltaのうち少なくとも1つの最大値が検出されます。

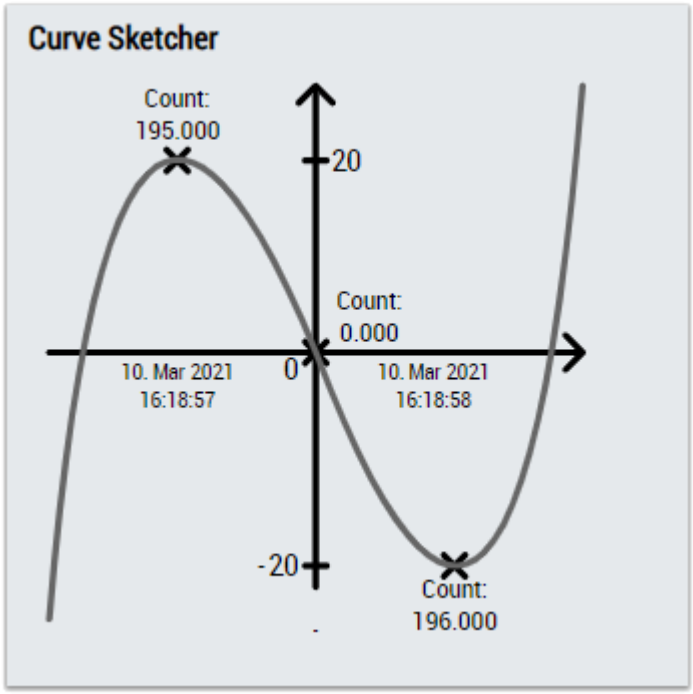
出力値

- **Last Peak:**最後に確認されたピークのy値を示します。
- **Time Last Peak:** 最後に確認されたピークのタイムスタンプを示します。
- **Count Peaks:**カウントされたピークの総数を示します。
- **Last Valley:**最後に確認されたvalley(極小値）のy値を示します。
- **Time Last Valley:**最後に確認されたvalley（極小値）のタイムスタンプを示します。
- **Count Valleys:**カウントされたvalley（極小値）の総数を示します。
- **Value at Max Delta:**最後に検出されたDeltaの最大値によって導かれるストリーム（入力変数）のy値を示します。このDelta値は、*Value at Max Delta*と1サイクル前の値との差です。
- **Max Delta :** 入力ストリーム内の連続する2つの値の差（絶対値）で、最後に検出された局所的な最大値を示します。
- **Time Max Delta:** *Value at Max Delta* のタイムスタンプ。
- **Count Max Delta :** Deltaのローカル最大値のカウント総数を示します。

HMIコントロール

Curve Sketcher 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.CurveSketcherコントロールは、データの入力値と、出力値Last Peak、Time Last Peak、Count Peaks、Last Valley、Time Last Valley、Count Valley、Last Delta、Time Last Delta、Count Deltaを表示します。



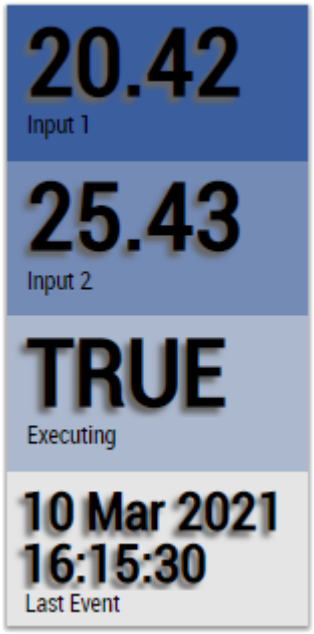
2.Tableコントロールは、以下、全ての出力値を表示します：Last Peak, Time Last Peak, Count Peaks, Last Valley, Time Last Valley, Count Valley, Value at Max Delta, Last Delta, Time Last Delta and Count Delta。

Data Table Vertical

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

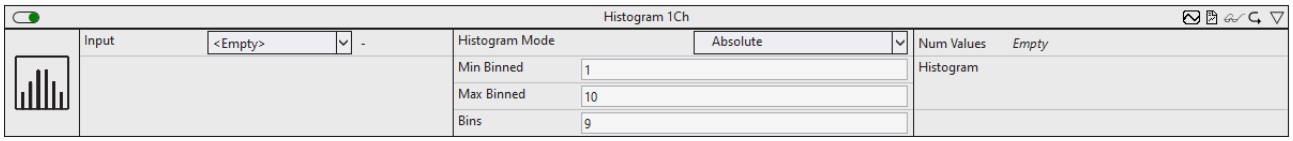
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをCurve Sketcher 1Chアルゴリズムにマッピングすることもできます。

6.5.2.4 Histogram 1Ch



*Histogram 1Ch*は、シングルチャンネルの入力値の分布を周期的に計算します。最小ビン、最大ビン、ビンの総量を設定できます。出力配列の次元は、ビンの数 + 2 です。最小ビンより小さい値は最初の配列要素に格納され、最大ビンより大きい値は最後の配列要素に格納されるためです。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Histogram Mode:**ヒストグラムの動作モード。絶対出力と相対出力は区別されます。後者は、パーセンテージ分布を表示するために使用できます。
- **Min Binned:** 解析する最小値。
- **Max Binned:**解析する最大値。
- **Bins:**計算されるヒストグラムクラスの総数。
- **Enable Display Names** : Histogram Display Names（ヒストグラム表示名）を有効にします。この設定は、プロパティウィンドウで非表示の状態で表示されます。
- **Histogram Display Names:**個々のビンの表示名。この設定は、プロパティウィンドウで非表示の状態で表示されます。

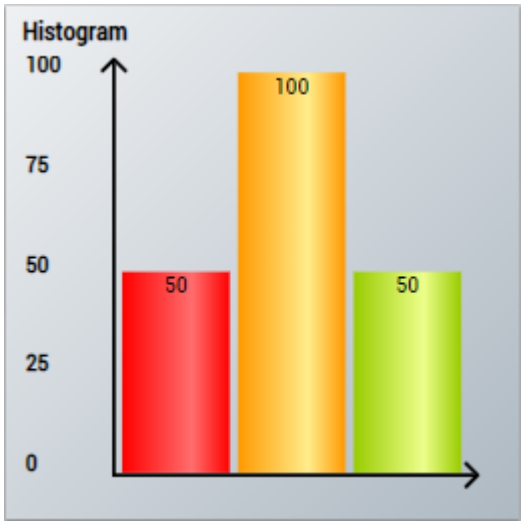
出力値

- **Num Values:** 分布の解析対象となる値の総数を示します。
- **Histogram:**Num Valuesの下にヒストグラムが表示されます。バーの上にカーソルを移動すると、値の区間とヒストグラムビンの値がツールチップに表示されます。

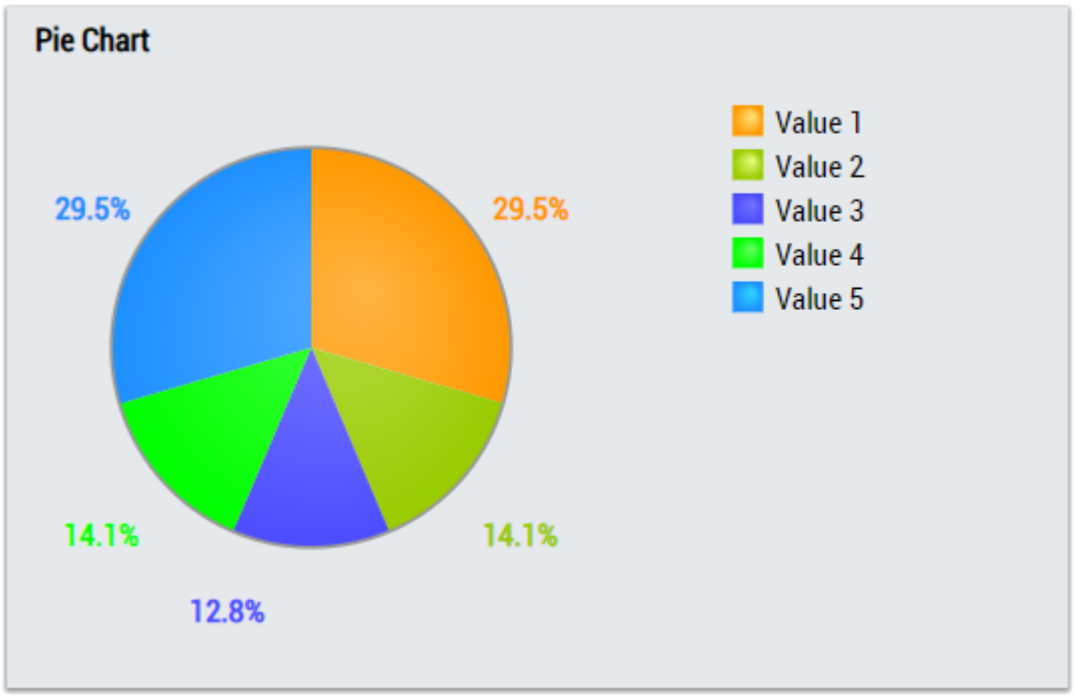
HMIコントロール

Histogram 1Ch アルゴリズムでは、Analyticsダッシュボードを生成するために、以下の HMIコントロールを使用できます：

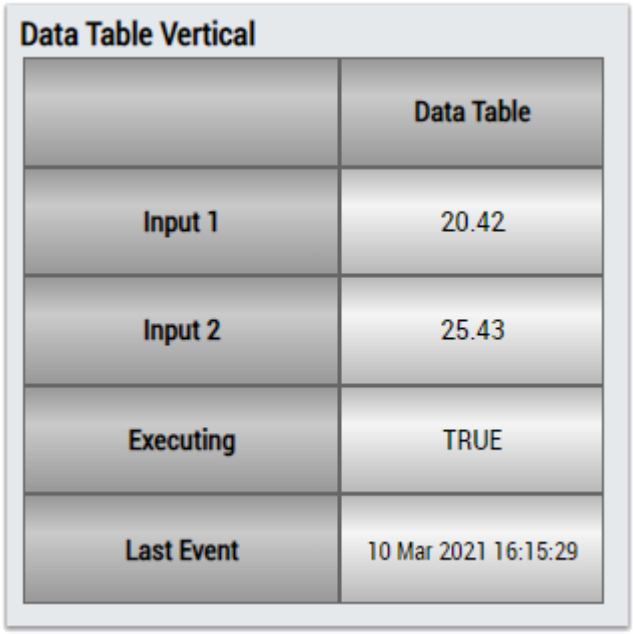
1.Histogramコントロールは、出力値 Num Values を表示します。

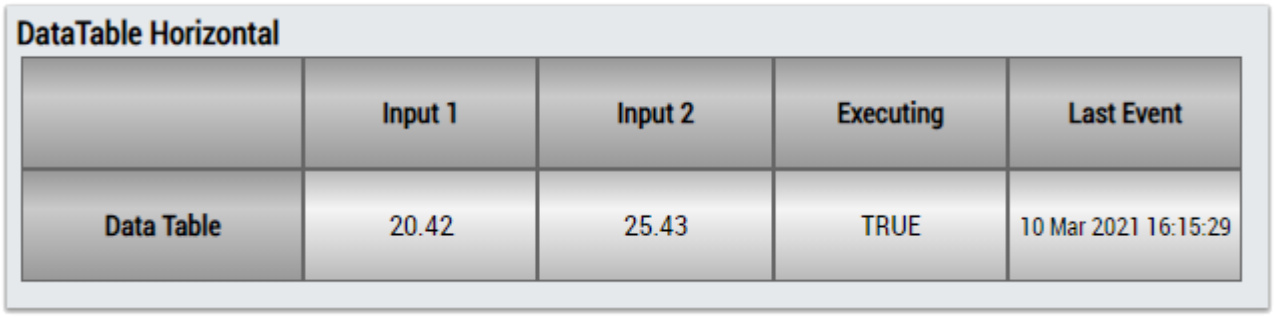


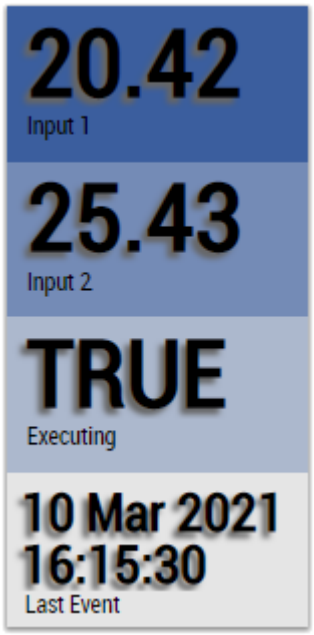
2.PieChart コントロールは、出力値 Num Values を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Num Values, Histogram。

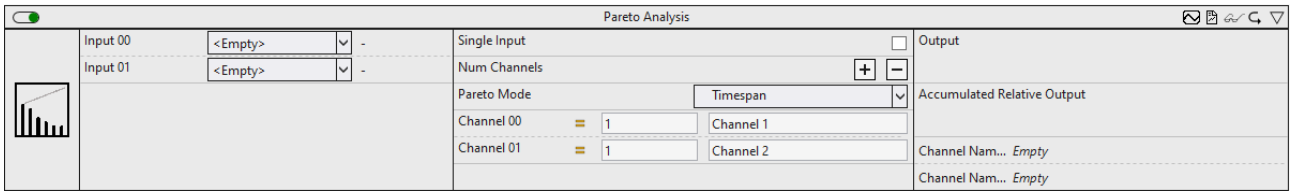






また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMI コントロールをHistogram 1Ch アルゴリズムにマッピングすることもできます。

6.5.2.5 Pareto Analysis



*Pareto Analysis*モジュールでは、異なる入力データを組み合わせてパレート図を表示できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Single Input:** 単一の入力チャンネルを使用するか、複数の独立した入力チャンネルを使用するかを選択します。
- **Num Channels:** *Single Input*がFALSEの場合の入力チャンネル数。
- **Num Elements:** *Single Input*がTRUEの場合に比較される要素数。
- **Pareto Mode:**出力値の計算モード。

  *Count:*各入力チャンネルがTRUE のサイクルがカウントされます。

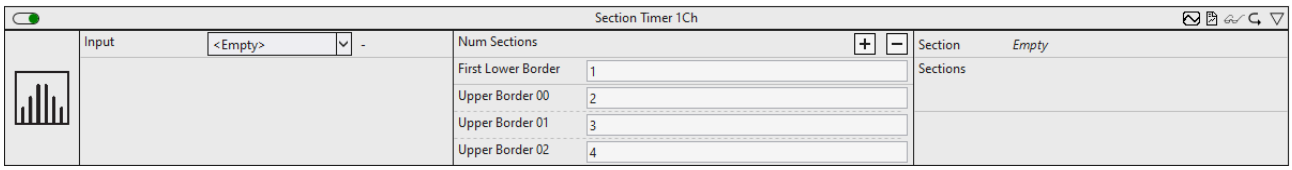
  *Timespan:*入力チャンネルがTRUE のタイムスパンが加算されます。
- **Threshold 00, Threshold 01, ..., Threshold n**：入力値を監視するための比較演算子と閾値。
- **Channel Name 00, Channel Name 01, ..., Channel Name n**: 入力チャンネルのエイリアス。

Enumeration（列挙）型の入力がリンクされている場合、このオプションは自動的に設定されます。
入力チャンネル数：1
比較される要素の数：列挙の数
閾値：個々の列挙の値

出力値

- **Output:**ソートされた絶対値。出力タイプはPareto Modeに依存します。
- **Accumulated Relative Output**：累積相対出力値。
- **Channel Name 00, Channel Name 01, ... Channel Name n:** ソートされた入力チャンネルのエイリアス名。

6.5.2.6 Section Timer 1 Ch



*Section Timer 1 Ch*は、入力が設定された各セクションの範囲内にある時間を計算します。セクション数やセクション同士の境界線を設定できます。各セクションは、下限（以上）と上限（未満）によって定義されます。次のセクションの下限は、前のセクションの上限によって決まります。下限より小さい値は、最初の配列要素に格納されます。上限より大きい値は、最後の配列要素に格納されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Sections:**セクションの数。
- **First Lower Border:**最初のセクションの下限。
- **Upper Border 00, Upper Border 01, ..., Upper Border n**：全てのセクションの上限。

- **Enable Display Names** : Section Display Names（セクション表示名）を有効にします。この設定は、プロパティウィンドウで非表示の状態で表示されます。
- **Section Display Names:**各セクションの表示名。この設定は、プロパティウィンドウで非表示の状態で表示されます。

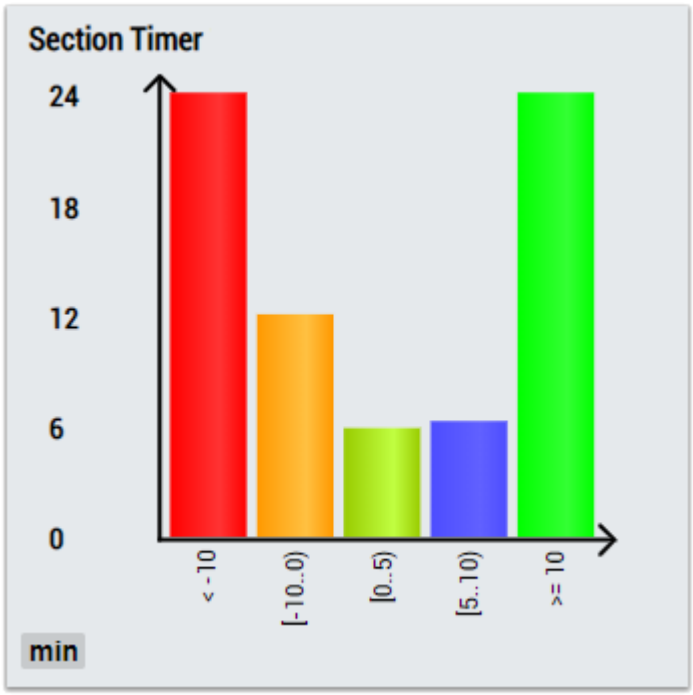
出力値

- **Section:**最後の入力値のセクションを指定します。入力値が*First Lower Border*より小さい場合、戻り値はゼロです。入力値が、区間[*First Lower Border*,*Upper Border 00*]にある場合、戻り値は1で、区間[*Upper Border 00*,*UpperBorder 01*]にある場合、戻り値は2となります。入力値が直近に指定された上限値*Upper Border 0n*より大きい場合、戻り値は*NumSections*+1です。
- **Array of Timespans** : 入力値が各セクションにソートされた合計時間。

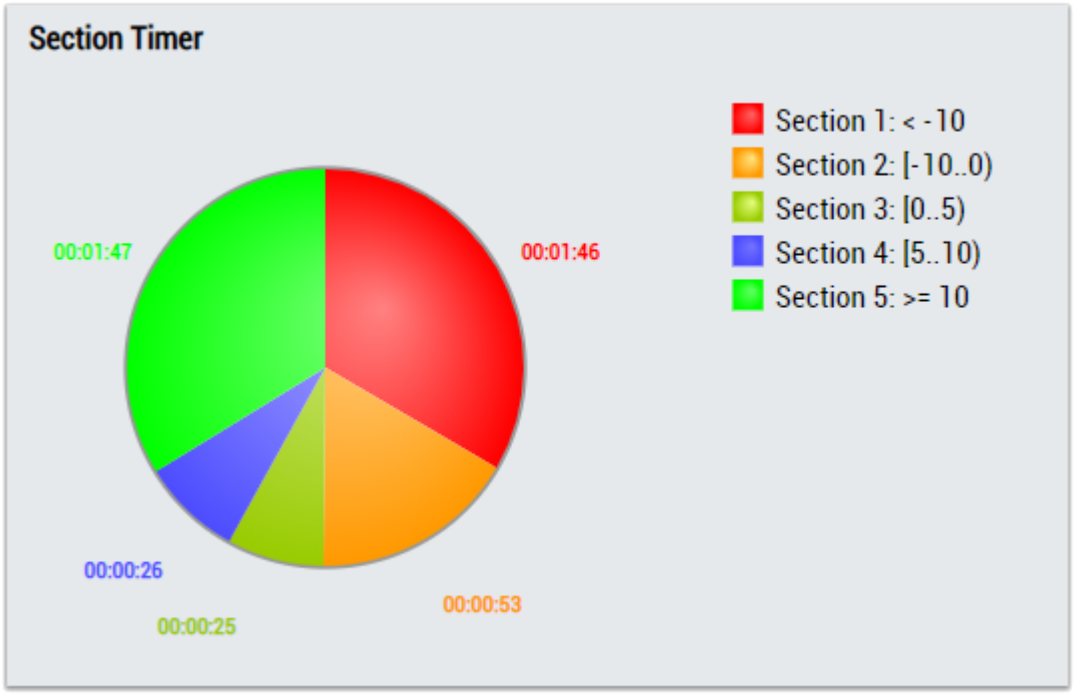
HMIコントロール

Section Timer 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.SectionTimer コントロールは、入力値 Array of Timespans と設定オプション First Lower Border と Upper Border を表示します。

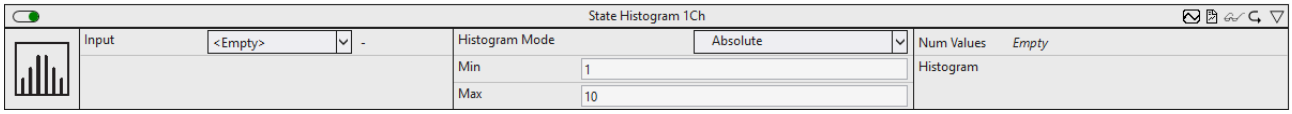


2.PieChart コントロールは、入力値 Array of Timespans と設定オプション First Lower Border と Upper Border を表示します。



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをセクション・タイマー1Chアルゴリズムにマッピングすることもできます。

6.5.2.7 State Histogram 1Ch



*State Histogram 1Ch*は、入力信号（INT）が設定した最小値と最大値の間で特定の値を持つ頻度をカウントし、その分布をヒストグラムで表示します。最初のバーは最小値より小さい下限を表し、最後のバーは最大値より大きい上限を表します。*State Histogram 1Ch*は、異なる状態の実行頻度を示すステートマシンに適しています。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Histogram Mode:**ヒストグラムの動作モード。絶対出力と相対出力は区別されます。後者は、パーセンテージ分布を表示するために使用できます。
- **Min:** 解析する最小値。
- **Max :** 解析する最大値。
- **Enable Display Names** : Histogram Display Names（ヒストグラム表示名）を有効にします。この設定は、プロパティウィンドウで非表示の状態で表示されます。
- **Histogram Display Names:**個々のビンの表示名。この設定は、プロパティウィンドウで非表示の状態で表示されます。

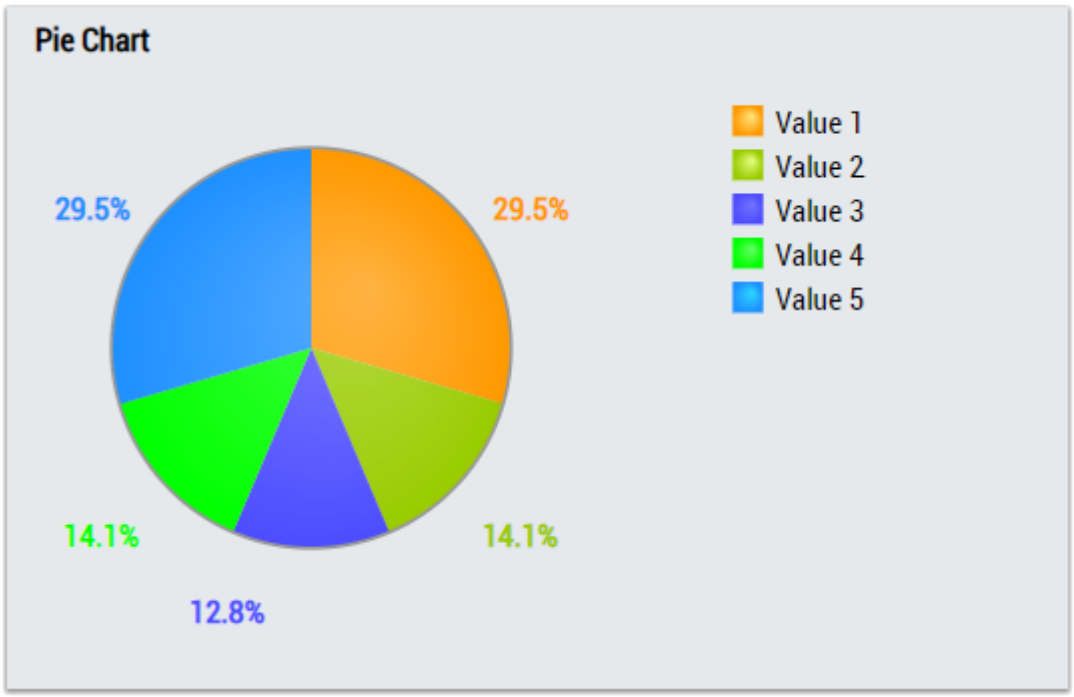
出力値

- **Num Values:**設定した境界線の間で実行された状態の総数を示します。
- **Histogram:**Num Valuesの下にヒストグラムが表示されます。各バーにそれぞれの値が表示されます。バーの上にカーソルを移動すると、値の区間とヒストグラムビンの値がツールチップで表示されます。

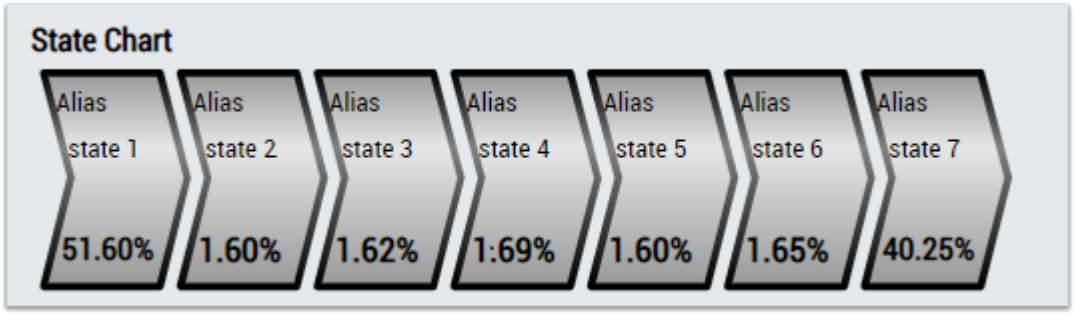
HMIコントロール

Histogram 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

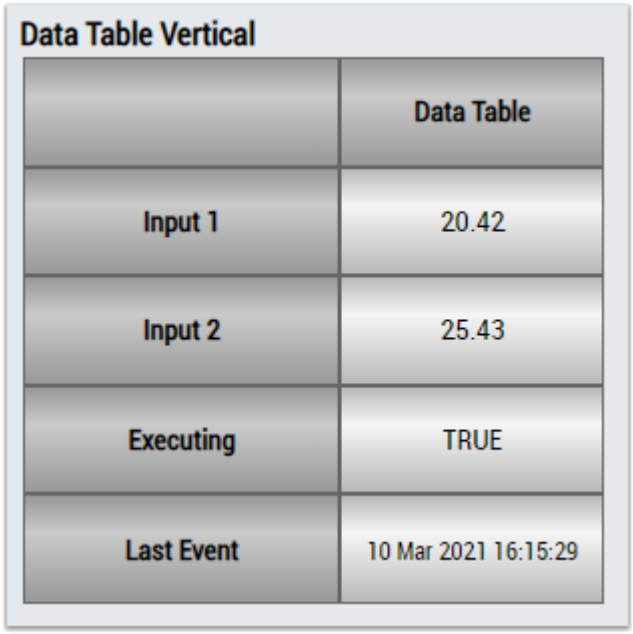
1.PieChart コントロールは、出力値のヒストグラムを表示します。

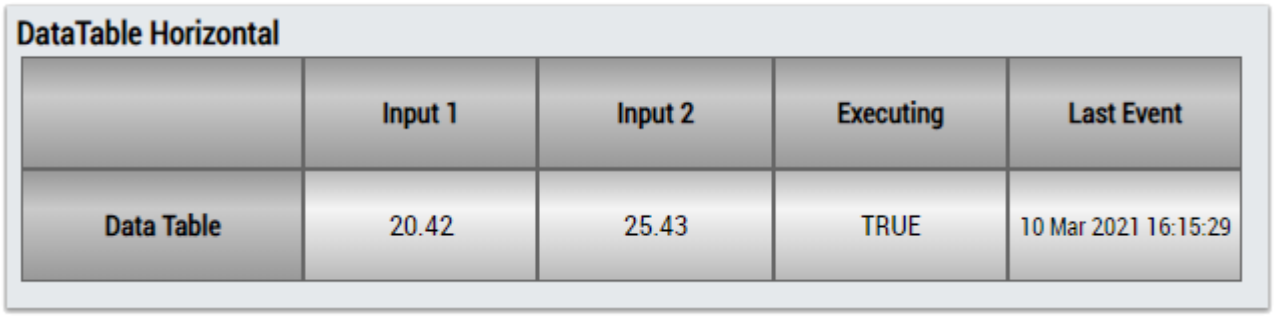


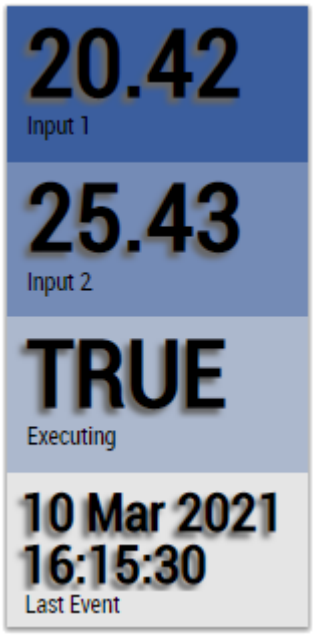
2.StateHistogram コントロールは、出力値のヒストグラムを表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Num Values, Histogram。

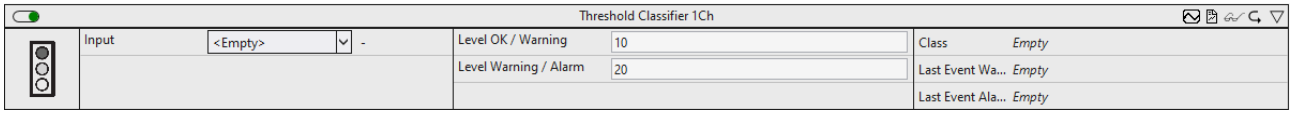






また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをState Histogram 1Chアルゴリズムにマッピングすることもできます。

6.5.2.8 Threshold Classifier 1Ch



*Threshold Classifier 1Ch*は、設定された閾値にしたがって、入力値を3つの異なるクラス：*OK*、警告、アラームに分類します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Level OK / 警告：**設定された閾値未満の入力値は*OK*として分類され、閾値以上の入力値は警告として分類されます。
- **Level Warning / Alarm: :** 設定された閾値未満の入力値は警告として分類され、閾値以上の入力値はアラームとして分類されます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。

出力値

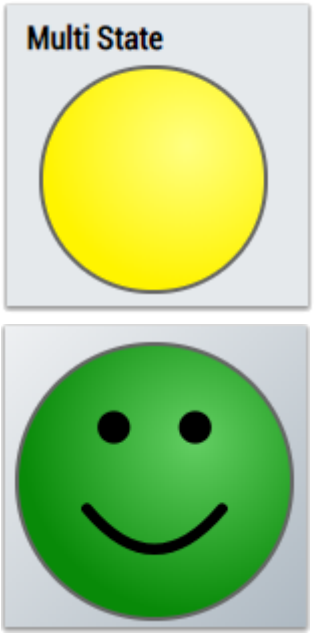
- **Class:**入力値が属するクラスを示します。
- **Last Event Warning:**最後にトリガーされた警告イベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Last Event Alarm:**最後にトリガーされたアラームイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

HMIコントロール

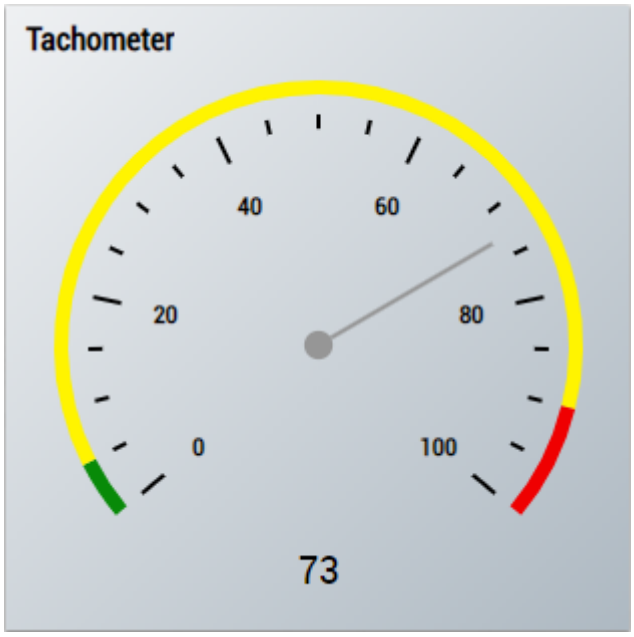
Threshold Classifier 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TrafficLightコントロールは、データの入力値と、出力値Class、Last Event Warning、Last Event Alarmを表示します。

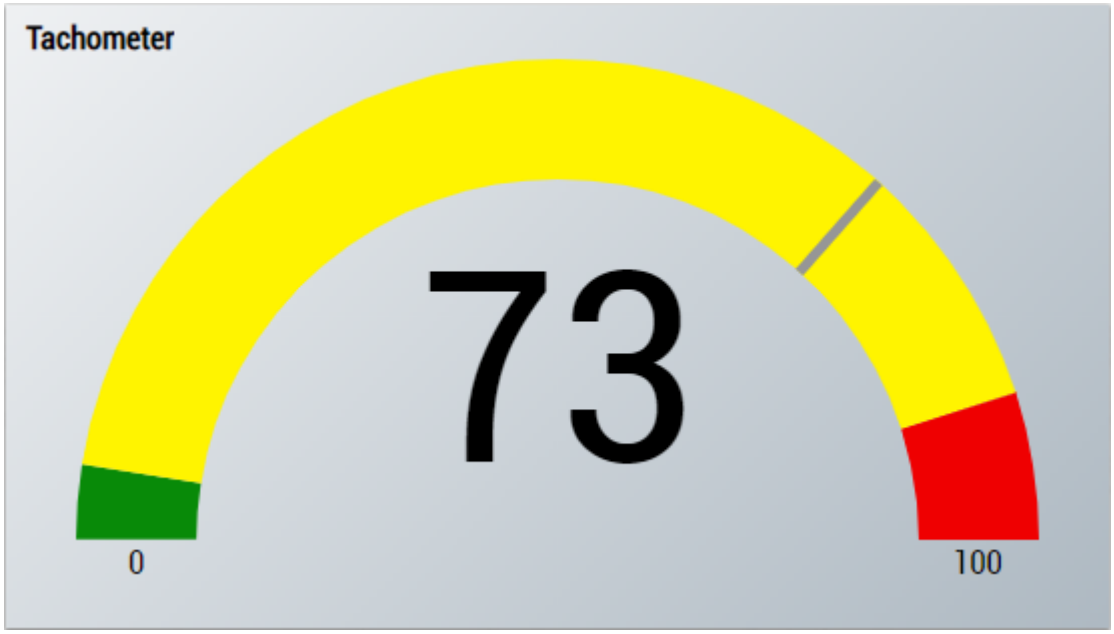
2.MultiState コントロールは、クラスの出力値を表示します（オプションで絵文字表示も可能）。



3.Tachometer コントロールは、設定した制限値を含む入力値を表示します。



4.Radial Gaugeコントロールは、設定した制限値を含む入力値を表示します。



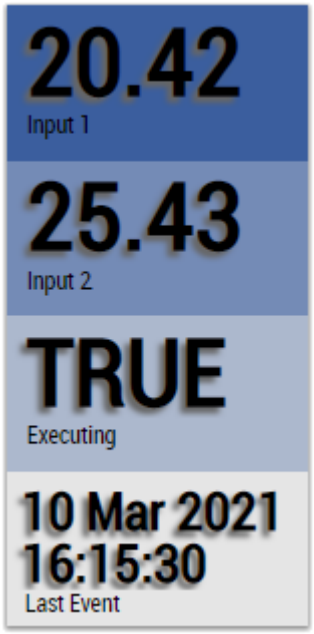
5.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Class, Last Event Warning Last Event Alarm。

Data Table Vertical

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

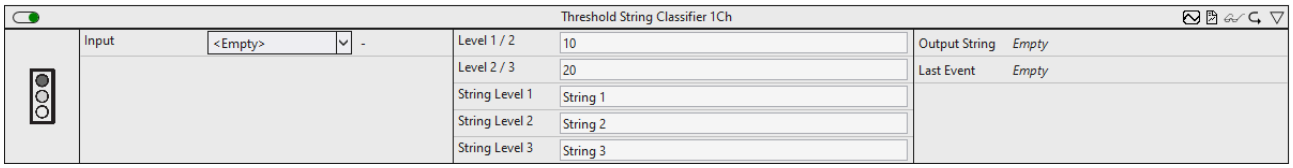
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをThreshold Classifier 1Chアルゴリズムにマッピングすることもできます。

6.5.2.9 Threshold String Classifier 1Ch



*Threshold String Classifier 1Ch*アルゴリズムは、設定した閾値にしたがって、入力値を3つの異なるクラスに分類します。クラス名（出力文字列）は、*String 1*、*String 2*、*String 3*として個別に設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Level 1 / 2:** 設定された閾値未満の入力値は*first*クラスに割り当てられ、設定された閾値以上の入力値は*second*クラスに割り当てられます。
- **Level 2 / 3:** 設定された閾値未満の入力値は*second*クラスに割り当てられ、設定された閾値以上の入力値は*third*クラスに割り当てられます。
- **String 1 :** 第一のクラス名。
- **String 2 :** 第二のクラス。
- **String 3:**第三のクラス名。
- **Tolerance（オプション） :** Equal / NotEqual 比較時に使用される許容値。

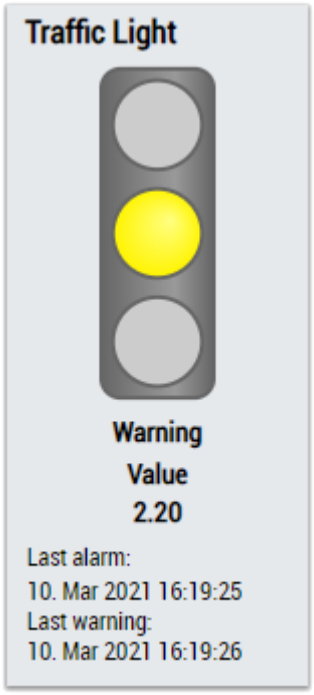
出力値

- **Output String:**入力値が属するクラスを示します。
- **Last Event:**最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

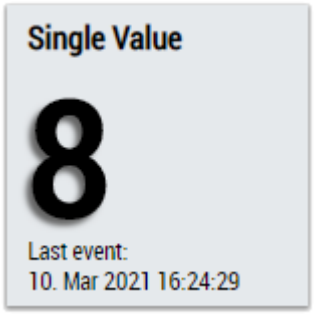
HMIコントロール

Threshold String Classifier 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

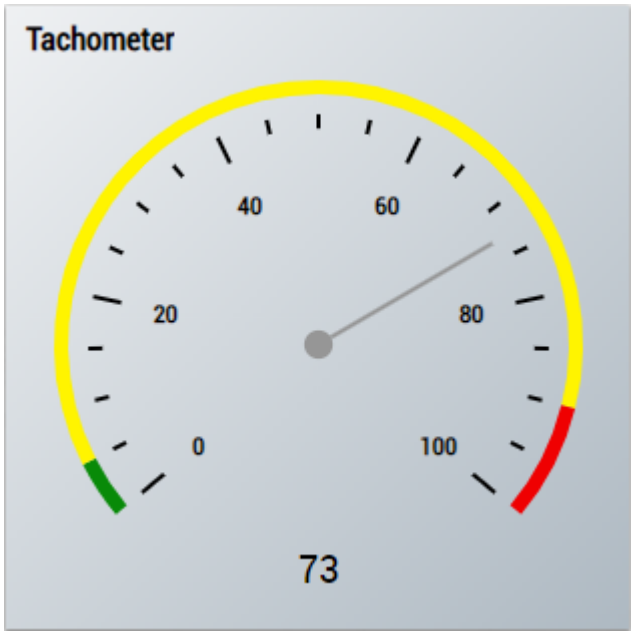
1.TrafficLight コントロールは、データの入力値と、出力値 Output String と Last Event を表示します。



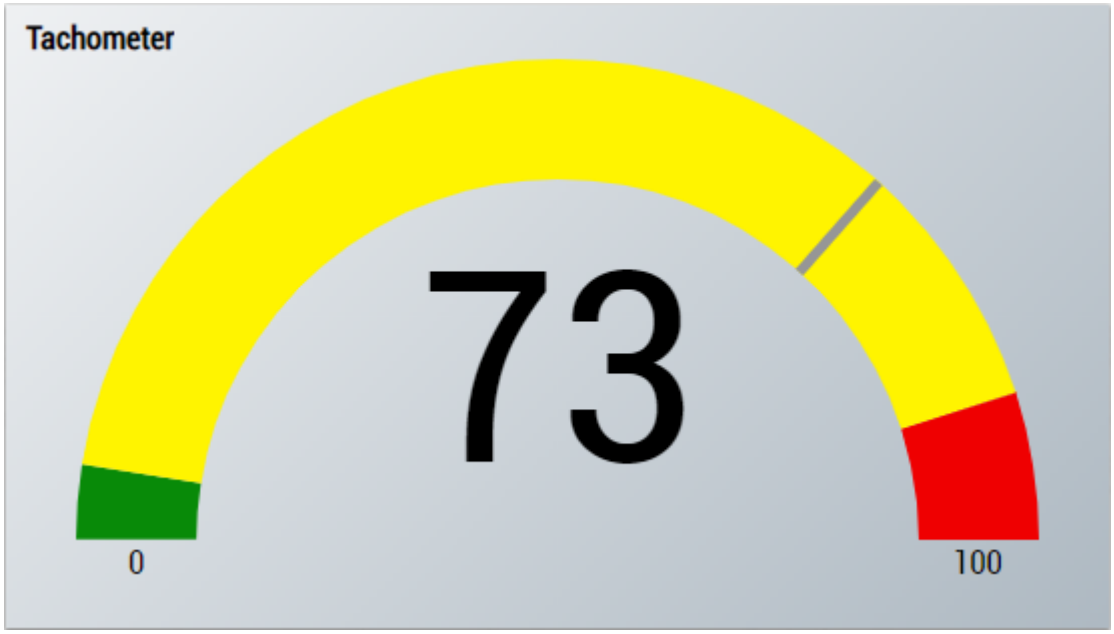
2.SingleValue コントロールは、出力値 Output Stringを表示します。



3.Tachometer コントロールは、設定した制限値を含む入力値を表示します。



4.Radial Gaugeコントロールは、設定した制限値を含む入力値を表示します。



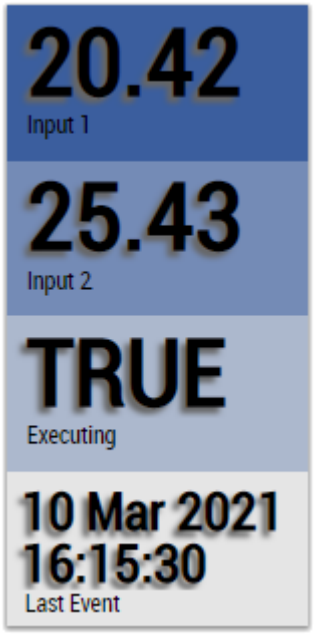
5.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Output String, Last Event。

Data Table Vertical

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

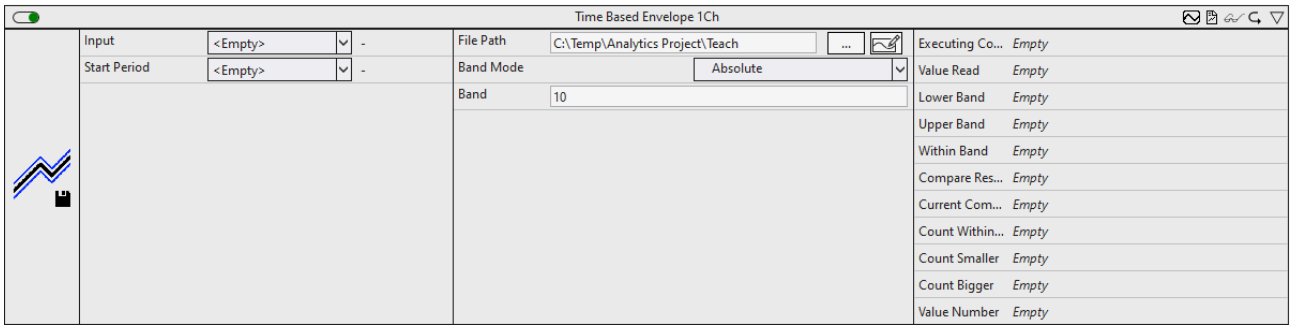
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをThreshold String Classifier 1Ch アルゴリズムにマッピングすることもできます。

6.5.2.10 Time Based Envelope 1Ch



*Time Based Envelope 1Ch*アルゴリズムは、周期的な入力データが、ファイルから読み込まれた設定値の範囲内にあるかどうかをチェックします。これは、Time Based Teach Path 1Ch [▶ 253] で紹介した基準信号に設定できます。比較は、Start Period（期間スタート）フラグの信号が*TRUE*になったときに開始されます。同時ファイルアクセスが発生するため、*Time Based Envelope 1Ch*とTime Based Teach Path 1Ch [▶ 253] の同時使用は推奨しません。代わりに、まず基準信号をTime Based Teach Path 1Ch [▶ 253] に教示してから、*Time Based Envelope 1Ch*を用いて評価を行うべきです。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **File Path:** 事前に教示されたデータファイルへのパス。
- **Band Mode:**帯域動作モード（絶対値または相対値を使用）。
- **Band :** 帯域動作の帯域幅。

出力値

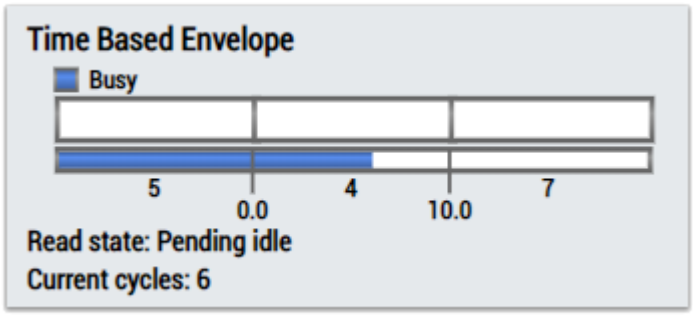
- **Executing Compare:** アルゴリズムがエンベロープ（信号包絡線）処理している場合は*TRUE* 、そうでない場合は*FALSE* を表示します。エンベロープ処理は、Start Period（期間スタート）フラグが*TRUE*になったときに開始されます。
- **Lower Band:** 帯域モードに応じて帯域の下限値を表示。
- **Upper Band:** 帯域モードに応じて帯域の上限値を表示します。

- **Within Band :** 現在の値が帯域内にあれば*TRUE*を表示し、そうでなければ*FALSE*を表示します。
- **Compare Result:**現在の比較結果。現在の値が帯域内にあるか、帯域より小さいか大きいかを示します。
- **Current Compared Cycles:** 比較されたサイクル数。
- **Count Within Band:**値が帯域内にある頻度をカウントします。
- **Count Smaller :** 値が帯域を下回る頻度をカウントします。
- **Count Bigger:** 値が帯域を上回る頻度をカウントします。
- **Value Number:** 現在比較されているファイルのデータポイントの値を表示します。

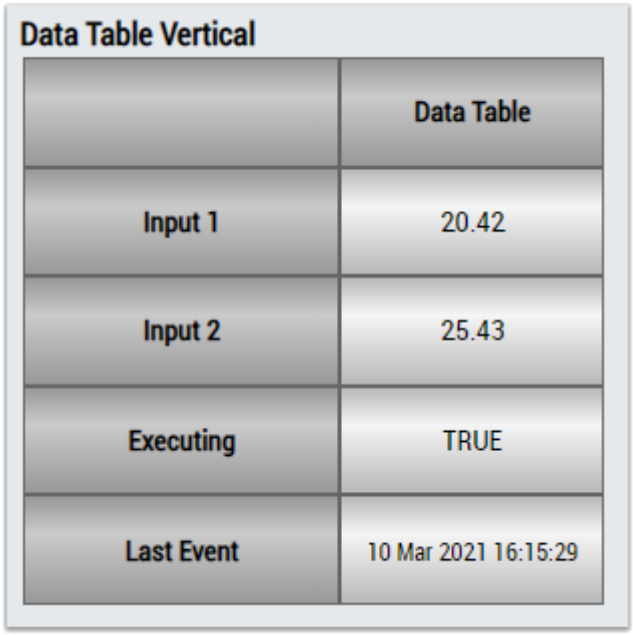
HMIコントロール

Time Based Envelope 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TimeBasedEnvelopeコントロールは、以下、全ての出力値を表示します：Executing Comparison, Lower Band, Upper Band, Within Band, Compare Result, Count Within Band, Count Smaller, Count Bigger, Current Compared Cycles, State and Value Number。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Executing Comparison, State, Lower Band, Upper Band, Within Band, Compare Result, Current Compared Cycles, Count Within Band, Count Smaller, Count Bigger, Value Number。

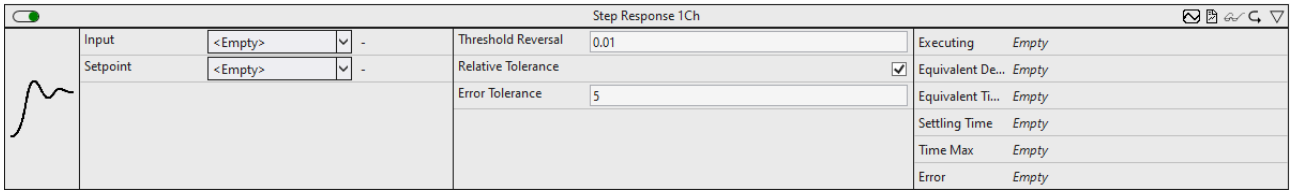


DataTable Horizontal

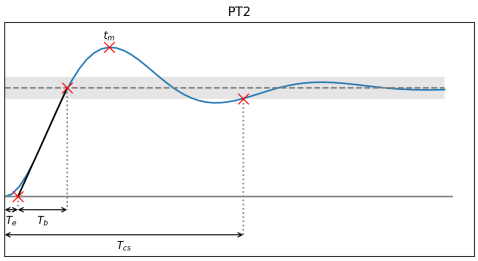
	Input 1	Input 2	Executing	Last Event
Data Table	20.42	25.43	TRUE	10 Mar 2021 16:15:29

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをTime Based Envelope アルゴリズムにマッピングすることもできます。

6.5.2.11 Step Response 1Ch



*Step Response 1Ch*は、PT2トラックのステップ応答のパラメータを識別します。これらには、遅延時間T_e、補償時間T_b、整定時間（定常状態に戻るまでの時間）T_{cs}、最大時間t_mが含まれます。



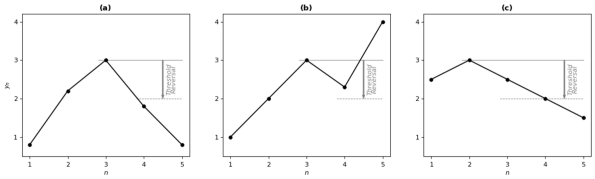
トラックが安定しているかどうかを検出するために、局所的な最小値または最大値が探索されます。これが許容範囲内（グレーで表示）であれば、トラックは安定していると見なされます。その後、整定時間が設定されます。

アルゴリズムは、新しい設定値が事前に保存された許容範囲の外にあるときに開始されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Threshold Reversal :** 反転を識別するための閾値 。Threshold Reversal の値を超える差がなければ反転は検出されません。
  以下に、パラメータ *Threshold Reversal* を用いたピーク識別の3つの例を示します。
  (a) 値 y_3 と y_4 の差が *Threshold Reversal* より大きいため、y_4 の処理直後に y_3 はピークとして識別されます。
  (b) y_3 と y_4 の差が *Threshold Reversal* 未満であり、y_4 の後に曲線が再び上昇し始めるため、y_3 はピークとして識別されません。
  (c) y_2 と y_5 の差が *Threshold Reversal* を超過するため、y_5 を処理した後に y_2 はピークとして識別されます。y_2とy_3（またはy_4）との差が *Threshold Reversal* 以下であり、その後も値が減少し続けるかどうかが不明なため、y_2 を事前にピークとして識別することはできません。



- **Relative Tolerance:**ブールフラグ。このフラグが*True* の場合、パラメータ*Error Tolerance*は、入力の設定値をパーセンテージで参照します。それ以外の場合は、絶対的な許容範囲が考慮されます。
- **Error Tolerance :** *Relative Tolerance*（相対公差）パラメータに対する公差帯の大きさを指定します。許容範囲は、パラメータ識別が再開されたときに更新されます。

出力値

- **Executing:**パラメータ識別が有効の場合True。
- **Equivalent Dead Time:** トラックの遅延時間。ステップ応答の開始から、変曲点の接線が開始値と交わるまでの時間。
- **Equivalent Time Constant:**トラックの補償時間。変曲接線が開始値と交わる点から、設定値と交わる点までの時間。
- **Settling Time :** トラックの整定時間。
- **Time Max :** 最大オーバーシュートの発生時間。
- **Error:**入力値と設定値の差。

HMIコントロール

*Step Response 1Ch*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールとMultivalueコントロールは、全ての出力値を表示します。

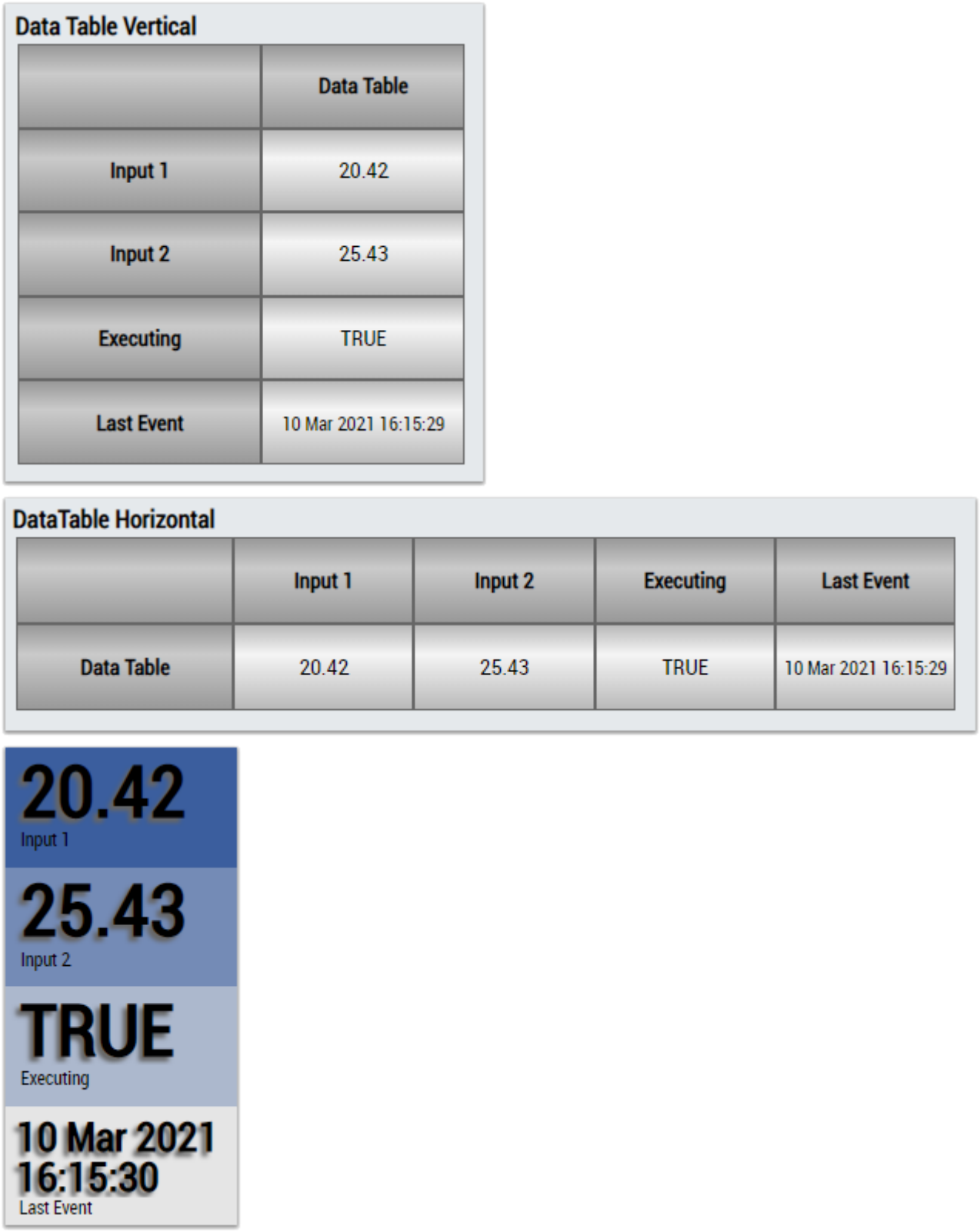


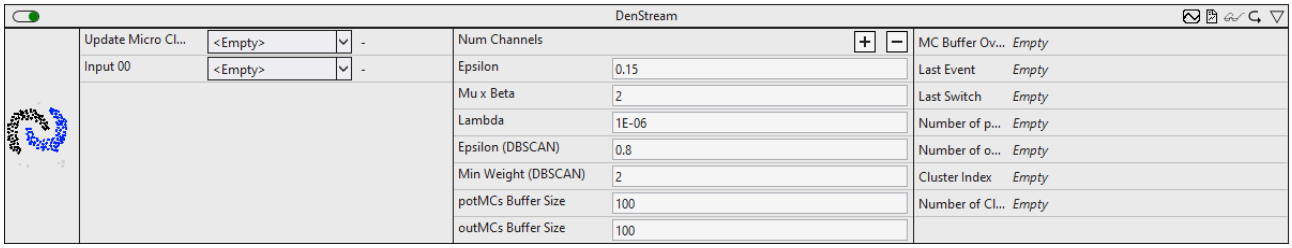
図 2:

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをマッピングすることもできます。

6.5.3 Analytics - Clustering（解析 - クラスタリング）

*Clustering*カテゴリのアルゴリズムは、入力データを事前に定義されていない様々なクラスタにクラスタリングする機能を提供します。クラスタリングはストリーミングデータをベースに行われます。クラスタは、入力データに存在する構造に基づいて検出されます。このようなクラスタアルゴリズムの例としては、*Sequential k-Means*や *DenStream*があります。

6.5.3.1 DenStream



DenStreamは、同名の教師なし密度ベースクラスタリングアルゴリズムの実装です[1]。後者は、よく知られたクラスタリングアルゴリズムDBSCAN [2, 3]に基づいており、特に時間と共に構造が変化するデータストリームに適しています。

このアルゴリズムの入力チャンネル数（以下*n*と表記）は、ユーザが選択できます。これらの入力は、クラスタを見つけることができるn次元の特徴空間を形成します。各解析サイクルで、データストリームはアルゴリズムに、この特徴空間内のデータポイントとして解釈できる新しい特徴ベクトルを提供します。クラスタとは、特徴空間においてデータポイントの密度が高い分離可能なエリアのことです。

アルゴリズムの第一段階では、入力されたデータポイントはマイクロクラスタ（*MC*）に割り当てられます。これらのマイクロクラスタは、含まれるデータポイントに依存する特性（中心点、重み、分散など）を持っています。重みがある閾値を超えるマイクロクラスタのみが第2フェーズに入り、DBSCANアルゴリズムによってクラスタリングされます。したがって、各データポイントの情報を保持する必要はありません。これにより、時間の経過とともにデータポイントよりもマイクロクラスタの数がはるかに少なくなるため、メモリ要件が削減されます。また、DBSCANアルゴリズムはすべてのデータポイントではなく、縮小されたマイクロクラスタの集合上で処理を行うため、計算負荷が大幅に軽減されます。また、マイクロクラスタの重みにフェード関数を適用することも可能です。このように、古いデータポイントは時間の経過とともに、クラスタリング処理における重要性を徐々に失っていきます。これにより、アルゴリズムの変化（クラスタの移動や、時間の経過に伴う消失/出現など）を捉えることができます。

DenStreamアルゴリズムには、他のクラスタリングアルゴリズムにはない更なる利点があります。マイクロクラスタの数は、DenStreamアルゴリズムが自動的に決定するので、ユーザが事前に把握する必要はありません。さらに、どのクラスタにも属さないデータの外れ値を検出できます。密度ベースのアルゴリズムであるため、（それらが絡み合っていたとしても）いかなる形状のクラスタも検出可能です。

パラメータ設定

ここでは、アルゴリズムがどのように機能するか、主にパラメータ設定について簡単に紹介します。アルゴリズムとパラメータの詳しい説明は、記載された文献を参照してください。ここで使用されている用語やパラメータ名のほとんどは、これらの出版物から引用したものです。

DenStreamのパラメータは、主に以下のアルゴリズム特性に影響を与えます：

- マイクロクラスタの粒度
- データポイント/マイクロクラスタ間の最大距離（それらが同じクラスタに割り当てられるようにするため）
- 最小密度（データポイントが外れ値ではなくクラスタとして識別されるようにするため）
- 古いデータポイントの重要性が失われていく減衰率

Parameters Epsilon, Lambda、*Mu x Beta*は、アルゴリズムの第一段階、マイクロクラスタの形成に属します。

可能であれば、データポイントは最も中心が近いマイクロクラスタに割り当てられます。このため、データポイントとすべてのマイクロクラスタの中心点とのユークリッド距離が比較され、最も距離が小さいマイクロクラスタが選択されます。データポイントは、挿入後にマイクロクラスタの半径が*Epsilon*の閾値を超えない場合に限り、そのマイクロクラスタに挿入できます。半径は、マイクロクラスタ内に含まれるすべてのデータポイントの分散に相当します。これは、クラスタの中心までのEuclidean（ユークリッド）距離が*Epsilon*（イプシロン）より大きい場合でも、マイクロクラスタ内にそれより近い距離の点が十分に存在すれば、そのデータポイントをマイクロクラスタに統合できることを意味します。

データポイントが最も近いマイクロクラスタに挿入できない場合、そのデータポイントを持つ新しいマイクロクラスタが作成されます。各マイクロクラスタの重みは、データポイントの挿入によって1ずつ増加します。

図の左側のプロットでは、例として 2 つの入力チャンネルに対するデータポイントのマイクロクラスタへの割り当てが示されています。4つの異なるマイクロクラスタに割り当てられた20のデータポイントが示されています。1番目のマイクロクラスタ（#1、赤）には6つのデータポイントが含まれ、2番目（#2、緑）にも6つ、3番目（#3、青）には7つ、4番目（#1、グレー）には1つのデータポイントしか含まれていません。指定されたイプシロン（破線）に基づき、新しいデータポイントが各マイクロクラスタに受け入れられるために存在すべき、クラスタ中心周辺の領域が色付けされています。この影響範囲は、マイクロクラスタにすでに複数のデータポイントが含まれ、それらの分散が小さい場合に、より広くなります（例：マイクロクラスタ #1 および #2 を参照）。さらに、複数のマイクロクラスタの影響範囲は、重なり合い、相互に影響し合うことがあります。マイクロクラスタ#2（緑）と#3（青）を参照。データポイントは常に近い方のマイクロクラスタに割り当てられ、そのため影響範囲は直線で区切られています。プロットにはマイクロクラスタ #2 に割り当てられたデータポイントが示されていますが、もし #2 が存在しなければ、そのポイントはマイクロクラスタ #3 に割り当てられることになります。

[1]の学習内容と同様、マイクロクラスタはその重みに基づいて、潜在的マイクロクラスタと外れ値マイクロクラスタに分類されます。潜在的なマイクロクラスタのみが、DBSCANアルゴリズムによってクラスタリングされます。マイクロクラスタ内の外れ値に該当するデータポイントは外れ値としてマークされます。ただし、外れ値のマイクロクラスタも潜在的なマイクロクラスタに発展する可能性があるため、保存され、新しいデータポイントによってアップデートされます。潜在的なマイクロクラスタとしてカウントされるには、マイクロクラスタの重みが*Beta x Mu*の閾値を超える必要があります。例えば、図の左側では、マイクロクラスタ#4（グレー）は1つのデータポイントしか含まず、したがって1以下の重みとなり、*Beta x Mu* = 1の外れ値マイクロクラスタとしてカウントされます。

フェード関数が適用されると、マイクロクラスタの重みは時間とともに減少します。この減衰率は、パラメータ*Lambda*によって決定されます。値がゼロに設定された場合、フェード関数は適用されません。そうでない場合は、重みは毎秒2^(*-Lambda*)の割合で減少します。外れ値マイクロクラスタの重みが内部の閾値（*Mu x Beta* と *Lambda*依存）を下回ると、メモリから削除されます。

パラメータ*Epsilon (DBSCAN)*と*Min Weight (DBSCAN)*は第2フェーズに影響します。これらのパラメータはDBSCANアルゴリズム[3]から採用されています。

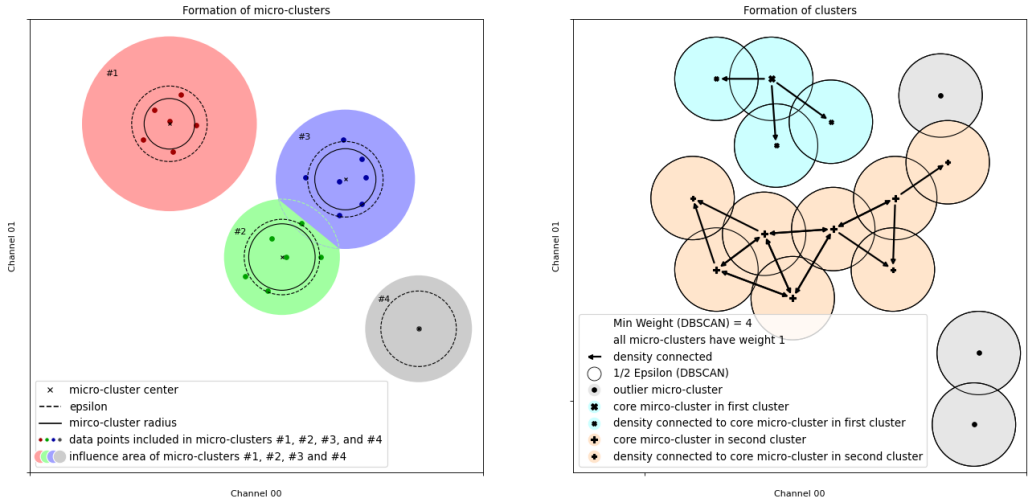
DBSCANアルゴリズムは、潜在的なマイクロクラスタの集合に対して実行され、それらにクラスタ指定を割り当てます。これは、クラスタが属するインデックス、あるいは外れ値の指定のいずれかとなります。処理中のデータポイントには、それが属するマイクロクラスタの名前が割り当てられます。

DBSCANがマイクロクラスタをクラスタリングする方法 このアルゴリズムは、密度アクセス性の概念に基づいて動作します。オブジェクト（この場合はマイクロクラスタ）は、密度結合していれば同じクラスタに属します。これは、最大距離*Epsilon (DBSCAN).*を持つマイクロクラスタの連鎖が存在する必要があることを意味します。この連鎖を形成する全てのマイクロクラスタは、第二の条件を満たす必要があります。この連鎖内の個々のマイクロクラスタの周りの距離*Epsilon*（*DBSCAN*）内にある全てのマイクロクラスタの重みの合計は、閾値*Min Weight*（*DBSCAN*）を超える必要があります。この2番目の条件を満たす少なくとも1つのマイクロクラスタに密度結合されていないマイクロクラスタは、外れ値としてマークされます。

これは図の右側に示されています。ここでは簡単にするために、すべてのマイクロクラスタの重みが1であると仮定します。これは、各マイクロクラスタにデータポイントが正確に1つあり、フェード関数が適用されていない場合に相当します。*Min Weight (DBSCAN)* パラメータが4に設定された場合、2つの外れ値マイクロクラスタを持つ2つのクラスタ（x（ターコイズ）および+（オレンジ）でマークされたもの）が生じます。大文字の 「x」または 「+」が付いているマイクロクラスタは、コアマイクロクラスタです。これは、少なくともあと3つのマイクロクラスタ（考慮されたマイクロクラスタを含めて計4つ）が、これらのマイクロクラスタとの最大距離が*Epsilon*（*DBSCAN*）であることを意味します。小さな 「x」または 「+」でマークされたマイクロクラスタは、コアマイクロクラスタではありません。ただし、コアマイクロ

クラスタのEpsilon（DBSCAN）近傍に位置し、したがって同じクラスタに属します。小さな点で示された右上のマイクロクラスタは、外れ値のマイクロクラスタです。クラスタに属するとカウントされるマイクロクラスタのEpsilon (DBSCAN)近傍に位置しますが、コアマイクロクラスタではありません。

同様に、右下の2つのマイクロクラスタは外れ値です。*Epsilon*（*DBSCAN*）のすぐ近くにありながら、2つしかありません。*Min Weight (DBSCAN)* の閾値を超過しません。



パラメータ *outMCs Buffer Size* と *potMCs Buffer Size* は、このアルゴリズムの実装に特有のものであり、外れ値および潜在的マイクロクラスタ用のメモリを実行前に割り当てる必要があるために必要です。したがって、*outMCs Buffer Size*と *potMCs Buffer Size*サイズは、実行中の外れ値や潜在的なマイクロクラスタの数を制限します。ユーザは、この制限を超えないように、これらのパラメータ値を見つける必要があります。

アルゴリズム実行中の外れ値および潜在的なマイクロクラスタの最大数は、入力データの分布だけでなく、他のパラメータの設定にも依存します。*Epsilon*の値が高いほどマイクロクラスタは少なくなり、より広い範囲のデータポイントを含む粗いマイクロクラスタになります。一般に、外れ値マイクロクラスタの数は、解析の初期には増加しますが、外れ値マイクロクラスタが潜在的マイクロクラスタに変化すると、再び減少します。データストリームのパターンが時間とともに変化しない場合、マイクロクラスタの数は初期フェーズ後に落ち着きます。

マイクロクラスタが多ければ多いほど、必要な計算量は多くなります。全ての外れ値と潜在的なマイクロクラスタについて、データポイントとの距離を比較し、その後、全ての潜在的なマイクロクラスタをDBSCANアルゴリズムの計算に含める必要があります。したがって、計算速度とマイクロクラスタの粗さの間で妥協点を見出す必要があります。

*outMCs Buffer Size*と *potMCs Buffer Size*の値が低く設定されすぎていて、解析中のある時点で入力データポイントを捕捉するために、より多くのマイクロクラスタが必要になった場合、どうなりますか？この場合、アルゴリズムはデータポイントを既存のマイクロクラスタに割り当て続け、それに応じてデータポイントをマークします。ただし、バッファのオーバーフローを防ぐために既存のマイクロクラスタは更新されなくなります。これは、データポイントのクラスタリングは継続するが、全体的に停滞した特徴空間（古いマイクロクラスタの集合）を持つことを意味します。データストリームのパターン変化は、検出できなくなります。

[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)

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

入力値

- **Update Micro Cluster:**TRUE の場合、マイクロクラスタは入力データによって更新されます。FALSEの場合、既存のマイクロクラスタは更新されず、入力データポイントのクラスタインデックスを決定するためにのみ使用されます。
- **Input 01, ..., Input 0n:**これらの入力はクラスタリングが実行される特徴空間を形成します。

設定オプション

- **Num Channels :** 入力チャンネル数を変更します。
- **Epsilon:**マイクロクラスタの最大半径の閾値。
- **Mu x Beta:**外れ値マイクロクラスタと潜在的マイクロクラスタを分けるための、マイクロクラスタの重みに対する閾値。
- **Lambda:**アルゴリズムの減衰率を指定します。各データポイントの重みは、毎秒2^(-*Lambda*)の割合で減少します。
- **Epsilon (DBSCAN):**DBSCANアルゴリズムのイプシロンパラメータを指定します。
- **Min Weight (DBSCAN):**DBSCANアルゴリズムで、マイクロクラスタのイプシロン近傍内に存在する重みの合計に対する閾値。
- **potMCs Buffer Size:**潜在的なマイクロクラスタの最大数。メモリは*potMCs Buffer Size*マイクロクラスタに割り当てられます。
- **outMCs Buffer Size:**外れ値マイクロクラスタの最大数。メモリは*outMCs Buffer Size*マイクロクラスタに割り当てられます。

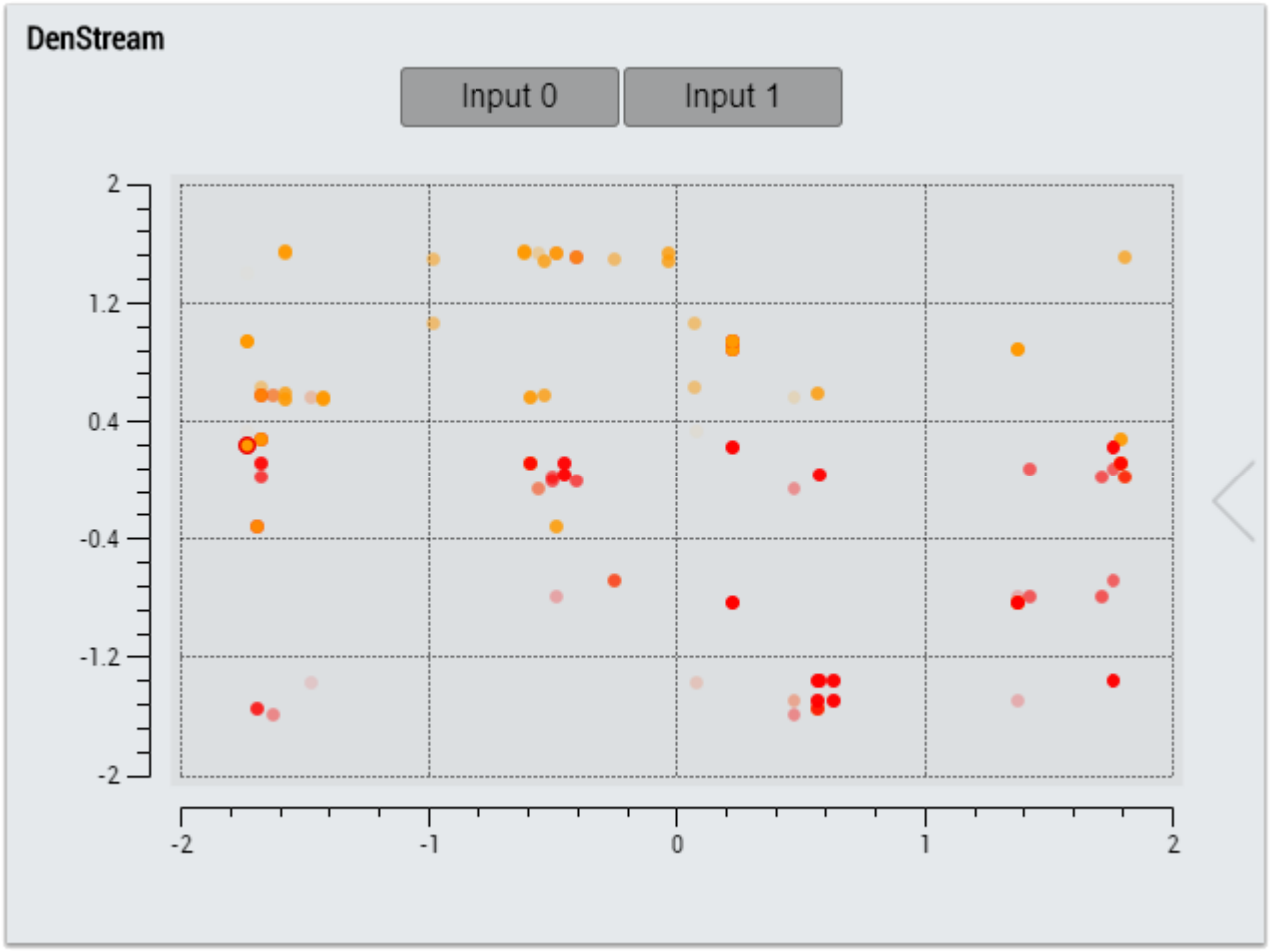
出力値

- **New Result:**新しいクラスタインデックスが最後のサイクルのクラスタインデックスと異なる場合にTRUE を返します。
- **MC Buffer Overflow:** *potMCs* または*outMCs Buffer Size*のオーバーフローを防ぐため、マイクロクラスタの更新が停止される場合にTRUEを返します。
- **Last Event:**最後にクラスタインデックスが変更されたサイクルのタイムスタンプ。
- **Last Switch:**マイクロクラスタの更新と非更新が交互に繰り返された最後のサイクルのタイムスタンプ（入力*Update Micro Cluster*をTRUEに設定するか、*potMCs/outMCs Buffer Size*のオーバーフローを内部的に防止することにより発生します）。
- **Number of potMCs:**現在存在する潜在的なマイクロクラスタの数を示します。
- **Number of outMCs:**現在する外れ値マイクロクラスタの数を示します。
- **Cluster Index:**DBSCANアルゴリズムが現在のサイクルのデータポイントに対して出力するクラスタインデックスを指定します。
- **Number of Clusters:** DBSCANアルゴリズムによって検出されたクラスタの総数を指定します。

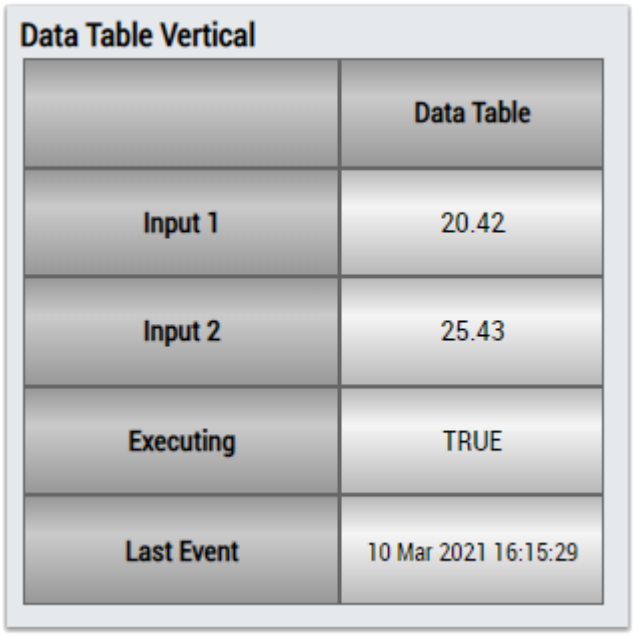
HMIコントロール

DenStream アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.DenStreamコントロールは、チャート内の入力と、クラスタの分類（cluster index）を色で表示します。ボタンを使用して、表示したい入力を一度に最大2つまで選択できます。右側のスライダーには、全ての出力値が表示されます：cluster count, number of potential micro-clusters, number of outlier micro-clusters, cluster index, last event, last switch。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：cluster count, number of potential micro-clusters, number of outlier micro-clusters, cluster index, last event, last switch。

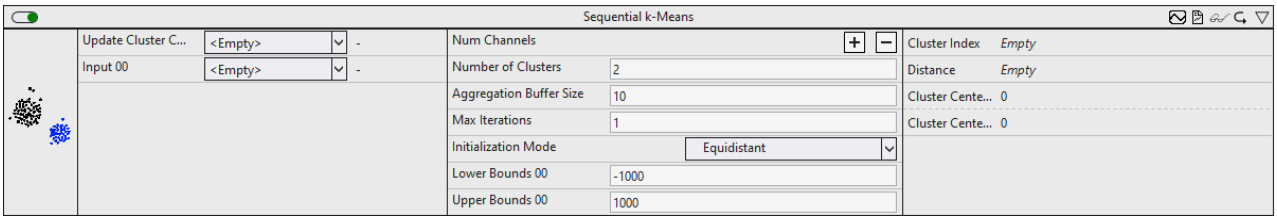


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをDenStreamアルゴリズムにマッピングすることもできます。

6.5.3.2 Sequential k-Means



*Sequential k-Means*アルゴリズムは、同名の教師なしクラスタリングアルゴリズムの実装です。これは、ストリーミングデータ用に広く使われているk-Meansクラスタリングアルゴリズムの逐次処理用の変形です。アルゴリズムの目的は、データの構造に基づいてクラスタを検出することです。各クラスタには類似したデータポイントが含まれ、異なるデータポイント同士は分離されます。

このアルゴリズムの入力チャンネル数（以下、*n*と表記）は、ユーザが自由に選択できます。これらの入力によって、クラスタが検出されるn次元の特徴空間が形成されます。各解析サイクルで、データストリームはアルゴリズムに、この特徴空間内のデータポイントとして解釈できる新しい特徴ベクトルを提供します。この特徴空間内で互いに近いデータポイントは、同じクラスタに割り当てられます。存在できるクラスタ数は、解析開始前にユーザが設定する必要があります。クラスタ数は固定されます。

従来のバッチ解析のためのk-Meansアルゴリズムとは対照的に、*Sequential k-Means*用のデータは、解析時に全て利用することはできません。代わりに、データポイントはストリーミングデータ形式で次々と到着します。これらは順次処理され、最も近いクラスタに割り当てられます。このアプローチの結果、複数の差異が生じますが、うち2つは、特にアルゴリズムの使用とパラメータ設定に関連しています。

一方、バッチ解析では、すべてのデータポイント、ひいては各特徴量の値域が解析開始時点で既に利用可能であるのに対し、逐次解析ではそうではないため、値域があらかじめ固定されているとは限りません。ただし、事前に入力チャンネルの値域を把握しておくことは、実際の値が解析中にしか得られない場合にも役立ちます。これは、クラスタ中心の初期化において特に重要です。初期化には3つのアプローチがあります。中心点はパラメータ配列で特定の値（*values)*を指定できます。あるいは、中心点を定義した値の範囲内でランダムに（*randomly*）、または等間隔に（*equidistantly*）設定することもできます。初期化モード*Random*と*Equidistant*では、値の範囲が必要です。各入力チャンネルのパラメータ*Lower Bounds*と*Upper Bounds*から設定する必要があります。

一方、バッチ解析では、クラスタ中心がわずかしか変化しなくなるまで、通常、すべてのデータポイントを複数回走査して更新します。これは逐次解析の枠組みでは不可能です。ただし、クラスタ中心を調整し、データポイントを複数回走査できるようにするため、*Sequential k-Means*アルゴリズムは、限られた数の値を一時的に保存できる*Aggregation Buffer*と呼ばれるバッファリングメカニズムを持っています。バッファを埋める際、入力された全てのデータポイントは最も近いクラスタに割り当てられます。データポイントとクラスタ中心間の距離は、ユークリッドノルムによって決まります。バッファが満たされた場合のみ、新しく割り当てられたデータポイントに基づいてクラスタ中心が更新されます。新しいクラスタ中心は、クラスタに含まれる全てのデータポイントの平均値に対応します。これはインクリメンタルに計算できるので、古いデータポイントは計算に必要ありません。バッファのサイズは、*Aggregation Buffer Size*のパラメータで設定します。パラメータ*Max Iterations*は、バッファの反復回数を指定するために使用します。デフォルト値は1です。値が2に設定されている場合、クラスタ中心の最初の調整後に、バッファ内のデータポイントはクラスタに再割り当てされ、その後、クラスタ中心が再調整されます。クラスタ中心が移動するため、個々のデータポイントは反復するごとに異なるクラスタに割り当てられる可能性があります。2サイクル間のデータ処理の計算能力が限られているため、パラメータ*Aggregation Buffer Size*と *Max Iterations* の値を過度に大きくすることは避けるべきです。大きくしすぎるとクラスタ中心の更新が保証されない可能性があります。パラメータが大きな値ではクラスタ中心が更新されず、より小さな値で更新される場合、設定したパラメータ値に対して計算能力が不十分であり、より小さな値を選択すべきであることを示しています。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

入力値

- **Update Cluster Centers:**TRUE の場合、各クラスタ中心は入力データによって更新されます。FALSEの場合、クラスタ中心は更新されず、入力データポイントのクラスタインデックスを決定するためにのみ使用されます。
- **Input 01, ..., Input 0n:**これらの入力は、クラスタリングが実行されるn次元特徴空間を形成します。

設定オプション

- **Num Channels :** 入力チャンネル数を決定します。
- **Number of Clusters:**クラスタ数を指定します。
- **Aggregation Buffer Size:**集約バッファのサイズを指定し、クラスタ中心が更新されるサイクル数を指定します。これらのサイクルの入力値は、内部（集約バッファ）に保存されます。このパラメータのデフォルト値は10です。
- **Max Iterations:**集約バッファ内の値を反復処理する頻度を指定します。このパラメータのデフォルト値は1です。
- **Initialization Mode:**クラスタ中心の初期化方法を指定します：
  - **Random:**クラスタ中心は*Lower Bounds*と *Upper Bounds*で設定された範囲内で、ランダムに設定されます。
  - **Equidistant:**クラスタ中心は、*Lower Bounds*と *Upper Bounds*で定義された値の範囲内で等間隔に分散されます。
  - **Values:**クラスタ中心は、配列*Initial Cluster Centers* で設定された値で初期化されます。
- **Initial Cluster Centers:** *initialization mode Values*では、クラスタ中心の初期値を設定します。個々のクラスタ値は、行ごとに設定されます。つまり、行列の行数が*Number of Clusters*（クラスタ数）に相当し、列数が*Number of Channels*（チャンネル数）に相当します。最初の行には、各入力チャンネルの最初のクラスタの値というように以下も同様に続きます。

- **Lower Bounds:** *Random*と*Equidistant* モードでは、各入力チャンネルの下限が設定されます。
- **Upper Bound:** *Random*と*Equidistant* モードでは、各入力チャンネルの上限が設定されます。

出力値

- **Cluster Index:**最後のサイクルのデータポイントに割り当てられたクラスタインデックスを指定し、対応する割り当てクラスタを示します。
- **Distance:**データポイントと割り当てられたクラスタ中心とのユークリッド距離を指定します。
- **Cluster Centers:** 全クラスタのクラスタ中心を一行ずつ出力します。これは、*Number of Clusters*（クラスタ数）×*Number of Channels*（チャンネル数）の行列に相当します。

HMIコントロール

*Sequential k-Means*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

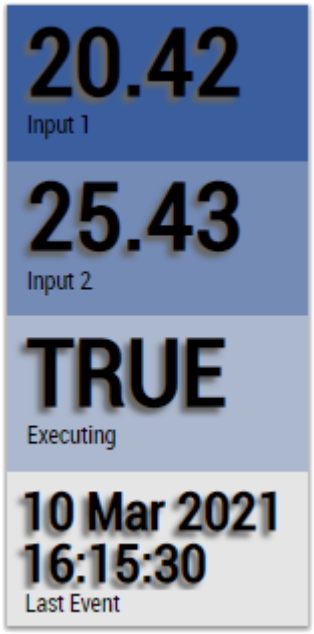
1.TableコントロールまたはMultivalueコントロールは、出力値を表示します：Lower Bounds, Upper Bounds and Initial Centers。

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

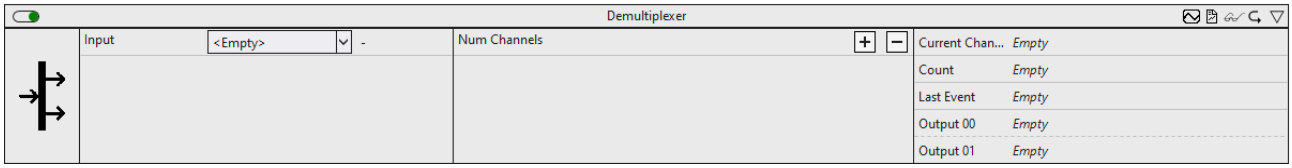


また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMI コントロールを*Sequential k-Means*アルゴリズムにマッピングすることもできます。

6.5.4 Analytics - Compare（解析 - 比較）

*Analytics-Compare*カテゴリのアルゴリズムは、比較解析や論理演算のための機能を提供します。

6.5.4.1 Demultiplexer（デマルチプレクサ）



デマルチプレクサは、入力値に基づいて出力チャンネルを選択します。このため、入力値は整数として解釈されます。この値は出力チャンネルに対応します。値が設定されたチャンネル数以外の場合、出力チャンネルは0に設定されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **NumChannels :** 出力チャンネルを追加または削除します。

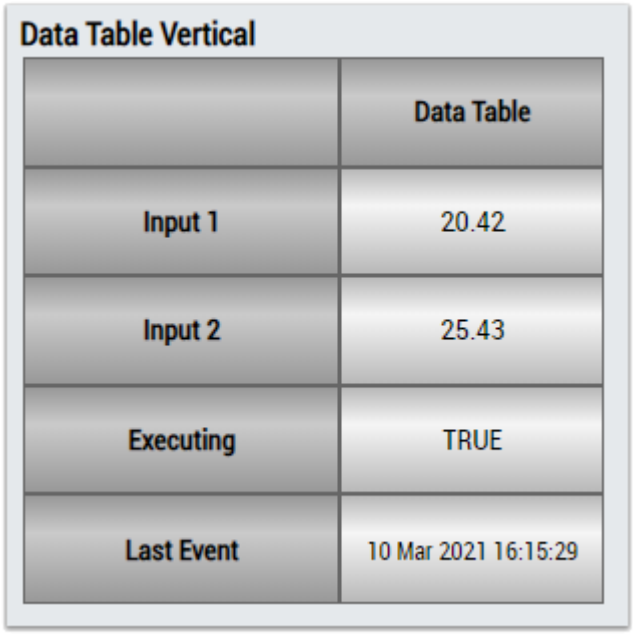
出力値

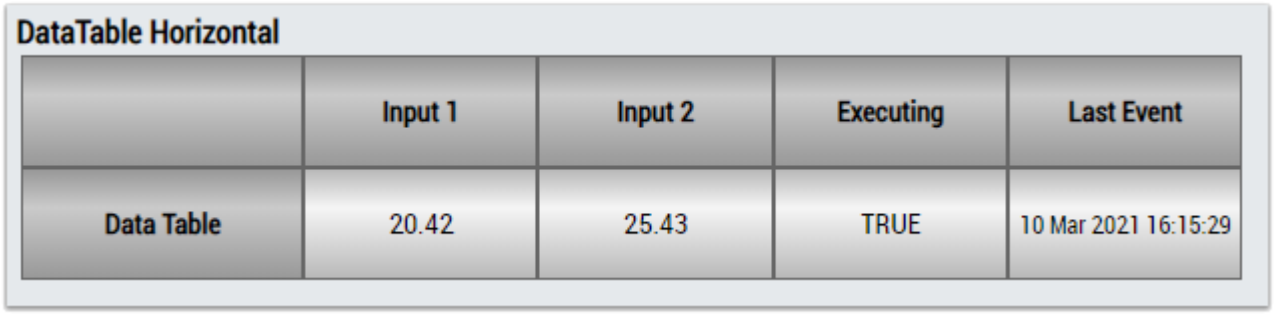
- **Current Channel:**選択されているチャンネルの番号を示します。選択したチャンネルが設定されたチャンネル以外の場合、値は0になります。
- **Count:**解析開始時に選択したチャンネルの1から始まり、他のチャンネルが選択されるたびにインクリメントします。
- **Last Event:** 最後のチャンネル変更のタイムスタンプ。
- **Out 00...n :** チャンネル0（選択チャンネル<1 または >n）～ nのブール出力

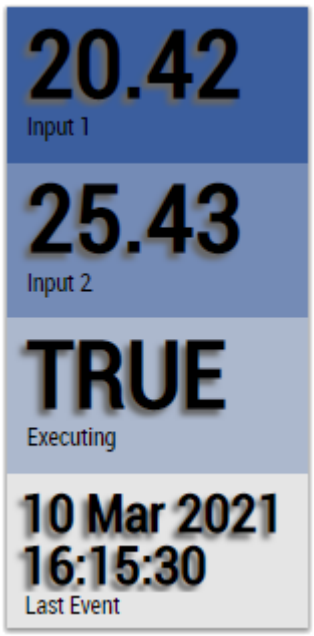
HMIコントロール

Demultiplexerアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールまたはMultivalueコントロールは、出力値Count, Current Channel and Last Event and Outを表示します。

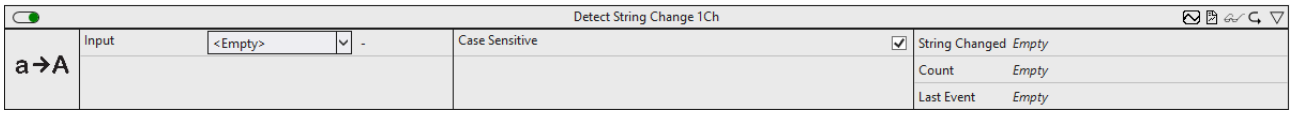






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをDemultiplexerアルゴリズムにマッピングすることもできます。

6.5.4.2 Detect String Change 1Ch



*Detect String Change 1Ch*は、文字列値の変化を検出してカウントします。大文字と小文字の区別は考慮することも、考慮しないことも可能です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Case Sensitivity:**チェックボックスをオフにすると、大文字と小文字の区別が考慮されます。オンの場合は、考慮されません。

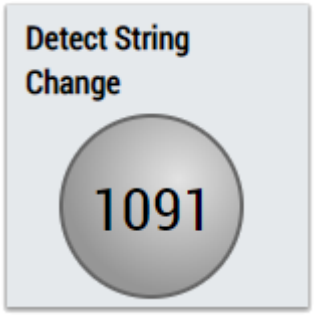
出力値

- **String Changed:**文字列の変更が検出された場合は*TRUE*、そうでない場合は*FALSE*。
- **Count:**ブールスイッチが*TRUE*になるたびにカウントアップします。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

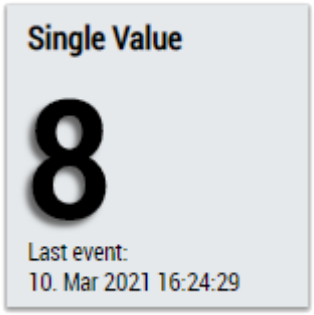
HMIコントロール

Detect String Change 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

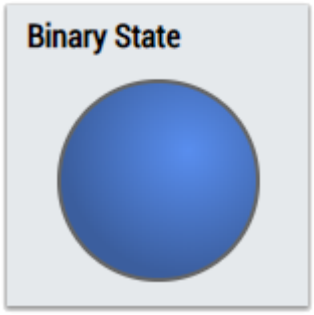
1.Comparisonコントロールは、出力値Boolean SwitchとCountを表示します。



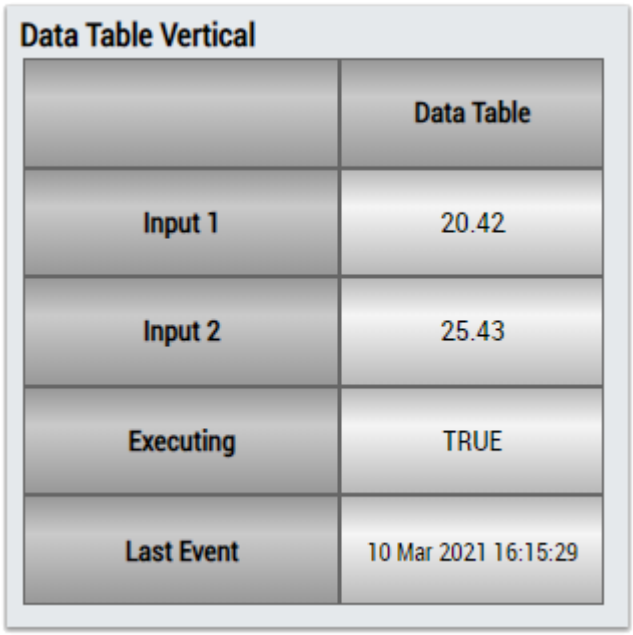
2.SingleValue コントロールは、出力値 Count と Last Event を表示します。

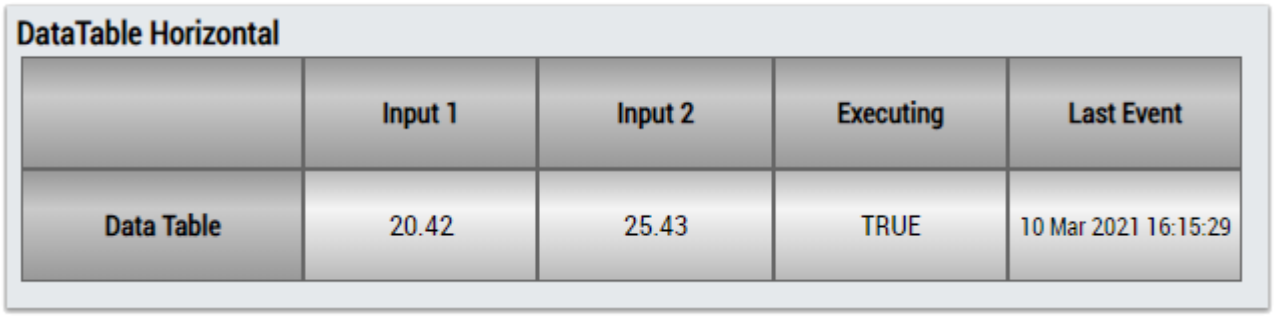


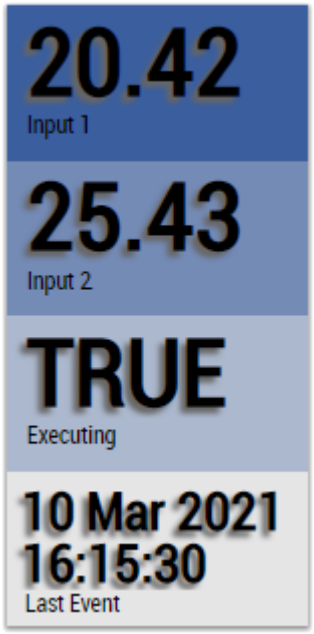
3.BinaryState コントロールは、出力値 Boolean Switch を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Operation Out, Count, Last Event。

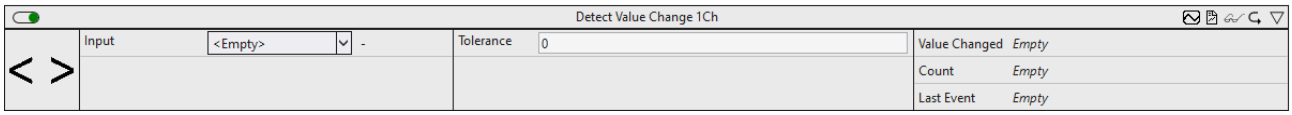






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをDetect String Change 1Chアルゴリズムにマッピングすることもできます。

6.5.4.3 Detect Value Change 1Ch



*Detect Value Change 1Ch*は、入力数値の変化を検出してカウントします。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Tolerance:**許容誤差は、最後に検出された値変化の入力値を指します。入力値がこの許容範囲を外れると、値の変化が検出されます。

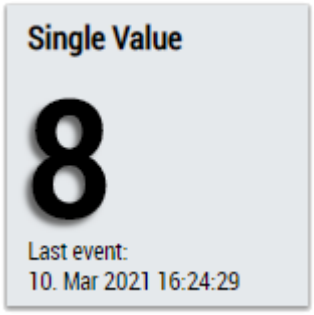
出力値

- **Value Changed:**値の変更が検出された場合は*TRUE*、そうでない場合は*FALSE*。
- **Count:**ブールスイッチが*TRUE*になるたびにカウントアップします。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

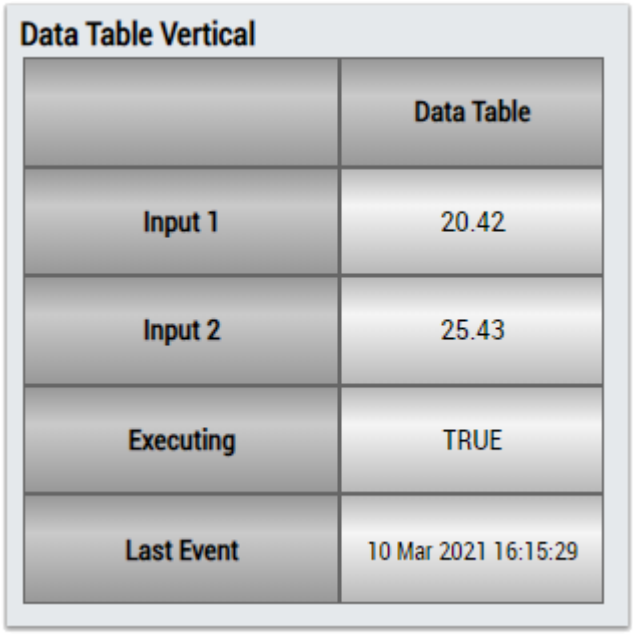
HMIコントロール

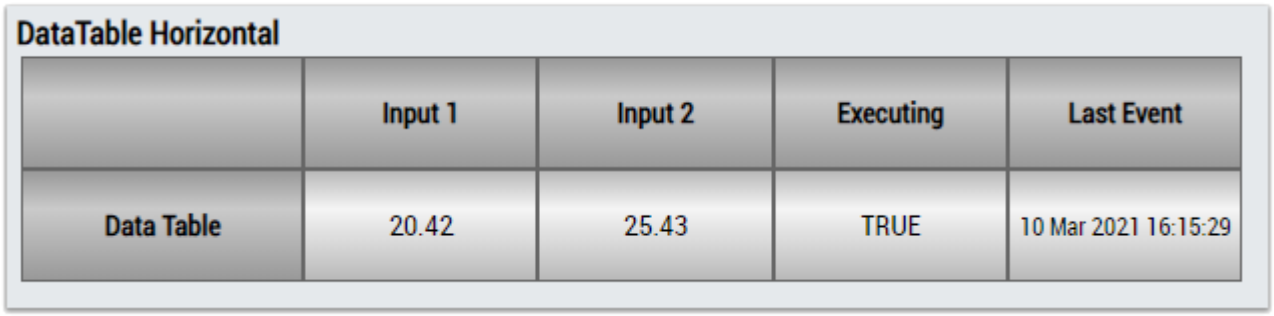
*Detect Value Change 1Ch*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

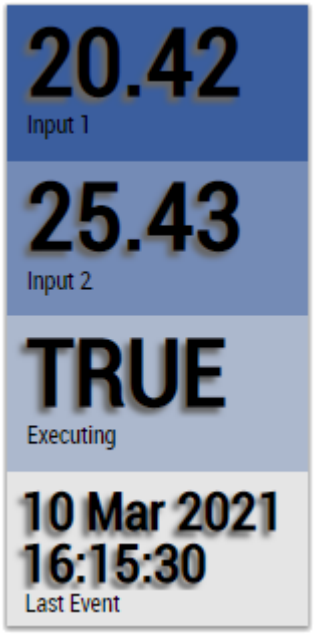
1.SingleValue コントロールは、出力値 Count と Last Event を表示します。



2.TableコントロールとMultivalueコントロールは、全ての出力値を表示します。

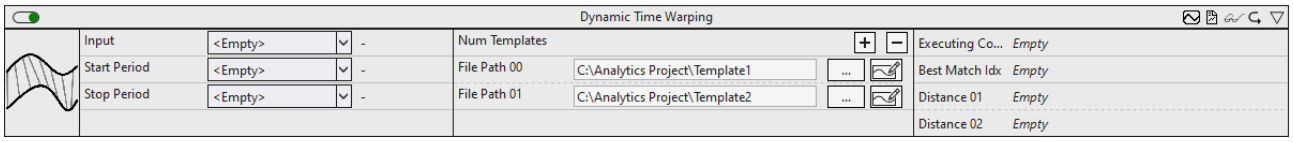






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをマッピングすることもできます。

6.5.4.4 Dynamic Time Warping



*Dynamic Time Warping*アルゴリズムは、入力データを過去に記録されたテンプレートと比較します。このアルゴリズムの特徴は、速度の異なる信号やタイミングがずれた信号でも比較できることです。その結果、入力信号と各テンプレートとの距離が出力されます。距離が小さいほど、比較した信号の一致度は高くなります。距離が0であれば、両方の信号は同一です。距離の大きさは、信号の一致度だけでなく、信号の長さにも依存します。

比較は、Start Period（期間スタート）フラグの信号が*TRUE*になったときに開始されます。Stop Period（期間ストップ）フラグの信号が TRUE であるか、再び TRUE になった場合に、結果が出力されます。

ファイルへの同時アクセスが発生するため、*Dynamic time Warping*を Time Based Teach Path 1Ch [▶ 253]と同時に使用することは推奨しません 。代わりに、まず基準信号をTime Based Teach Path 1Ch [▶ 253] に教示してから、*Dynamic time Warping*を用いて評価を行うべきです。テンプレートには、Time Based Teach Path 1Ch [▶ 253] で過去に記録された基準信号が含まれています。原則として、テンプレートは数百個のサポートポイント（支持点）で構成されます。そのため、ファンクションブロック（Downsampling 1Ch [▶ 88] ）によるデータの削減が役立ちます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Templates :** 比較用のテンプレートを追加または削除します。
- **File Path 01 ... n:**過去に記録したテンプレートへのパス。

出力値

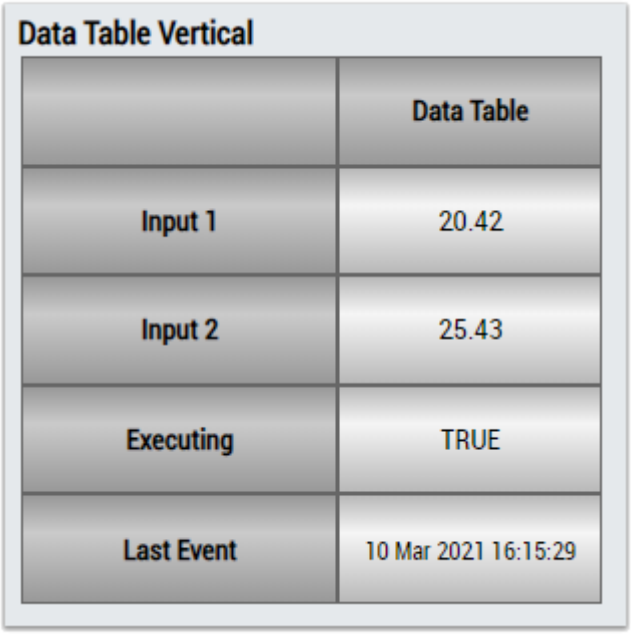
- **Executing Compare:** アルゴリズムが入力値を処理している場合は*TRUE* を表示し、そうでない場合は*FALSE* を表示します。Start Period（期間スタート）フラグが*TRUE*になったときに処理を開始し、Stop Period（期間ストップ）フラグがTRUEになったときに処理を終了します。
- **Best Match Idx :** 入力チャンネルとの距離が最も小さいテンプレートのインデックスを出力します。

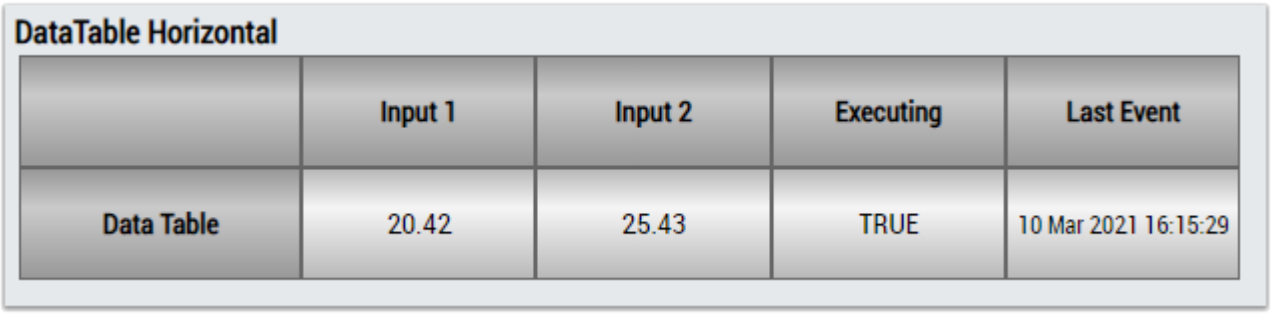
- **Distance 01 ... n :** 入力信号と各テンプレートとの距離を指定します。距離が小さいほど、比較した信号の一致度は高くなります。

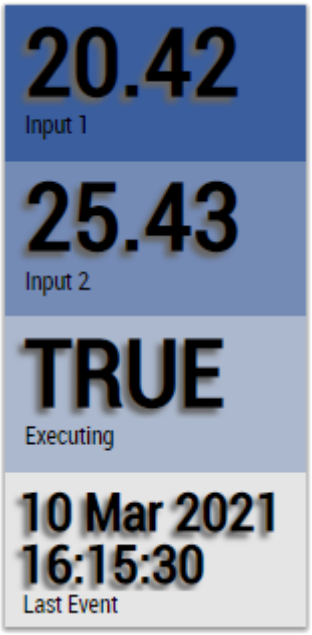
HMIコントロール

*Dynamic Time Warping*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールとMultivalueコントロールは、以下、全ての出力値を表示します：*Executing Compare*、*Best Match Idx*、*Distance*。

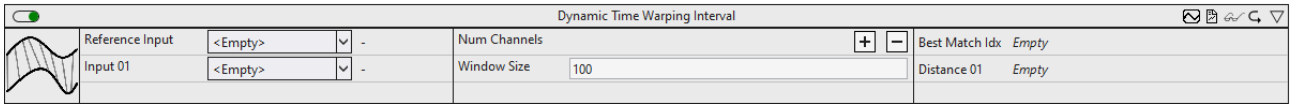






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Dynamic Time Warping*アルゴリズムにマッピングすることもできます。

6.5.4.5 Dynamic Time Warping Interval



*Dynamic Time Warping Interval*アルゴリズムは、複数の入力データを相互に比較します。このアルゴリズムの特徴は、速度の異なる信号やタイミングがずれた信号でも比較できることです。比較のために考慮されるのは、設定されたウィンドウの信号区間のみです。新しい結果はウィンドウが期限切れになった後に出力されます。その結果、基準信号と各入力信号との距離が出力されます。距離が小さいほど、比較した信号の一致度は高くなります。距離が0であれば、両方の信号は同一です。距離の大きさは、信号の一致度だけでなく、信号の長さにも依存します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 入力チャンネルを追加または削除します。
- **Window Size:**計算が行われるサイクル数を指定します。アルゴリズムに必要なメモリは、このパラメータに比例します。

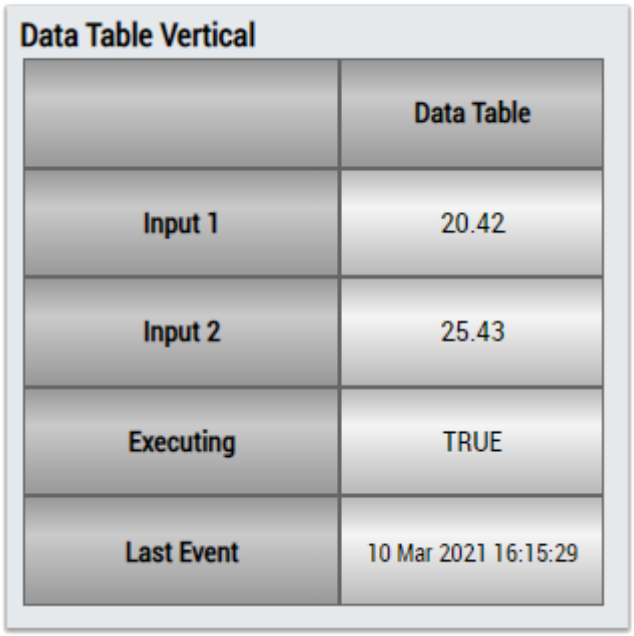
出力値

- **Best Match Idx:** 参照チャンネルとの距離が最も小さい入力チャンネルのインデックスを出力します。
- **Distance 01 ... n:** 基準信号と各入力チャンネル間の距離を指定します。距離が小さいほど、比較した信号の一致度は高くなります。

HMIコントロール

*Dynamic Time Warping Interval*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールとMultivalueコントロールは、以下、全ての出力値を表示します：*Best Match Idx*、*Distance*。

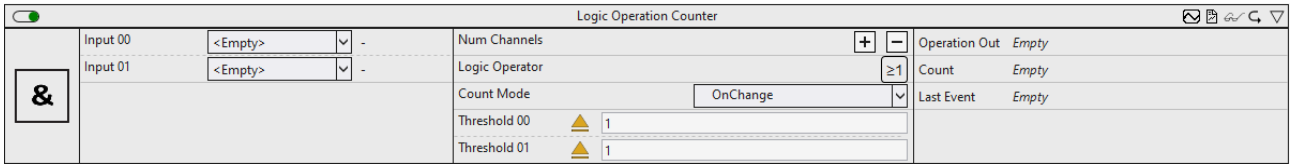


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Dynamic Time Warping Interval*アルゴリズムにマッピングすることもできます。

6.5.4.6 Logic Operation Counter



*Logic Operation Counter*は、2つ以上のチャンネルの値に対して論理演算を実行し、結果を提供します。したがって、各入力値は閾値と演算子と組み合わせることができます。さらに、論理演算子とカウントモードは個別に設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 入力チャンネルを追加または削除します。
- **Threshold 00:** 第1チャンネルの信号の閾値。
- **Threshold 01 ..n:** 2番目からn番目のチャンネルの信号の閾値。
- **Logic Operator :** 論理演算子：

  ≥1 論理 OR

論理 XOR（排他的論理和）

論理 AND

論理 NAND（NOT AND）

論理 NOR（NOT OR）

- **Count Mode:**結果のカウンタモード。
  OnChange：結果がTRUEに変わるたびにカウントします。
  Cyclic：条件が*TRUE*のとき、1サイクルごとにインクリメントします。
- **Use Absolute Values :** チェックボックスをオンにすると、入力信号の絶対値が使用されます。
- **Tolerance (オプション):**Equal / NotEqual比較時に使用される許容値。

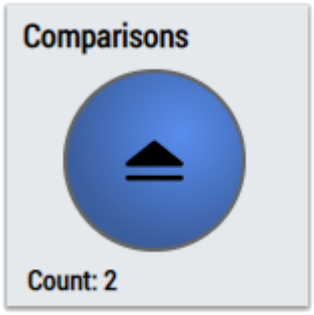
出力値

- **Operation Out:**論理演算の結果。
- **Count:**出力値がTRUEになるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

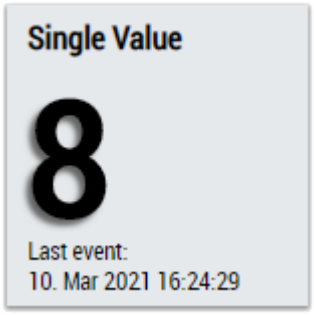
HMIコントロール

Logic Operation Counterアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

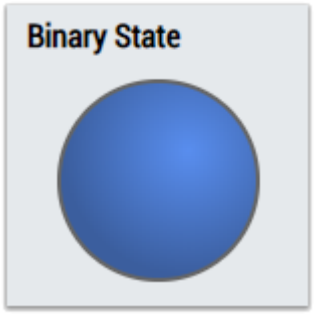
1.Comparisonコントロールは、出力値Operation OutとCount、および設定オプションLogic Operatorを表示します。



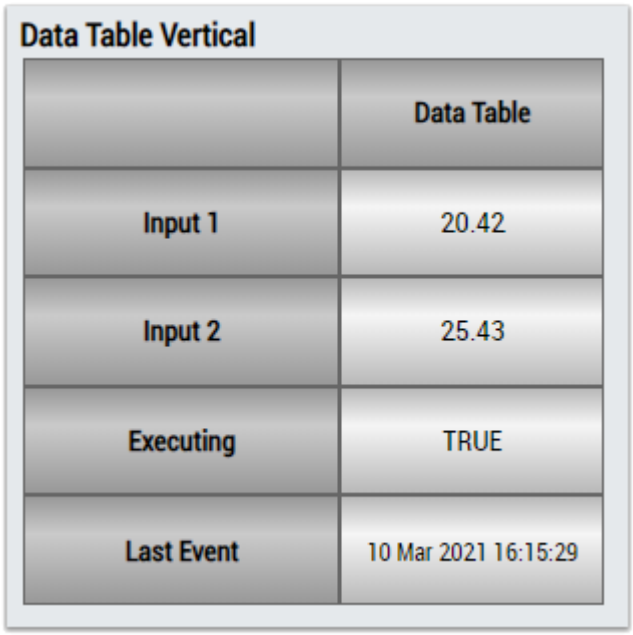
2.SingleValue コントロールは、出力値 Count と Last Event を表示します。

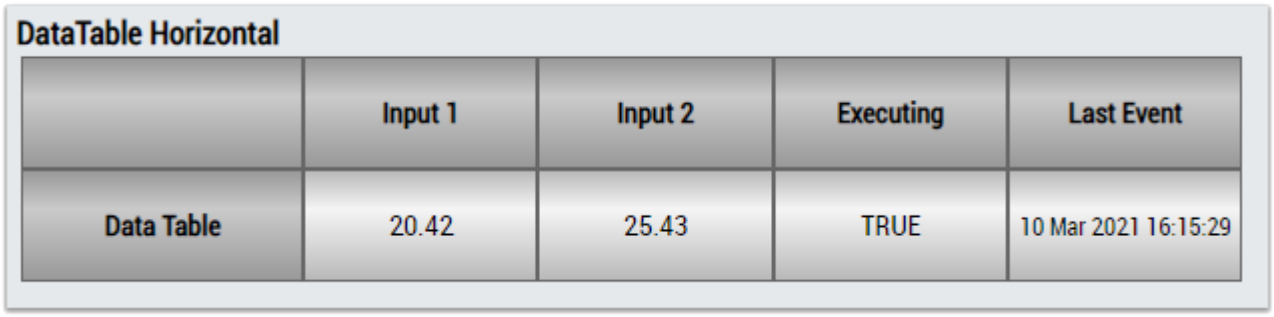


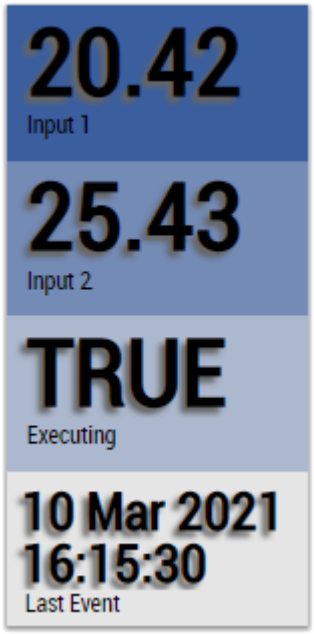
3.BinaryState コントロールは、出力値 Operation Out を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Operation Out, Count, Last Event。

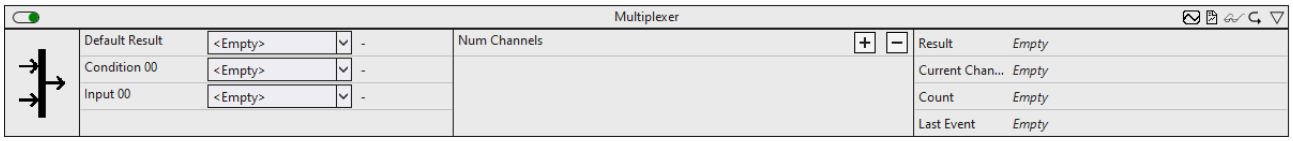






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをLogic Operation Counterアルゴリズムにマッピングすることもできます。

6.5.4.7 Multiplexer



Multiplexerは、複数の入力チャンネルから1つのチャンネルを選択します。各入力チャンネルには、ブーリアン入力を追加する必要があります。出力は、条件入力がTRUEの最初の入力チャンネルに対応します。チャンネルの優先順位は、設定した順です。どのチャンネルも条件を満たさない場合、指定したデフォルトチャンネルが返されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** チャンネル数。各チャンネルに2つの入力変数があります：1つはブール値の条件（*Condition0n*）で、もう1つは任意のデータ型の入力値（*Input0n*）です。条件を満たした場合に*Result*に渡されます。

出力値

- **Result:**選択した入力チャンネルの信号*Input0n* を出力します。
- **Current Channel:**選択されているチャンネルの番号を示します。デフォルトの結果を選択した場合、値は0となります。入力チャンネルには、設定順に番号が振られています。
- **Count:**解析開始時に選択したチャンネルの1から始まり、他のチャンネルが選択されるたびにインクリメントします。
- **Last Event:** 最後のチャンネル変更のタイムスタンプ。

HMIコントロール

Multiplexerアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

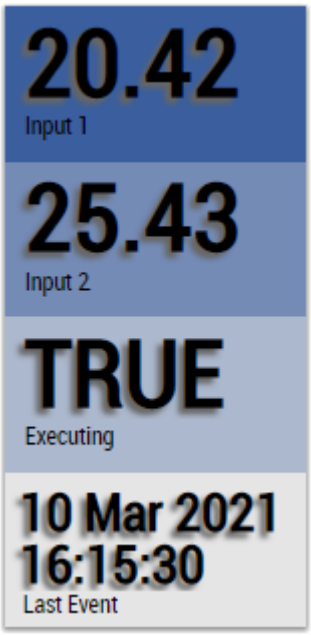
1.TableコントロールまたはMultivalueコントロールは、出力値 Count, Result, Current Channel and Last Eventを表示します。

Data Table Vertical

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

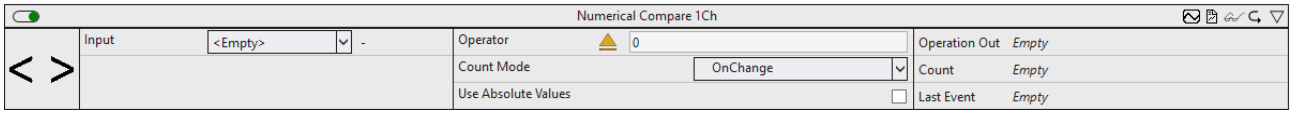
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをMultiplexerアルゴリズムにマッピングすることもできます。

6.5.4.8 Numerical Compare 1Ch



*Numerical Compare 1Ch*は、入力値を基準値と比較し、比較演算の結果を提供します。演算子、基準値、カウントモードは個別に設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が基準値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Reference :** 比較演算の基準値。
- **Count Mode:**結果のカウンタモード。
  OnChange : 結果がTRUEに変わるたびにカウントします。
  Cyclic : 条件が*TRUE*のとき、1サイクルごとにインクリメントします。
- **Use Absolute Values :** チェックボックスをオンにすると、入力信号の絶対値が使用されます。
- **Tolerance (オプション):**Equal / NotEqual比較時に使用される許容値。

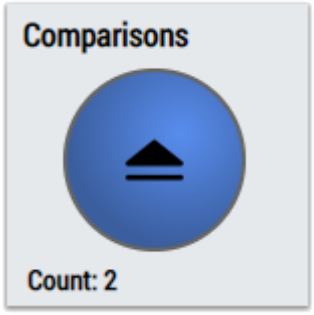
出力値

- **Operation Out:**比較演算の結果。
- **Count:**出力値がTRUEになるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

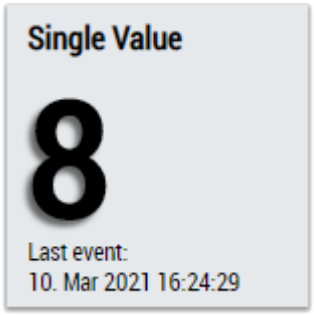
HMIコントロール

Numerical Compare 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

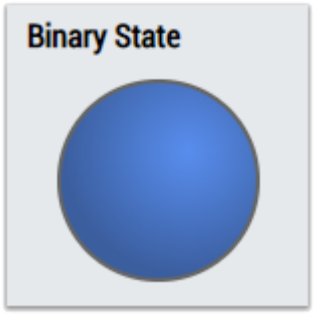
1.ComparisonコントロールまたはMultivalueコントロールは、出力値Operation OutとCount、および設定オプションOperatorを表示します。



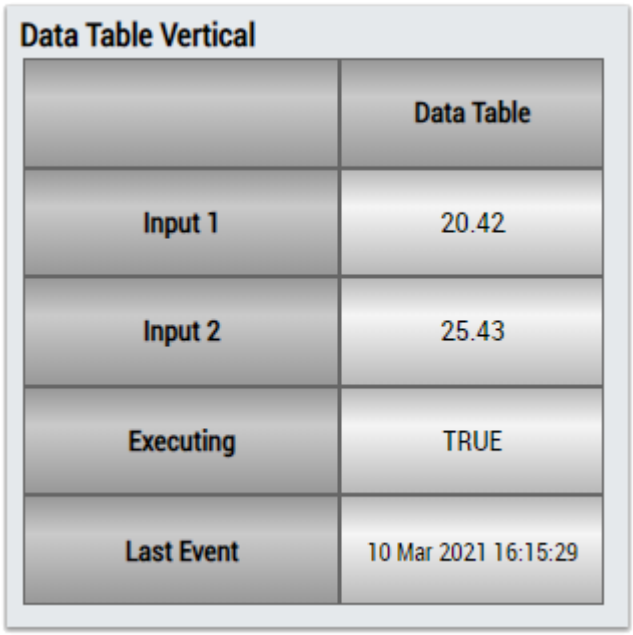
2.SingleValue コントロールは、出力値 Count と Last Event を表示します。

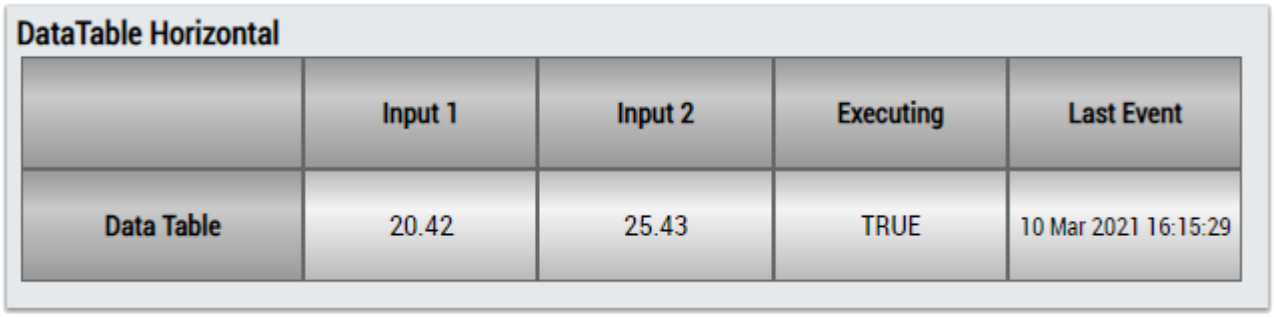


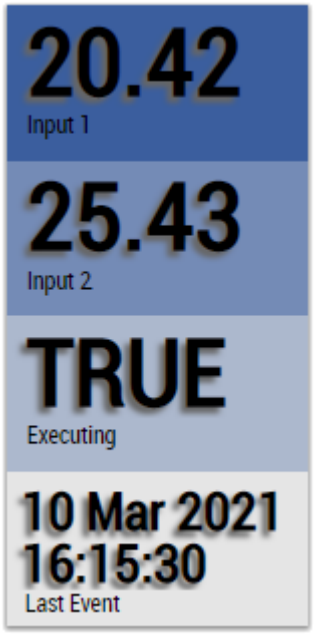
3.BinaryState コントロールは、出力値 Operation Out を表示します。



4.TableコントロールとMultivalueコントロールは、以下、全ての出力値を表示します：Operation Out, Count, Last Event。

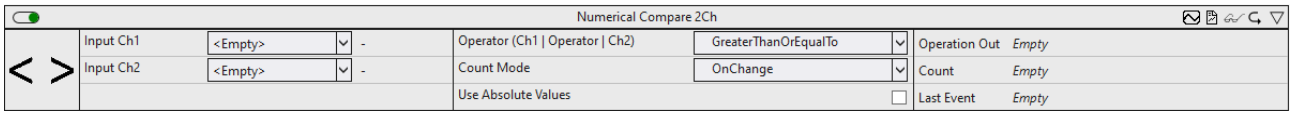






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Numerical Compare 1Ch*アルゴリズムにマッピングすることもできます。

6.5.4.9 Numerical Compare 2Ch



*Numerical Compare 2Ch*は、第1チャンネルの入力値と第2チャンネルの入力値を比較し、その結果を提供します。演算子とカウントモードは個別に設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が基準値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Count Mode:**結果のカウンタモード。
  OnChange：結果がTRUEに変わるたびにカウントします。
  Cyclic：条件が*TRUE*のとき、1サイクルごとにインクリメントします。
- **Use absolute values :** チェックボックスをオンにすると、入力信号の絶対値が使用されます。
- **Tolerance (オプション):**Equal / NotEqual比較時に使用される許容値。

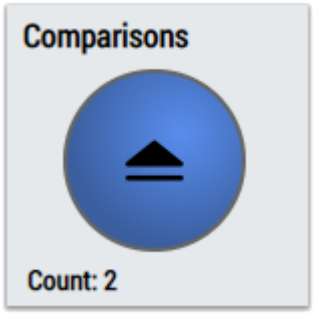
出力値

- **Operation Out:**比較演算の結果。
- **Count:**出力値がTRUEになるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

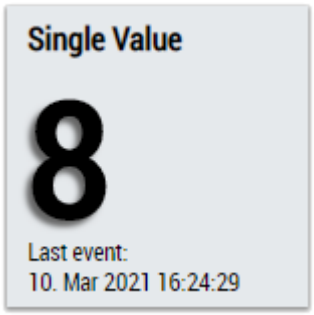
HMIコントロール

Numerical Compare 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

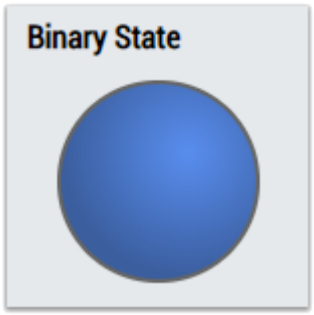
1.ComparisonコントロールまたはMultivalueコントロールは、出力値Operation OutとCount、および設定オプションOperatorを表示します。



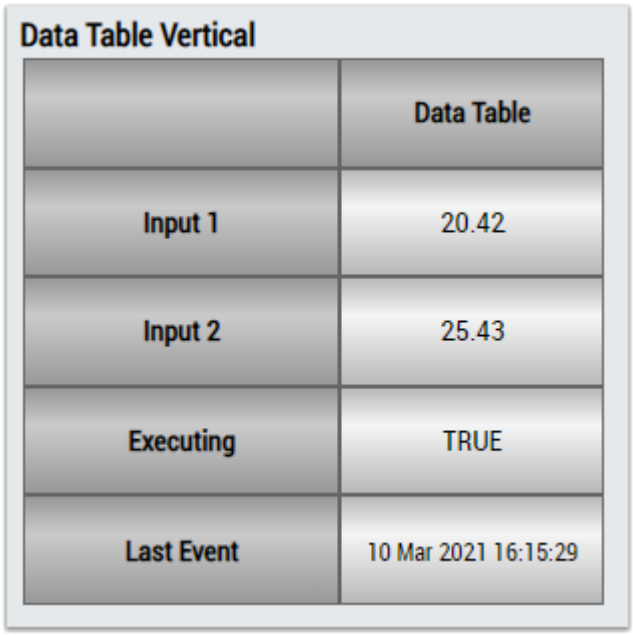
2.SingleValue コントロールは、出力値 Count と Last Event を表示します。



3.BinaryState コントロールは、出力値 Operation Out を表示します。



4.Tableコントロールは、以下、全ての出力値を表示します：Operation Out, Count, Last Event。

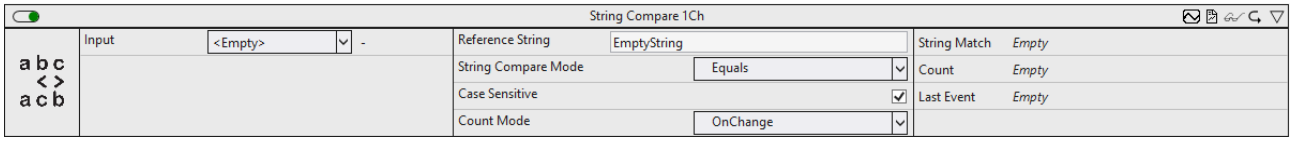


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをNumerical Compare 1Chアルゴリズムにマッピングすることもできます。

6.5.4.10 String Compare 1Ch



*String Compare 1Ch*は、入力文字列と参照文字列を比較し、一致した文字列をカウントします。そのため、大文字と小文字を区別したり、カウントモードを変更したりできます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Reference String:** 比較用の参照文字列。
- **String Compare Mode**：異なる文字列の比較モード用の列挙。
  Equals：入力文字列は参照文字列に等しい。
  BeginsWith：入力文字列は参照文字列から始まる。
  Contains：入力文字列は参照文字列を含む。
- **Case Sensitivity:**チェックボックスがチェックされている場合、入力は大文字と小文字を区別します。チェックされていない場合、入力は大文字と小文字を区別しません。
- **Count Mode:**結果のカウンタモード。
  OnChange：結果がTRUEに変わるたびにカウントします。
  Cyclic：条件が*TRUE*のとき、1サイクルごとにインクリメントします。

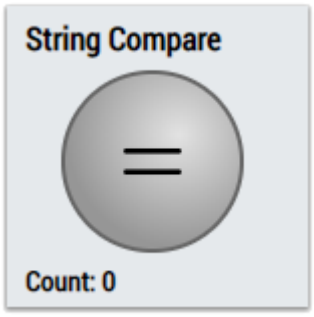
出力値

- **String Match:**入力文字列が参照文字列と一致する場合に*TRUE*、一致しない場合に*FALSE*。
- **Count:**出力値がTRUEになるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

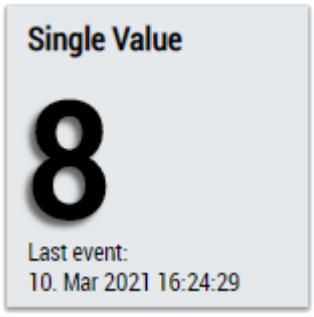
HMIコントロール

String Compare 1Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

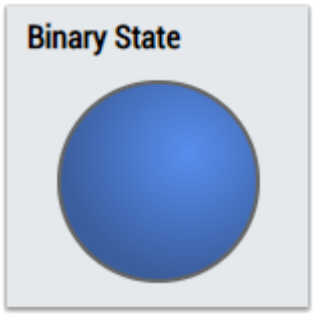
1.Comparisonコントロールは、出力値String MatchとCount、および設定オプションString Compare Modeを表示します。



2.SingleValue コントロールは、出力値 Count と Last Event を表示します。



3.BinaryState コントロールは、出力値 String Match を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：String Match, Count, Last Event.。

Data Table Vertical

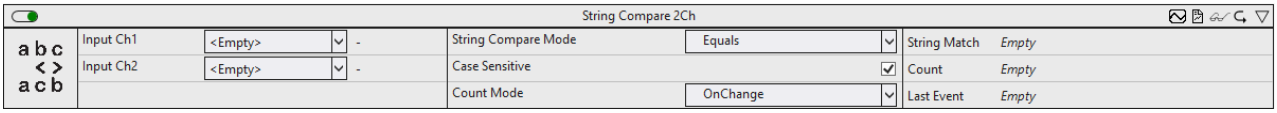
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをString Compare 1Chアルゴリズムにマッピングすることもできます。

6.5.4.11 String Compare 2Ch



*String Compare 2Ch*は、最初の入力文字列の値と2番目の文字列の値を比較し、一致した文字列をカウントします。そのため、大文字と小文字を区別したり、カウントモードを変更したりできます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **String Compare Mode**：異なる文字列の比較モード用の列挙。
  Equals：入力文字列は参照文字列に等しい。
  BeginsWith：入力文字列は参照文字列から始まる。
  Contains：入力文字列は参照文字列を含む。
- **Case Sensitivity:**チェックボックスがチェックされている場合、入力は大文字と小文字を区別します。チェックされていない場合、入力は大文字と小文字を区別しません。
- **Count Mode:**結果のカウンタモード。
  OnChange：結果がTRUEに変わるたびにカウントします。
  Cyclic：条件が*TRUE*のとき、1サイクルごとにインクリメントします。

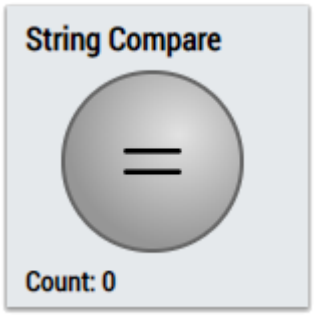
出力値

- **String Match:**最初の入力文字列の値が2番目の入力文字列の値と一致する場合は*TRUE*、そうでない場合は*FALSE*。
- **Count:**出力値がTRUEになるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

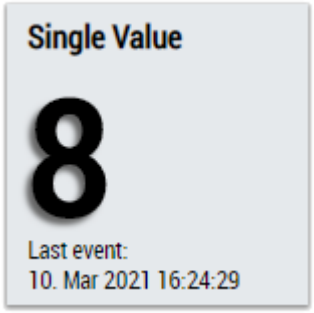
HMIコントロール

String Compare 2Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

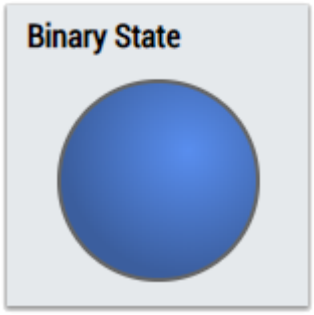
1.Comparisonコントロールは、出力値String MatchとCount、および設定オプションString Compare Modeを表示します。



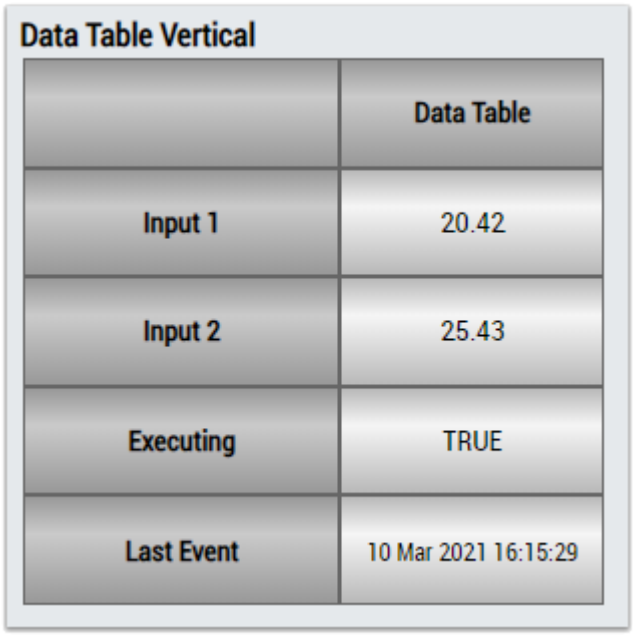
2.SingleValue コントロールは、出力値 Count と Last Event を表示します。

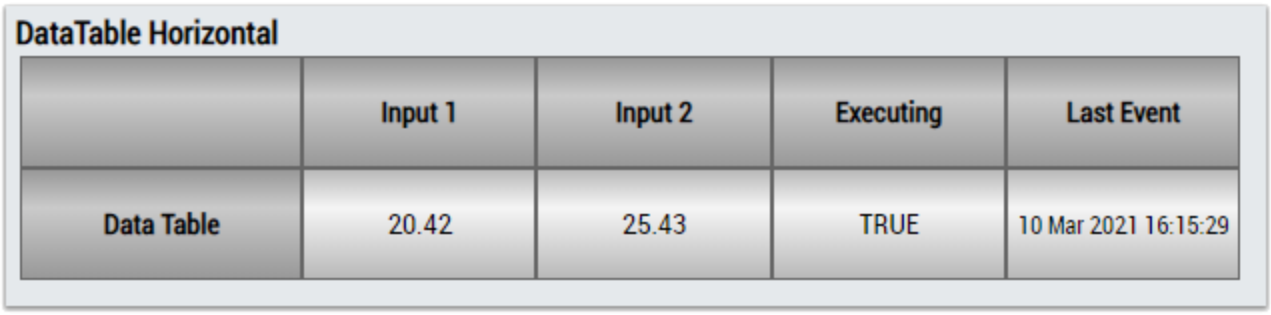


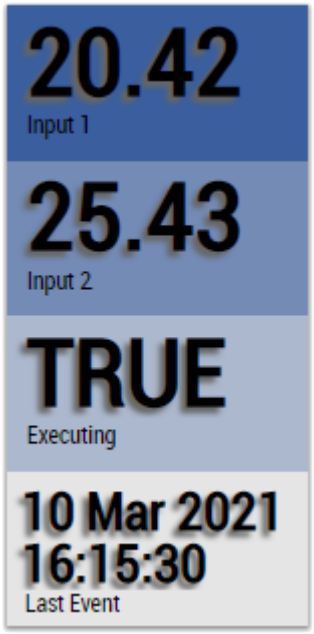
3.BinaryState コントロールは、出力値 String Match を表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：String Match, Count, Last Event.。

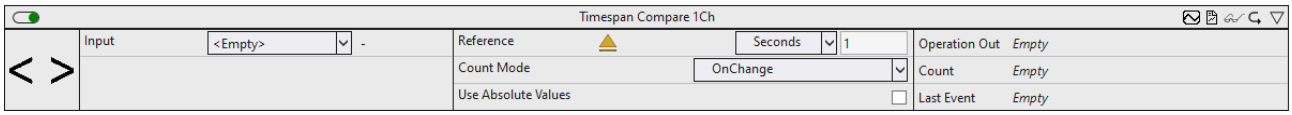






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをString Compare 2Chアルゴリズムにマッピングすることもできます。

6.5.4.12 Timespan Compare 1Ch



*Timespan Compare 1Ch*は入力値を基準値と比較し、その結果を返します。入力として1nsの分解能を持つタイムスパンがサポートされています。演算子、基準値、カウントモードは個別に設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Operator:**入力値が基準値に対して「より大きい」「以上」「以下」「より小さい」「等しくない」のいずれで比較するかを指定します。
- **Reference:**比較用の基準タイムスパン。
- **Count Mode:**結果のカウンタモード。
  OnChange：結果が*TRUE*に変わるたびにカウントします。
  Cyclic：条件が*TRUE*のとき、1サイクルごとにインクリメントします。
- **Use Absolute Values:**チェックボックスをオンにすると、入力信号の絶対値が使用されます。

出力値

- **Operation Out:**比較演算の結果。
- **Count:**出力値が*TRUE*になるとインクリメントします。動作は*Count Mode*パラメータに依存します。
- **Last Event:** 最後にトリガーされたイベントの時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

HMIコントロール

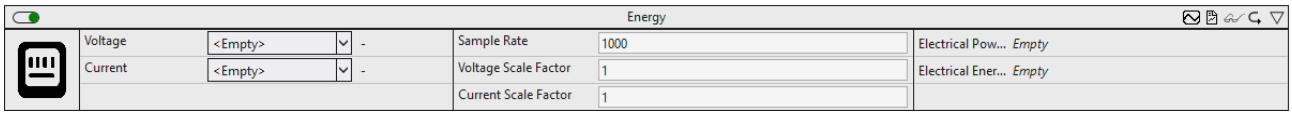
Timespan Compare 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.Tableコントロールは、以下、全ての出力値を表示します：Operation Out, Count, Last Event。

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Timespan Compare 1Ch*アルゴリズムにマッピングすることもできます。

6.5.5 Analytics - Energy（解析 - 電力）

6.5.5.1 電力



*Energy module*は、様々な電気機器の電力監視に適しています。電流と電圧の入力データは、設定した区間内で平均化され、計算されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Voltage Scale Factor:**入力電圧のスケールファクタ。
- **Current Scale Factor:**入力電流の倍率係数。
- **Interval（オプション）:**デフォルトでは、入力データは1秒間隔で平均化されます。

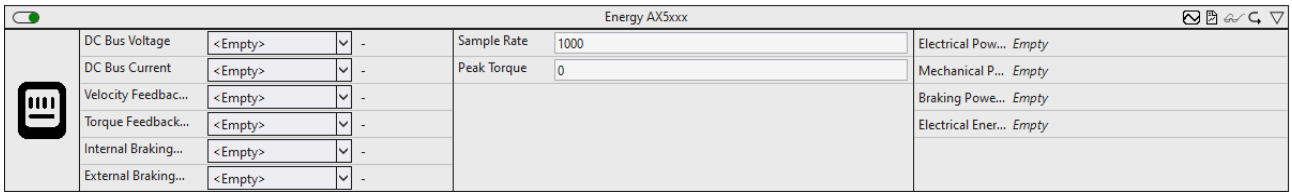
入力値

- **Voltage:**電圧の実効値（V）
- **Current:**電流の実効値（A）

出力値

- **Electrical Power [W]:**電力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.2 Energy AX5xxx



*Energy AX5xxx*モジュールは、AX5xxxシリーズのサーボドライブの電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Peak Torque:**トルクのピーク値（Nm）
  プロセスデータ：CoE：P-0-0094
- **Interval（オプション）:**デフォルトでは、入力データは1秒間隔で平均化されます。

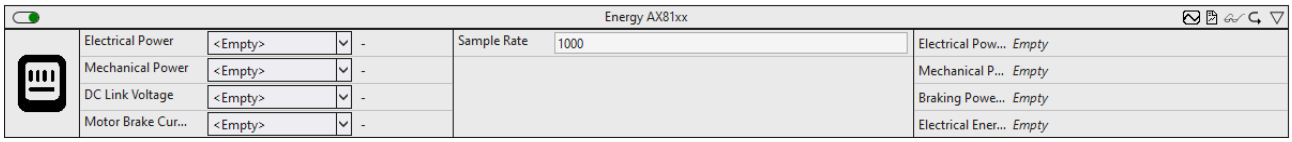
入力値

- **DC Bus Voltage:DCバス電圧（0.1V単位）**
  プロセスデータ：CoE：S-0-0016 CoE S-0-0380「DC bus voltage」をPDOコンテンツに追加します。
- **DC Bus Current:**DCリンク入力電流（mA）
  プロセスデータ：CoE：S-0-0016 CoE S-0-0381「DC bus current」をPDOコンテンツに追加します。
- **Velocity Feedback Value:**モータ速度
  プロセスデータ：CoE：S-0-0016 CoE S-0-0040「Velocity feedback 1 value」を PDO コンテンツに追加します。
- **Torque Feedback Value:**トルク発生電流
  プロセスデータ CoE：S-0-0016 CoE S-0-0084「Torque/force feedback value」をPDOコンテンツに追加します。
- **Internal Braking Resistor Power :** 内部制動抵抗器を介して出力される制動電力（W）
  プロセスデータ：CoE：S-0-0016 CoE P-0-0209「Internal braking resistor actual averaged power」をPDOコンテンツに追加します。
- **External Braking Resistor Power :** 外部制動抵抗器を介して出力される制動電力（W）
  プロセスデータ：CoE：S-0-0016 CoE P-0-0210「External braking resistor actual averaged power」をPDOコンテンツに追加します。

出力値

- **Electrical Power [W]:**電力（W）
- **Mechanical Power [W]:**機械的出力（W）
- **Braking Power [W]:**制動力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.3 Energy AX81xx



*Energy AX81xx*モジュールは、AX81xxシリーズのサーボドライブの電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

入力値

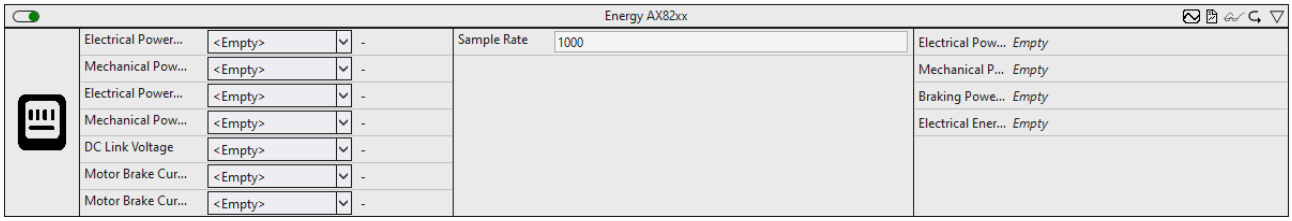
- **Electrical power:**電力（W）
  プロセスデータ：PDO入力0x1A32を有効化し、CoE 0x3242:12「Electrical power」をPDOコンテンツに追加します。
- **Mechanical Power:**機械的出力（W）
  プロセスデータ：PDO 入力 0x1A32 を有効化し、CoE 0x3242:13 「Mechanical power」を PDO コンテンツに追加します。
- **DC link voltage:**DCリンク電圧（mV）
  プロセスデータ：PDO入力0x1A00を有効化し、CoE 0x6079「DC link circuit voltage」をPDOコンテンツに追加します。

- **Motor Brake Current:**モータのブレーキ電流（A）
  プロセスデータ：PDO入力0x1A20を有効化し、CoE 0x3001:01「Actual motor brake current」をPDOコンテンツに追加します。

出力値

- **Electrical Power [W]:**電力（W）
- **Mechanical Power [W]:**機械的出力（W）
- **Braking Power [W]:**制動力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.4 Energy AX82xx



*Energy AX82xx*モジュールは、AX82xxシリーズのサーボドライブの電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

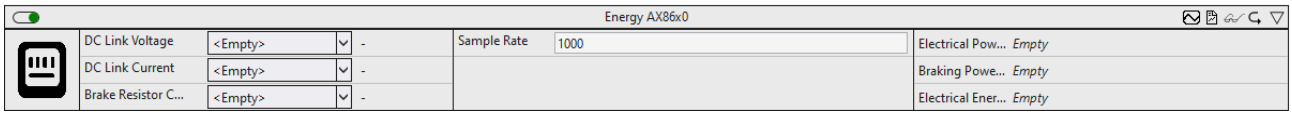
入力値

- **Electrical Power ChA:**第1チャンネルの電力（W）
  プロセスデータ：PDO 入力 0x1A32 を有効化し、CoE 0x3242:12「Electrical power」を PDO コンテンツに追加します。
- **Mechanical Power ChA:**第1チャンネルの機械的出力（W）
  プロセスデータ：PDO 入力 0x1A32 を有効化し、PDOコンテンツに CoE 0x3242:13 「Mechanical power」を追加します。
- **Electrical Power ChB:**第2チャンネルの電力（W）
  プロセスデータ：PDO 入力 0x1A52 を有効化し、CoE 0x3642:12「Electrical power」を PDO コンテンツに追加します。
- **Mechanical Power ChB:**第2チャンネルの機械的出力（W）
  プロセスデータ：PDO 入力 0x1A52 を有効化し、CoE 0x3642:13「Mechanical power」を PDO コンテンツに追加します。
- **DC link voltage:**DCリンク電圧（mV）
  プロセスデータ：PDO入力0x1A00を有効化し、CoE 0x6079「DC link circuit voltage」をPDOコンテンツに追加します。
- **Motor Brake Current ChA:**第1チャンネルのモータブレーキ電流（A）
  プロセスデータ：PDO 入力 0x1A20 を有効化し、CoE 0x3001:01「Actual motor brake current」を PDOコンテンツに追加します。
- **Motor Brake Current ChB:**第2チャンネルのモータブレーキ電流（A）
  プロセスデータ：PDO 入力 0x1A40 を有効化し、CoE 0x3401:01「Actual motor brake current」を PDOコンテンツに追加します。

出力値

- **Electrical Power [W]:**電力（W）
- **Mechanical Power [W]:**機械的出力（W）
- **Braking Power [W]:**制動力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.5 Energy AX86x0



*Energy AX86xx*モジュールは、AX86xxシリーズ電源モジュールの電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

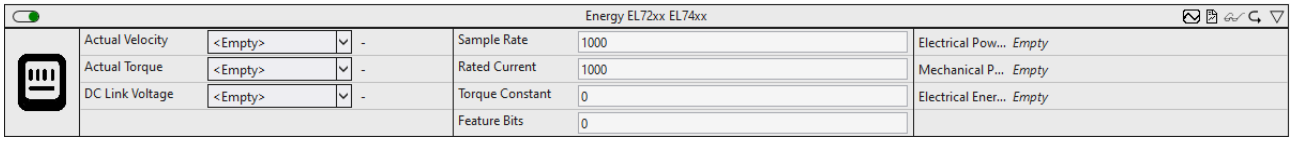
入力値

- **DC Link Voltage:**DCリンク電圧(mV)
  プロセスデータ：PDO 入力 0x1A05 を有効化します。
- **DC Link Current:**DCリンク入力電流（0.01 A単位）
  プロセスデータ：PDO 入力 0x1A6 を有効化します。
- **Brake Resistor Continuous Power:**制動抵抗器経由の電力（W）
  プロセスデータ：PDO 入力 0x1A07 を有効化します。

出力値

- **Electrical Power [W]:**電力（W）
- **Braking Power [W]:**制動力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.6 Energy EL72xx EL74xx



*Energy AX72xx EL74xx*モジュールは、EtherCATターミナル EL72xxおよびEL74xx の電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Rated Current:**定格電流（mA）
  プロセスデータ：CoE：0x8011:12

- **Torque Constant:**トルク定数（mNm/A）
  プロセスデータ：CoE: 0x8011:16
- **Feature bits:**電流タイプ特定のための機能ビット
  プロセスデータ：CoE: 0x8010:53
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

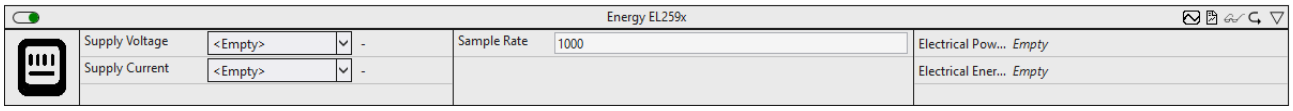
入力値

- **Actual velocity:**実際の速度値
  プロセスデータ：PDO 入力 0x1A02 を有効化します。
- **Actual Torque：**実際のトルク値
  プロセスデータ：PDO 入力 0x1A03 を有効化します。
- **DC link voltage:**DCリンク電圧（mV）
  プロセスデータ： PDO 入力 0x1A04 を有効化し、CoE 0x8010:39 で「DC link voltage (mV)」を選択します。

出力値

- **Electrical Power [W]:**電力（W）
- **Mechanical Power [W]:**機械的出力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.7 Energy EL259x



*Energy EL259x*モジュールは、EtherCATターミナル EL259x の電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

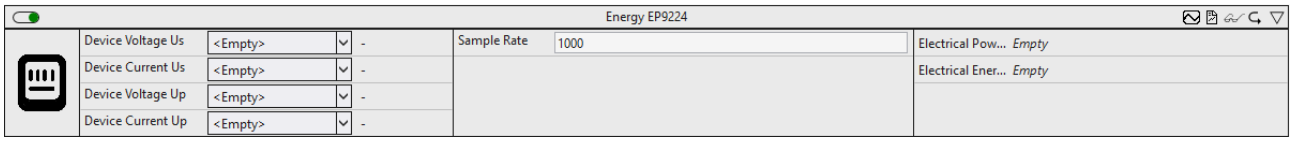
入力値

- **Supply Voltage:**電源電圧（0.01 V単位）
  プロセスデータ：PDO 入力 0x1A01 を有効化し、CoE 0x8002:11 で CoE 0xF900:05「Supply voltage」を選択します。
- **Supply Current:**入力電流（mA）
  プロセスデータ：PDO 入力 0x1A01 を有効化し、CoE 0x8002:19 で CoE 0xF900:06 「Supply current」を選択します。

出力値

- **Electrical Power [W]:**電力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.8 Energy EP9224



*Energy EP9224*モジュールはEP9224 EtherCATボックスモジュールの電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

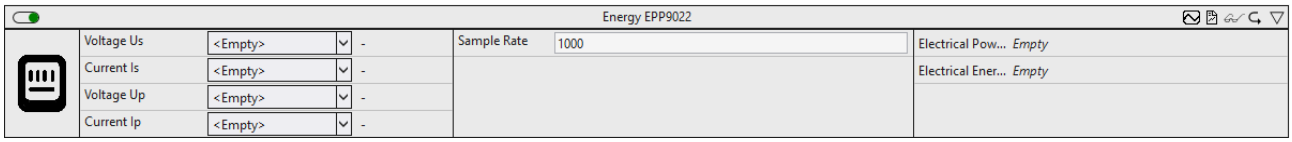
入力値

- **Device Voltage Us :** 制御電圧（V）
  プロセスデータ：PDO入力 「Device」
- **Device Current Us:**制御電流（A）
  プロセスデータ：PDO入力 「Device」
- **Device Voltage Up:**負荷電圧（V）
  プロセスデータ：PDO入力 「Device」
- **Device Current Up:**負荷電流（A）
  プロセスデータ：PDO入力 「Device」

出力値

- **Electrical Power [W]:**電力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.5.9 Energy EPP9022



*Energy EPP9022*モジュールは、EtherCATボックス EPP9022 の電力監視に適しています。入力データは設定した区間内で平均化され、計算されます。必要な入力データを確保するため、プロセスイメージに追加する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Interval（オプション）：**デフォルトでは、入力データは1秒間隔で平均化されます。

入力値

- **Device Voltage Us :** 制御電圧（V）
  プロセスデータ：PDO入力 「Device」
- **Device Current Is:**制御電流（A）
  プロセスデータ：PDO入力 「Device」

- **Device Voltage Up:**負荷電圧（V）
  プロセスデータ：PDO入力 「Device」
- **Device Current Ip:**負荷電流（A）
  プロセスデータ：PDO入力 「Device」

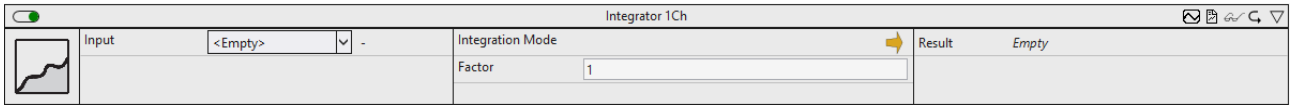
出力値

- **Electrical Power [W]:**電力（W）
- **Electrical Energy [Wh]:**ワット時（Wh）

6.5.6 Analytics - Math（解析 - 算術）

*Analytics-Math*カテゴリのアルゴリズムは、基本的な数学演算、積分、勾配解析などの数学的機能を提供します。

6.5.6.1 Integrator 1Ch



*Integrator 1Ch*は、入力値を基準時間 1 秒で時間積分し、その積分結果を出力します。この積分の近似には台形則が用いられます。後続の2つのタイムスタンプ t_n および t_{n+1} 、それぞれの時点での値 y_n および y_{n+1} に基づく台形近似 $T(t_n, t_{n+1})$ は、以下の式で計算されます。

$$T(t_n, t_{n+1}) = (t_{n+1}[s] - t_n[s]) \cdot \frac{y_n + y_{n+1}}{2}$$

.

設定で積分モード 「絶対値（|x|）」が選択されている場合、上の式の y_n と y_{n+1} は、絶対値で置き換えられます。

各サイクルにおいて、現在のタイムスタンプと最後のタイムスタンプの間の台形則が計算され、解析開始からの台形則の合計に加算されます。さらに、この合計値は、倍率係数を設定してスケーリングできます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Integration Mode:**積分モードを選択できます。→：入力値がそのまま積分されます。|x|:入力信号の絶対値が積分されます。
- **Factor:**この係数を使用して積分を乗算します。

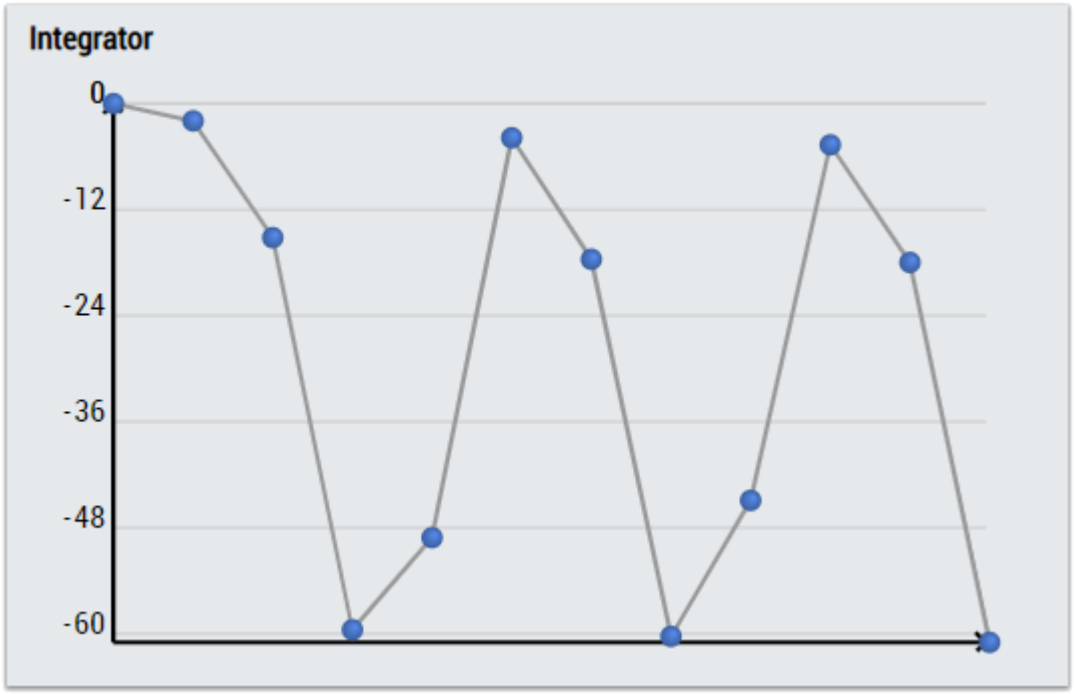
出力値

- **Result:**積分の結果を表示します。

HMIコントロール

Integrator 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.Integratorコントロールは、直近の x個の出力値 Resultを表示します。



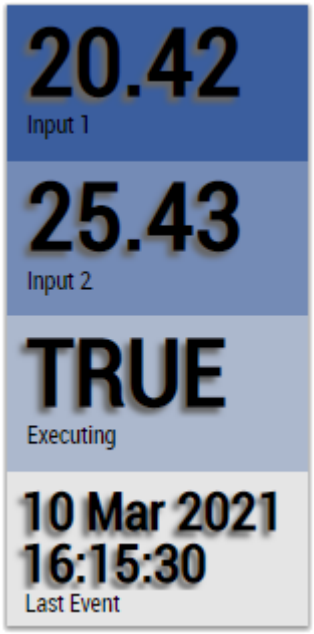
2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Result。

Data Table Vertical

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

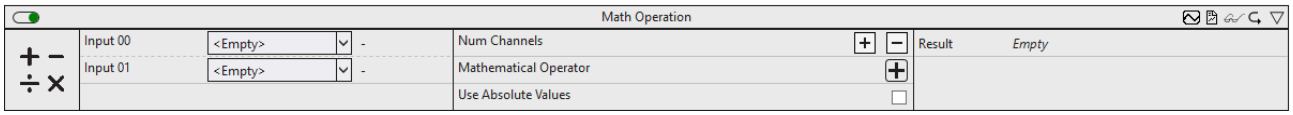
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをIntegrator 1Chアルゴリズムにマッピングすることもできます。

6.5.6.2 算術演算



*Math Operation*は、2つ以上の異なる入力チャンネルに対して数学演算を実行し、結果を提供します。演算子は全てのオペランド（演算対象）で共通です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels:**関数に入力されるオペランド（演算対象）の数。
- **Mathematical Operator:**演算に使用する演算子（+、-、x、/、x^y、%）。
- **Use Absolute Values:**チェックボックスをオンにすると、入力信号の絶対値が使用されます。

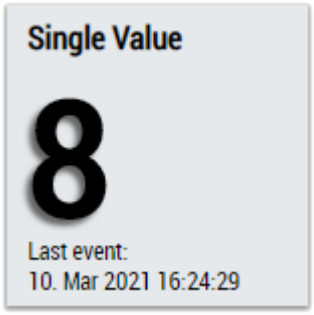
出力値

- **Result:**数学演算の結果。

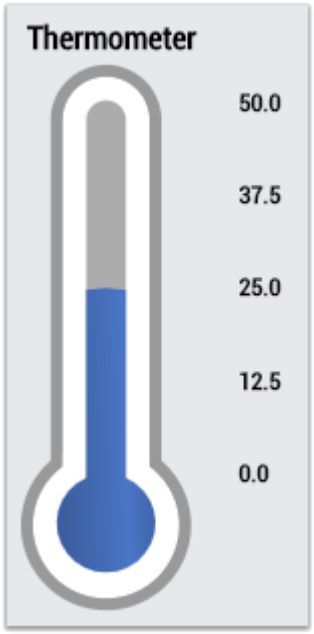
HMIコントロール

Math Operationアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

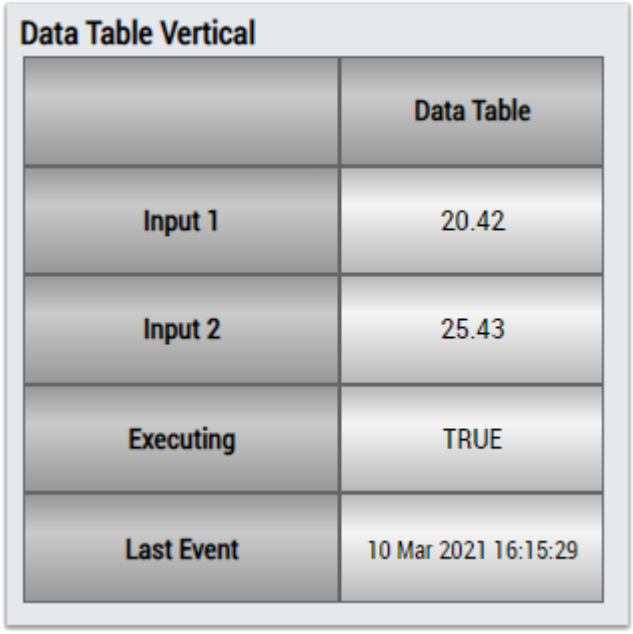
1.SingleValue コントロールは、出力値 Result を表示します。



2.Thermometerコントロールは、出力値 Resultを可視化し、温度表示に使用できます。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Result。

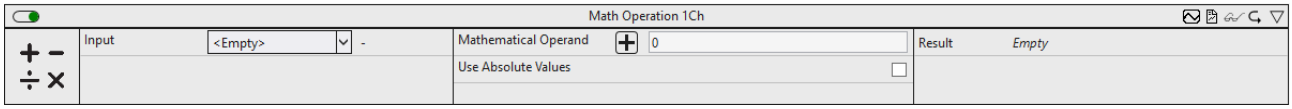


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールを数学演算アルゴリズムにマッピングすることもできます。

6.5.6.3 Math Operation 1Ch



Math Operation 1Ch は、入力チャンネルの信号と基準値に対して数学演算を実行します。アルゴリズムは数学演算の結果を提供し、演算子は個別に設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Mathematical Operator:**演算に使用する演算子（+、-、x、/、x^y、%）。
- **Mathematical Operand:**オペランド（演算対象）。
- **Use Absolute Values:**チェックボックスをオンにすると、入力信号の絶対値が使用されます。

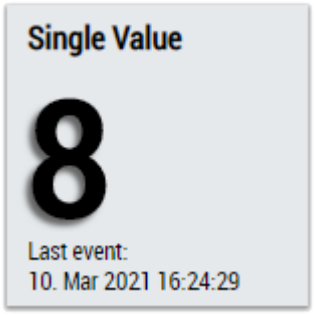
出力値

- **Result:**数学演算の結果。

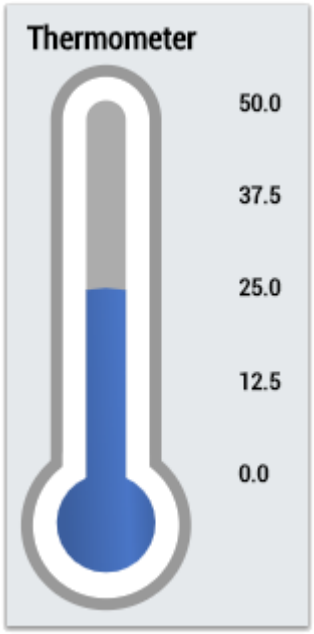
HMIコントロール

Math Operationアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

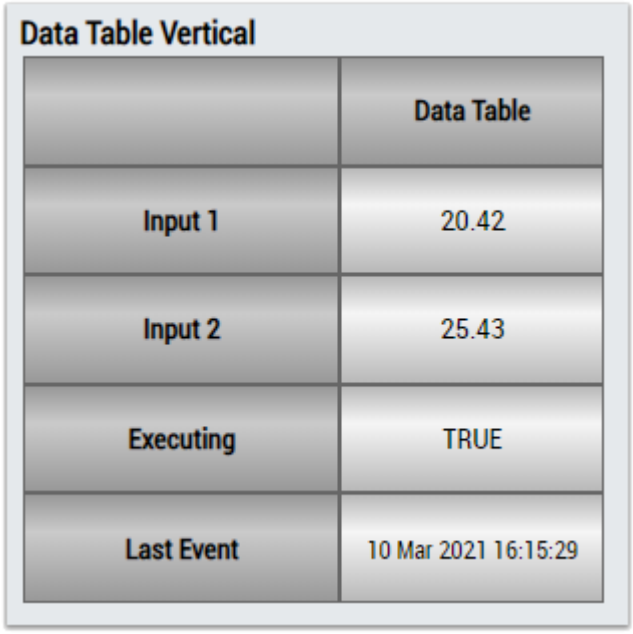
1.SingleValue コントロールは、出力値 Result を表示します。

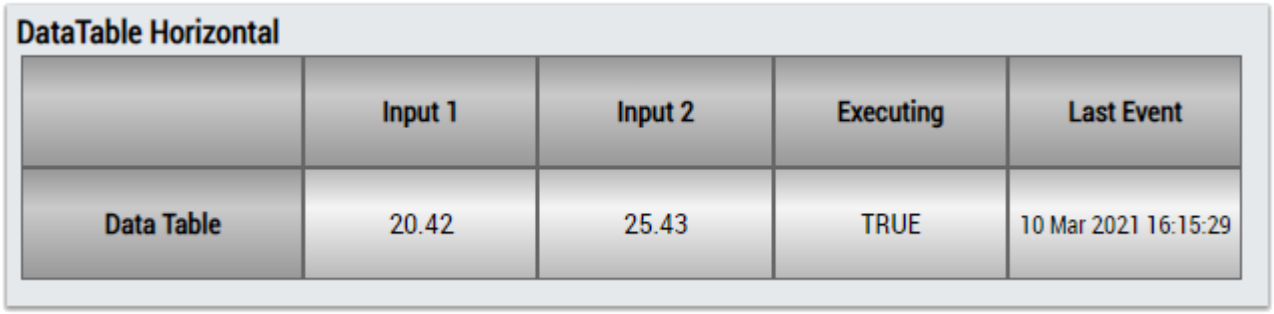


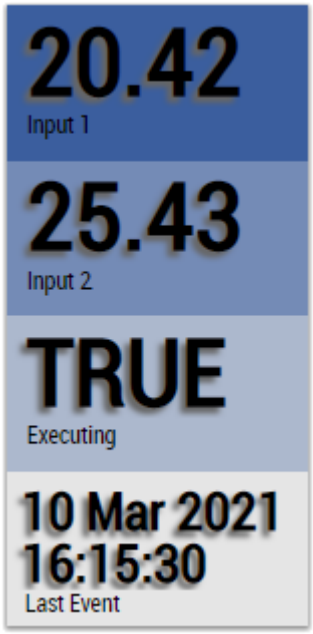
2.Thermometerコントロールは、出力値 Resultを可視化し、温度表示に使用できます。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Result。

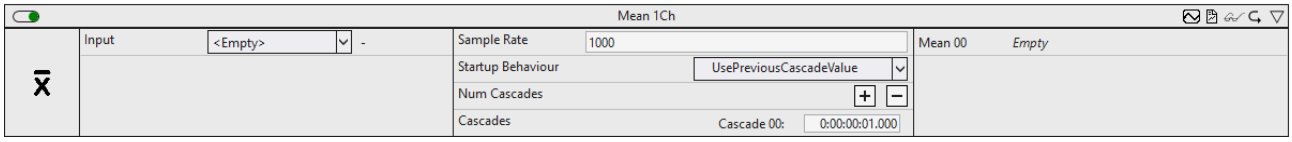






また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールを数学演算アルゴリズムにマッピングすることもできます。

6.5.6.4 Mean 1Ch



*Mean 1Ch*は、以下の式にしたがって入力値の平均値を計算します。

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x$$

計算に含まれるサンプル数Nは、時間区間を指定することで設定できます。カスケード出力は、省リソースで長期的な平均値を算出し、中間結果を取得できるように設定できます。カスケード区間は、前のカスケード区間の整数倍に設定する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Cascades:**カスケード出力の数
- **Cascades:**カスケード出力の設定。カスケード区間は、前のカスケード区間の整数倍に設定する必要があります。
- **Sample Rate:**解析するシステムのサンプルレート（Hz）

- **Startup Behavior:**
  *WaitUntilFilled*は、設定されたカスケード出力のタイムスパンが経過するまで待機します。結果および 「NewResult」ブールフラグ は、タイムスパンが終了した後に初めて設定されます
  UsePreviousCascadeValue：設定されたタイムスパンがまだ終了していないカスケードでは、すでに設定された中で次に短いタイムスパンの結果が使用されます。

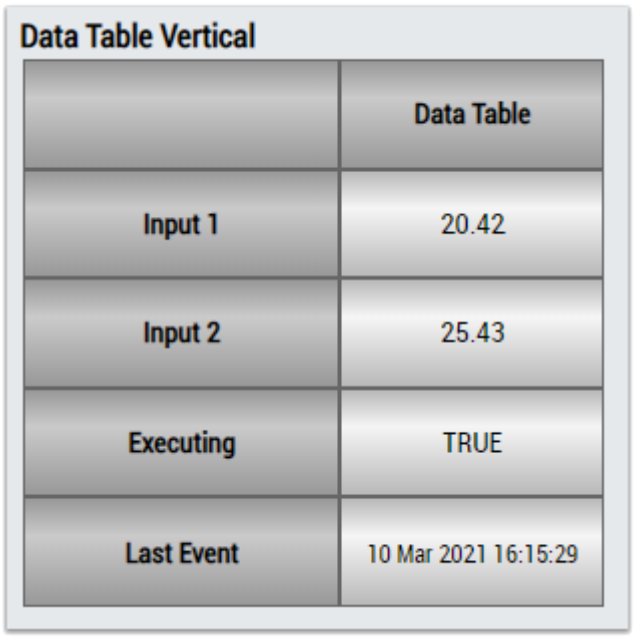
出力値

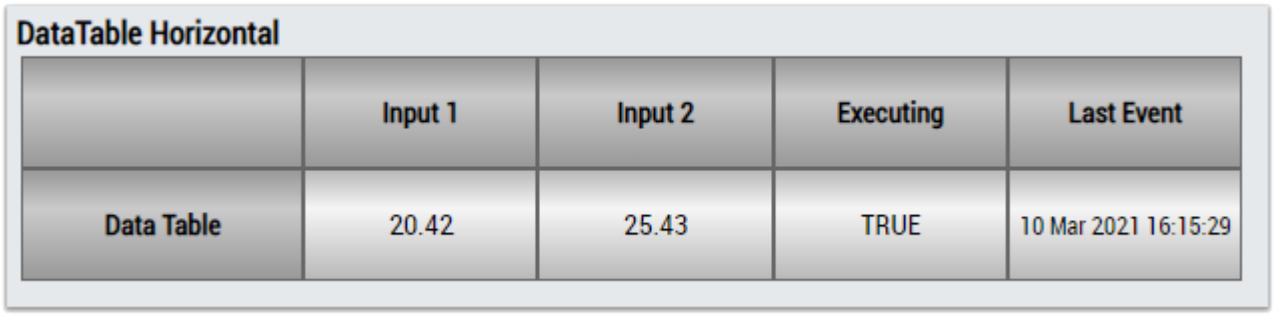
- **Mean:**平均値の計算結果が保存される出力配列。次元は設定したカスケード出力の数に相当します。起動時の動作は、*Startup Behavior*パラメータで設定できます。

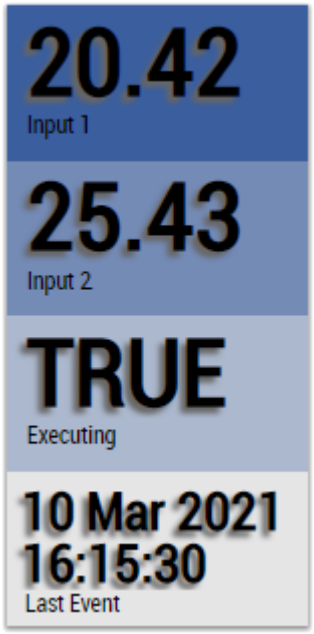
HMIコントロール

Meanアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Mean Results。

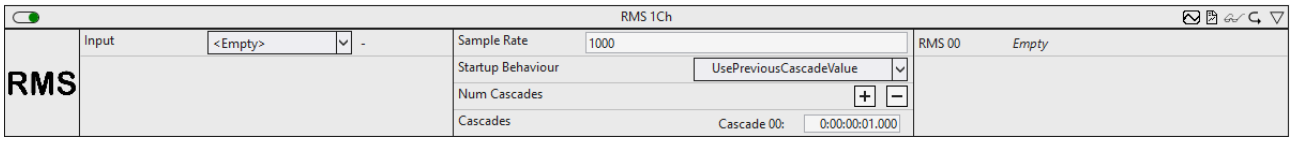






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをMeanアルゴリズムにマッピングすることもできます。

6.5.6.5 RMS 1Ch



*RMS 1Ch*は、次の式にしたがって入力値の二乗平均平方根（RMS値）を計算します。

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x[n]^2}$$

計算に含まれるサンプル数Nは、時間区間を指定することで設定できます。カスケード出力は、省リソースで長期的なRMS値を算出し、中間結果を取得できるように設定できます。カスケード区間は、前のカスケード区間の整数倍に設定する必要があります。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Cascades:**カスケード出力の数
- **Cascades:**カスケード出力の設定。カスケード区間は、前のカスケード区間の整数倍に設定する必要があります。
- **Sample Rate:**解析するシステムのサンプルレート（Hz）
- **Startup Behaviour:**
  WaitUntilFilled は、設定されたカスケード出力のタイムスパンが経過するまで待機します。RMS結果および「NewResult」ブールフラグは、タイムスパンが終了した後に初めて設定されます。
  UsePreviousCascadeValue 設定されたタイムスパンがまだ終了していないRMSカスケードでは、すでに設定された中で次に短いタイムスパンのRMS結果が使用されます。

出力値

- **RMS：**RMS計算結果が格納される出力配列。次元は設定したカスケード出力の数に相当します。起動時の動作は、パラメータ*Startup Behaviour*で設定できます。

HMIコントロール

RMSアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

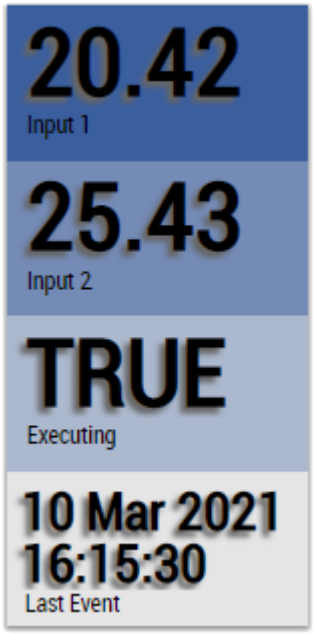
1.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：RMS Results。

Data Table Vertical

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

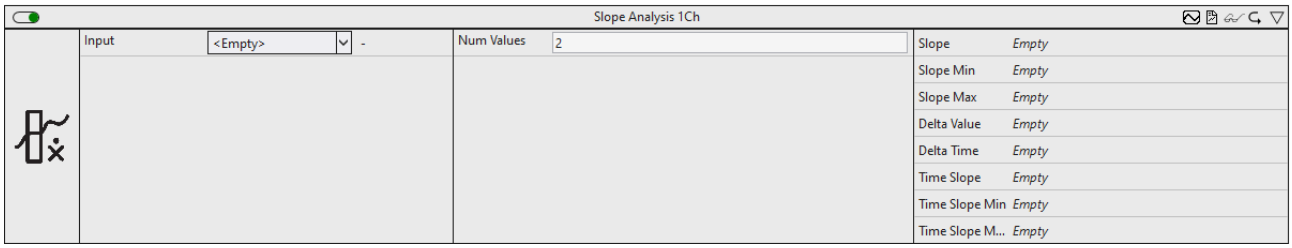
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをRMSアルゴリズムにマッピングすることもできます。

6.5.6.6 Slope Analysis 1Ch



*Slope Analysis 1Ch*は、入力ストリームのうち、2つの値の間の勾配を計算します。2つの値のうち1つは現在の入力値であり、もう1つは、入力ストリーム上で所定のサイクル数（*Num Values*パラメータで設定）前に記録された入力値です。この2つの値の差が*Delta Value*として返されます。

時間座標上における該当する2つの値の距離は、それぞれのタイムスタンプの差分として計算され、出力値*Delta Time*として提供されます。*Delta Time*はナノ秒単位で表示されますが、勾配の計算では秒を基本単位としてスケーリングされることに注意してください。

*Slope*は、*Delta Value*を*Delta Time*（秒単位にスケーリング）で割った値として計算され、*Delta Time*の算出に用いた2つのタイムスタンプの中間時点における勾配を推定します。これは、入力ストリームのタイムスタンプに対応している場合に、*Time Slope*として返される値です。設定によっては、*Num Values* が奇数の場合、中央のタイムスタンプに正確に一致する入力値は存在しません。この場合、算出された中央のタイムスタンプの直後の値のタイムスタンプが*Time Slope*として返されます。

さらに、このアルゴリズムは、最小勾配、最大勾配、最小時と最大時の時間値を提供します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

Num Values: 勾配の計算に使用される値の間のサイクル数。

出力値

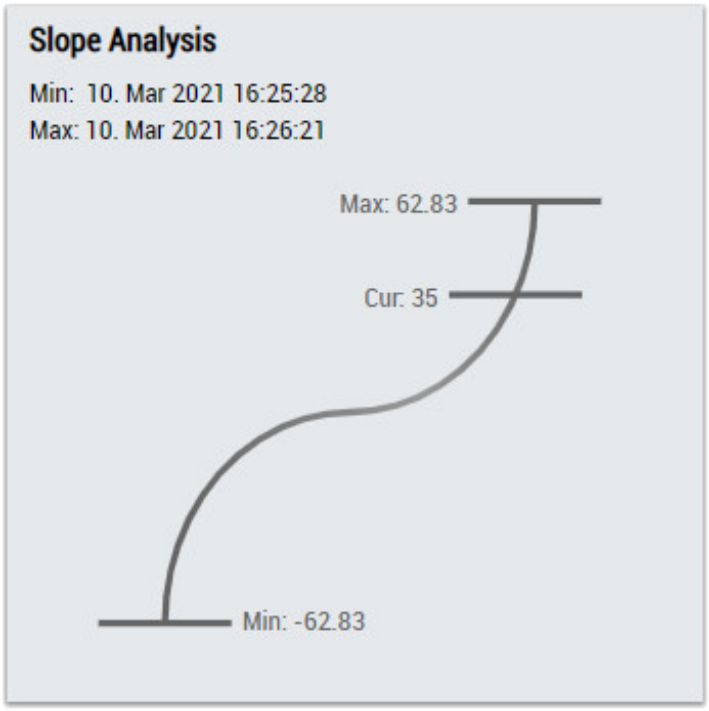
- **Slope:**現在の勾配値を示します。
- **Slope Min:**最小勾配値を示します。

- **Slope Max:**最大勾配値を示します。
- **Delta Value:**最新の勾配を計算するために使用される2つの値の差を示します。
- **Delta Time :** 最新の勾配を計算するために使用される期間を示します。
- **Time Slope:**最新の勾配の時間値を示します。
- **Time Slope Min:**最小勾配時の時間値を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。
- **Time Slope Max:**最大勾配時の時間値を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

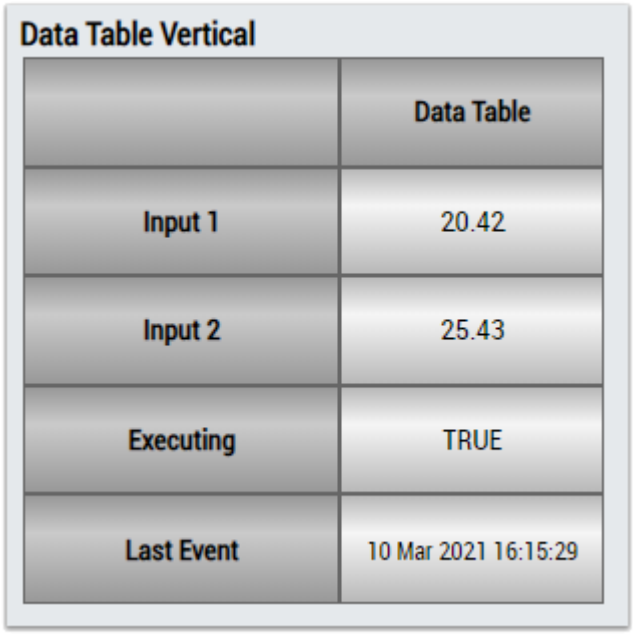
HMIコントロール

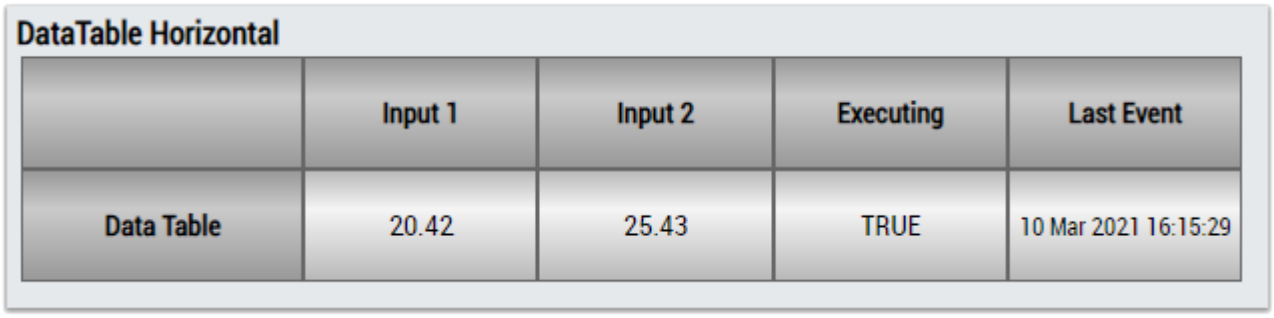
*Slope Analysis 1Ch*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

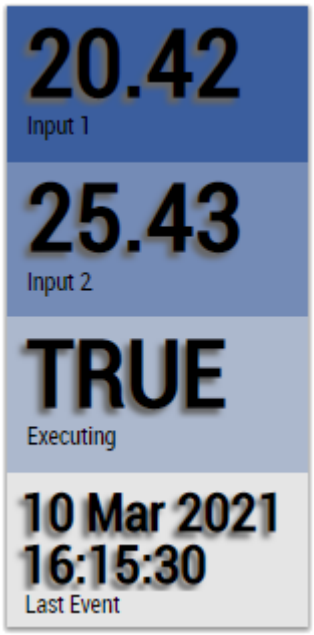
1.SlopeAnalysisコントロールは、出力値Slope、Slope Min、Slope Max、Time Slope Min、Time Slope Maxを表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Slope, Slope Min, Slope Max, Delta Value, Delta Time, Time Slope, Time Slope Min, Time Slope Max。

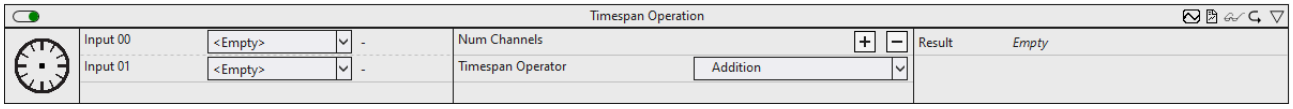






また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをアルゴリズムにマッピングすることもできます。

6.5.6.7 Timespan operation



*Timespan operation*は、2つ以上の異なる入力チャンネルに対して数学演算を実行し、結果を返します。入力として1nsの分解能を持つタイムスパンがサポートされています。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels:**関数に入力されるオペランド（演算対象）の数。
- **Timespan Operator:**演算に使用する演算子（*addition*/*subtraction*）。

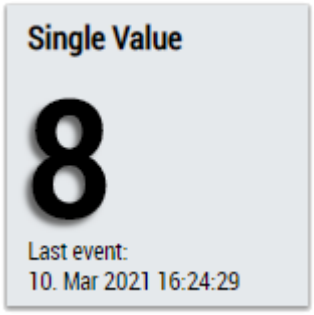
出力値

- **Result :** タイムスパンの数学演算の結果。

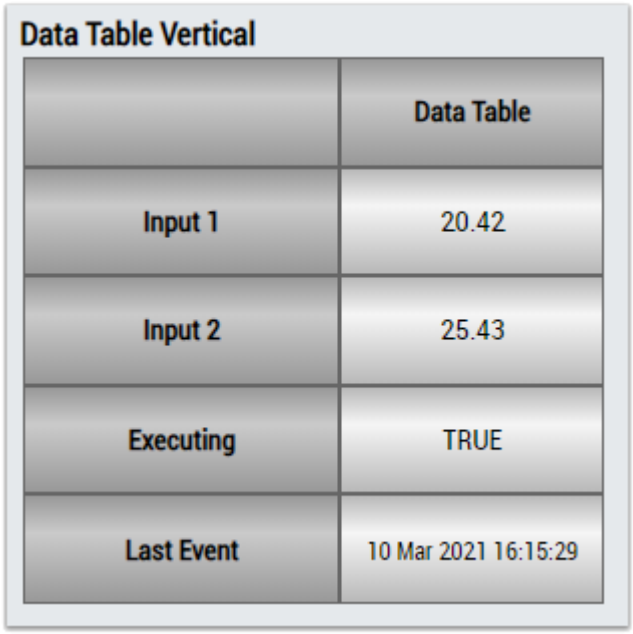
HMIコントロール

Timespan Operationアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.SingleValue コントロールは、出力値 Result を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Result。



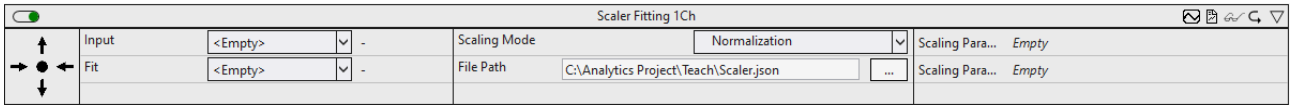
DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをTimespan Operationアルゴリズムにマッピングすることもできます。

6.5.7 Analytics - Preprocessing（解析 - 前処理）

6.5.7.1 Scaler Fitting



*Scaler Fitting*モジュールは、特徴量スケーリングのパラメータを算出し、それらをファイルに保存するために使用できます。これらのパラメータは、*Scaler*モジュールで使用できます。

スケーリングに必要なパラメータは、動作モードに応じて決定されます。

Normalization：

$$x' = \frac{x - low(x)}{high(x) - low(x)}$$

Standardization：

$$x' = \frac{x - \bar{x}}{\sigma}$$

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

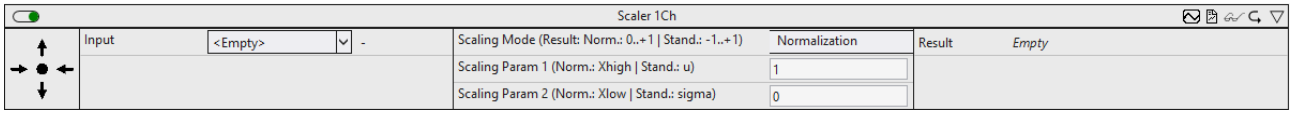
設定オプション

- **Scaling Mode:** スケーリングモード。
  Normalization（正規化）
  Standardization（標準化）
- **File Path:** スケーリングパラメータを保存するファイルパスを指定します。ファイル形式：*json*

出力値

- **Scaling Param 1:** 最初のスケーリングパラメータの値は、設定したスケーリングモードに依存します。
  Normalization：入力信号の最大値
  Standardization：入力信号の平均値
- **Scaling Param 2：** 2番目のスケーリング・パラメータの値は、設定したスケーリングモードに依存します。
  Normalization：入力信号の最大値
  Standardization：入力信号の平均値

6.5.7.2 Scaler（スケーラー）



*Scaler*モジュールは、特徴量のスケーリングを実行するために使用できます。

動作モードに応じて、スケーリングパラメータは異なる方法で使用されます。

Normalization

$$x' = \frac{x - low(x)}{high(x) - low(x)}$$

Standardization：

$$x' = \frac{x - \bar{x}}{\sigma}$$

スケーリングパラメータは直接設定できます。また、ファイルからインポートすることもできます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Scaling Mode:**スケーリングモード。
  Normalization（正規化）*:*結果の値の範囲は0～+1。
  Standardization:（標準化）結果の値の範囲は-1～+1。
- **Scaling Param 1:** 最初のスケーリングパラメータの値は、設定したスケーリングモードに依存します。
  Normalization：入力信号の最大値
  Standardization：入力信号の平均値
- **Scaling Param 2：** 2番目のスケーリング・パラメータの値は、設定したスケーリングモードに依存します。
  Normalization：入力信号の最大値
  Standardization：入力信号の平均値

- **File Path（オプション）：** 事前にパラメータ設定したファイルのファイルパス。このファイルは *Scaler Fitting*モジュールにより作成されます。ファイル形式：*json*

出力値

- **Result:** 計算された値を出力します。これは、スケーリングモードに応じて、2つのスケーリングパラメータから計算されます。
  Normalization（正規化）*:*結果の値の範囲は0～+1。
  Standardization:（標準化）結果の値の範囲は-1～+1。

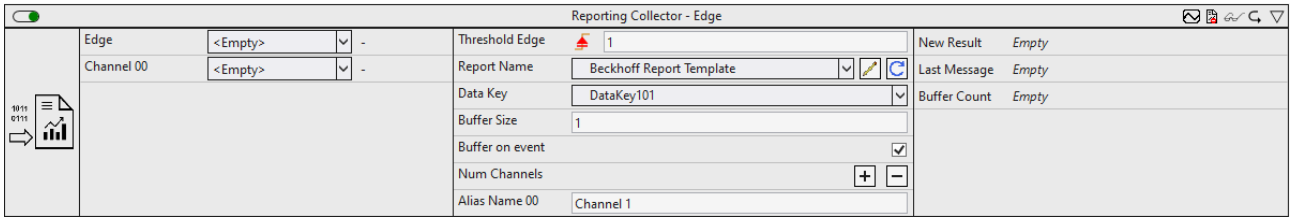
6.5.8 Analytics - Reporting（解析 - レポート）

Reporting カテゴリのアルゴリズムを使用して、TwinCAT Analyticsに24時間365日のレポート機能を実装できます。Reporting Collector（レポートコレクタ）はデータを収集し、Reporting Server（レポートサーバ）に送信します。レポートトリガーはレポートの作成をトリガーします。

6.5.8.1 Reporting Collector（レポートコレクタ）

Reporting Collector（レポートコレクタ）はデータを収集し、イベント後にデータメッセージとして Reporting Server（レポートサーバ）に送信します。

6.5.8.1.1 Reporting Collector Edge（レポートコレクタエッジ）



Reporting Collector Edge（レポートコレクタエッジ）は入力チャンネルからデータを収集し、設定に応じて、イベント後またはバッファが満たされた後にReporting Server（レポートサーバ）にデータ送信します。入力チャンネルの信号が、設定したエッジで閾値を超過すると、イベントがトリガーされます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。
- **Data Key:**データキーはレポート内で一意である必要があります。データキーにより、データオブジェクトをレポートに対して一意に割り当てることが可能です。
- **Buffer Size :** バッファリングされるデータのサイズを指定します。バッファサイズが1の場合、キーと値の構造体が構築されます。バッファサイズが1より大きい場合、データは表で示されます。
- **Buffer on event :** データをどのように収集し、バッファリングするかを指定します。
  パラメータがTrueの場合、入力データは入力のエッジ信号ごとにバッファリングされます。Buffer Count 出力が Buffer Size パラメータと同じ値になるとすぐに、データは TcReportingServer に送信されます。

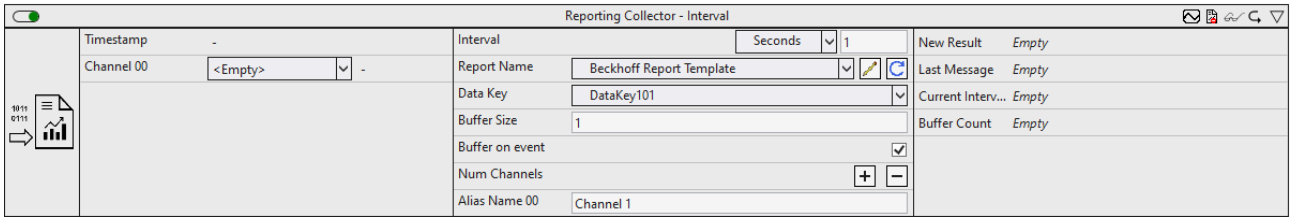
パラメータが False の場合、入力のデータはサイクルごとにバッファリングされます。バッファサイズに達すると、新しいデータが前のデータを置き換えます。入力にエッジ信号がある場合に、データはTcReportingServerに送信されます。

- **Num Channels :** データを収集する入力チャンネルの数。また、入力チャンネルに説明（ラベル付け）を可能にするパラメータ数を示します。
- **Alias Name:**入力チャンネルの識別のための説明（ラベル名）です。BufferSize = 1 : エイリアス名は、キーと値の構造体でキーとして機能します。BufferSize > 1: エイリアス名は、テーブル列の見出しとなります。
- **Include Timestamps (オプション):** タイムスタンプを含む列を挿入します。

出力値

- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Buffer Count :** バッファ内の要素数を指定します。

6.5.8.1.2 Reporting Collector Interval



Reporting Collector Intervalは入力チャンネルからデータを収集し、設定に応じて、イベント後またはバッファが満たされた後にReporting Server（レポートサーバ）にデータ送信します。設定した時間区間が経過すると、イベントがトリガーされます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

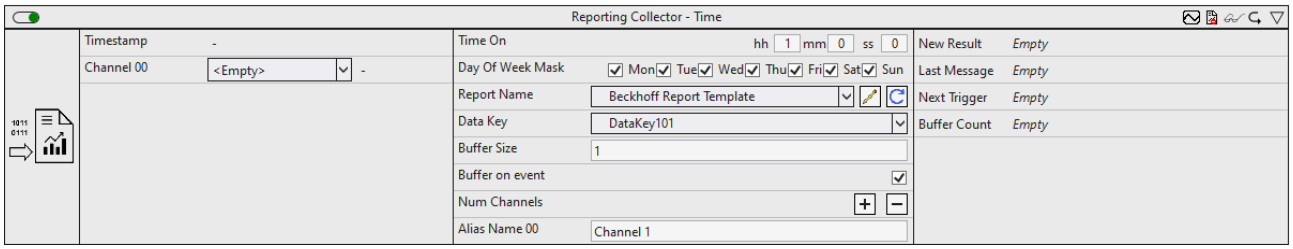
設定オプション

- **Interval :** データをReporting Serverに送信する時間区間。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。
- **Data Key:**データキーはレポート内で一意である必要があります。データキーにより、データオブジェクトをレポートに対して一意に割り当てることが可能です。
- **Buffer Size :** バッファリングされるデータのサイズを指定します。バッファサイズが1の場合、キーと値の構造体が構築されます。バッファサイズが1より大きい場合、データは表で示されます。
- **Buffer on event :** データをどのように収集し、バッファリングするかを指定します。
  パラメータがTrueの場合、入力データは入力のエッジ信号ごとにバッファリングされます。Buffer Count 出力が Buffer Size パラメータと同じ値になるとすぐに、データは TcReportingServer に送信されます。
  パラメータが False の場合、入力のデータはサイクルごとにバッファリングされます。バッファサイズに達すると、新しいデータが前のデータを置き換えます。入力にエッジ信号がある場合に、データはTcReportingServerに送信されます。
- **Num Channels :** データを収集する入力チャンネルの数。また、入力チャンネルに説明（ラベル付け）を可能にするパラメータ数を示します。
- **Alias Name:**入力チャンネルの識別のための説明（ラベル名）です。BufferSize = 1 : エイリアス名は、キーと値の構造体でキーとして機能します。BufferSize > 1: エイリアス名は、テーブル列の見出しとなります。
- **Include Timestamps (オプション):** タイムスタンプを含む列を挿入します。

出力値

- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Current Interval Time:**現在の区間の経過時間を示します。
- **Buffer Count :** バッファ内の要素数を指定します。

6.5.8.1.3 Reporting Collector Time



Reporting Collector Timeは入力チャンネルからデータを収集し、設定に応じて、イベント後またはバッファが満たされた後にデータをReporting Server（レポートサーバ）に送信します。設定されたスイッチオン時間に達するとイベントがトリガーされます。スイッチオンの時間と曜日を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Time On :** スイッチオン時間。
- **Day of Week Mask:**タイマーを有効にする曜日。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。
- **Data Key:**データキーはレポート内で一意である必要があります。データキーにより、データオブジェクトをレポートに対して一意に割り当てることが可能です。
- **Buffer Size :** バッファリングされるデータのサイズを指定します。バッファサイズが1の場合、キーと値の構造体が構築されます。バッファサイズが1より大きい場合、データは表で示されます。
- **Buffer on event :** データをどのように収集し、バッファリングするかを指定します。
  パラメータがTrueの場合、入力データは入力のエッジ信号ごとにバッファリングされます。Buffer Count 出力が Buffer Size パラメータと同じ値になるとすぐに、データは TcReportingServer に送信されます。
  パラメータが False の場合、入力のデータはサイクルごとにバッファリングされます。バッファサイズに達すると、新しいデータが前のデータを置き換えます。入力にエッジ信号がある場合に、データはTcReportingServerに送信されます。
- **Num Channels :** データを収集する入力チャンネルの数。また、入力チャンネルに説明（ラベル付け）を可能にするパラメータ数を示します。
- **Alias Name:**入力チャンネルの識別のための説明（ラベル名）です。BufferSize = 1 : エイリアス名は、キーと値の構造体でキーとして機能します。BufferSize > 1: エイリアス名は、テーブル列の見出しとなります。
- **Include Timestamps (オプション):** タイムスタンプを含む列を挿入します。

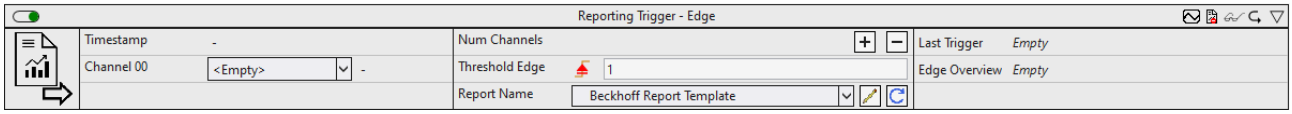
出力値

- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Next Message:**次のメッセージまでの残り時間を示します。
- **Buffer Count :** バッファ内の要素数を指定します。

6.5.8.2 Reporting Trigger

レポーティング・トリガーはイベント発生後にReporting Serverにトリガー・メッセージを送信し、レポートの作成をトリガーします。

6.5.8.2.1 Reporting Trigger Edge



Reporting Trigger Edgeは、イベントのトリガー後に、レポートの作成をトリガーします。入力チャンネル信号が指定された閾値で設定されたエッジを超えるとイベントがトリガーされます。内部的には、一度Trueになった入力は、Trueのままになります。入力は、全ての入力が少なくとも一度はTrueになった場合のみ、入力はFalseにリセットされます。これにより、出力 bNewResult を複数の Reporting Collector が共通の入力として使用できるようになり、すべての Reporting Collector がデータメッセージを送信した時点で、トリガーメッセージが送信されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

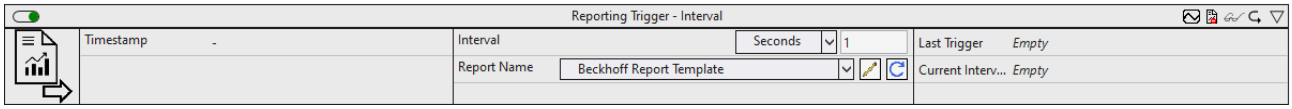
設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。

出力値

- **Last Trigger:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Edge Overview:**どの入力チャンネルが少なくとも一度はTrueであったかを示します。

6.5.8.2.2 Reporting Trigger Interval



Reporting Trigger Intervalは、イベントのトリガー後に、レポートの作成をトリガーします。設定した時間区間が経過すると、イベントがトリガーされます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

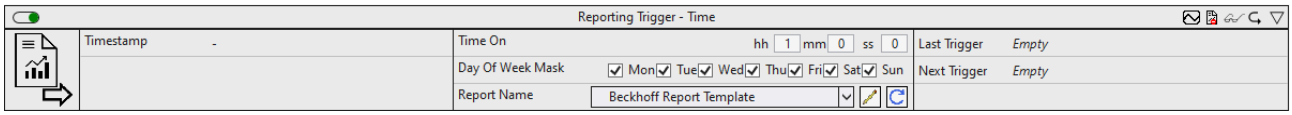
- **Interval :** データをReporting Serverに送信する時間区間。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。

出力値

- **Last Trigger:**最後のメッセージがいつReporting Serverに送信されたかを示します。

- **Current Interval Time:**現在の区間の経過時間を示します。

6.5.8.2.3 Reporting Trigger Time



Reporting Trigger Timeは、イベントのトリガー後に、レポートの作成をトリガーします。設定されたスイッチオン時間に達するとイベントがトリガーされます。スイッチオンの時間と曜日を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Time On :** スイッチオン時間。
- **Day of Week Mask:**タイマーを有効にする曜日。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。

出力値

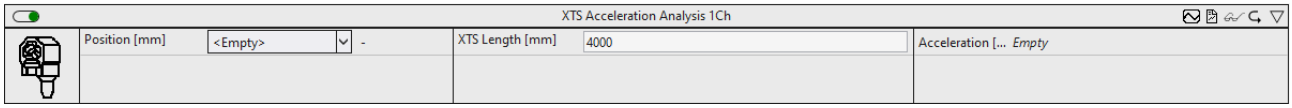
- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Next Trigger:**次のメッセージまでの残り時間を示します。

6.5.9 Analytics - Specific（解析 - 特殊）

6.5.9.1 XTS

*Analytics-XTS*カテゴリのアルゴリズムは、ベッコフのXTSシステムに特別な機能を提供します。例えば、距離、速度、加速度の解析などです。

6.5.9.1.1 XTS Acceleration Analysis 1Ch



*XTS Acceleration Analysis 1Ch*は、XTS可動子の現在の加速度を計算します。このためは、XTSの長さをミリメートル単位で宣言し、入力信号として可動子の位置が必要です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **XTS Length [mm]:** XTSシステムの長さ（ミリメートル単位）。

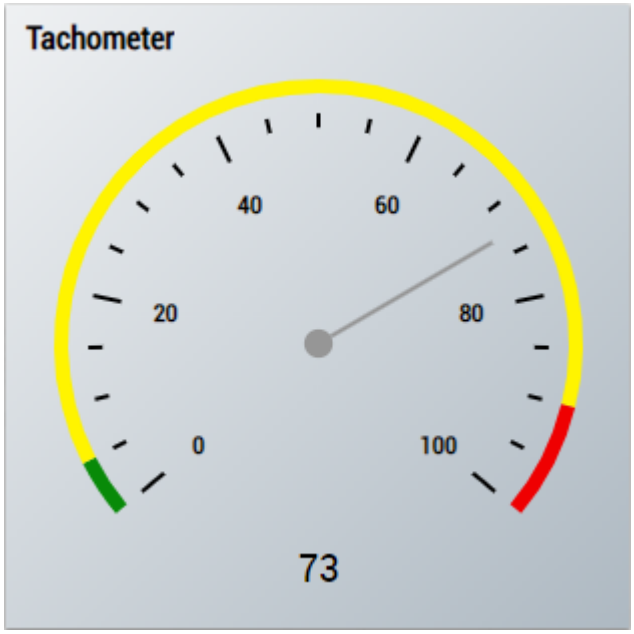
出力値

- **Acceleration:** XTS可動子の現在の加速度。

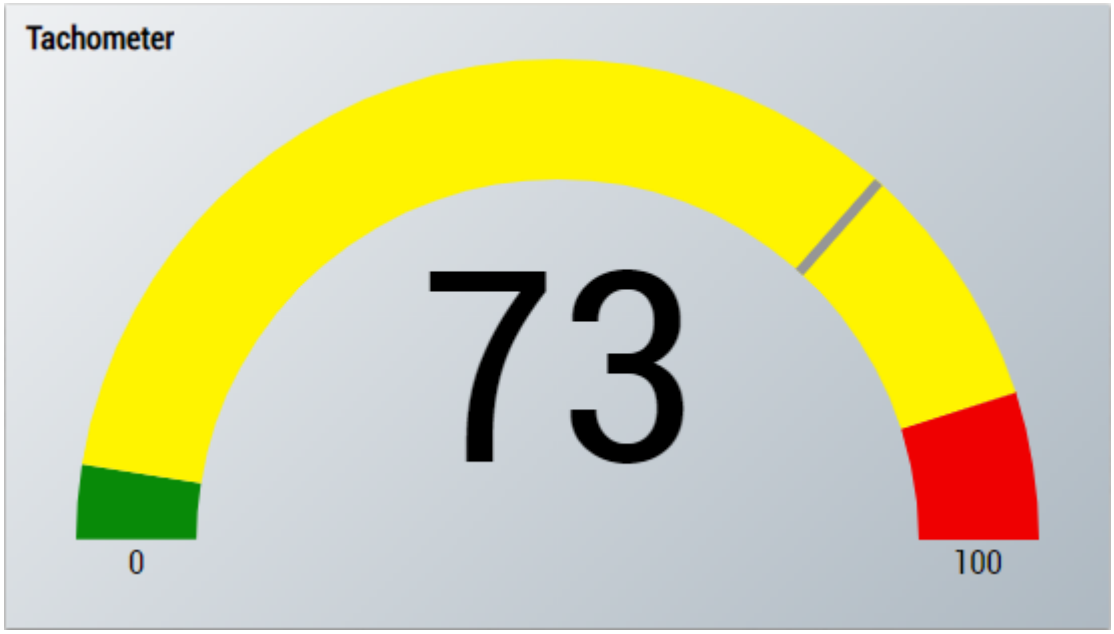
HMIコントロール

XTS Acceleration Analysis 1Chでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

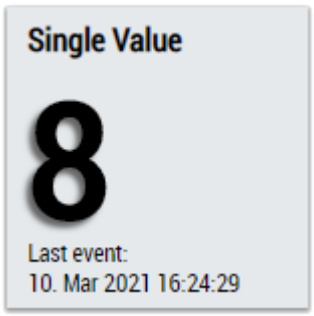
1.Tachometerコントロールは、出力値Accelerationを表示します。



2.Radial Gaugeコントロールは、出力値Accelerationを表示します。



3.SingleValue コントロールは、出力値 Acceleration を表示します。



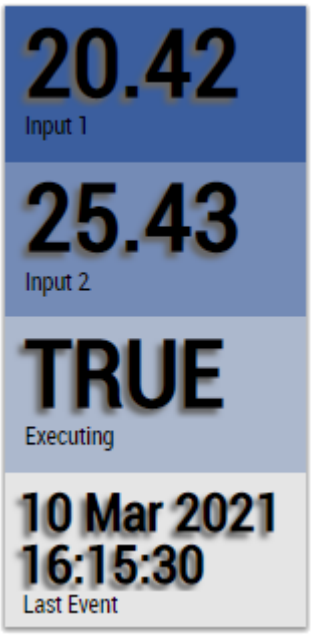
4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：
Acceleration。

Data Table Vertical

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

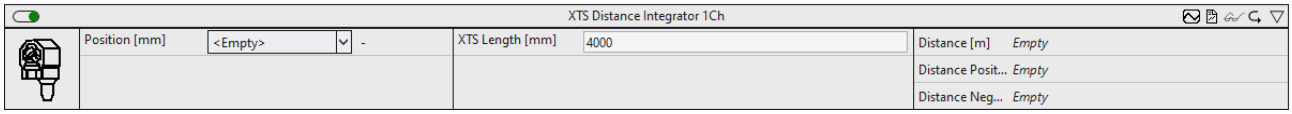
DataTable Horizontal

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



また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールを XTS Acceleration Analysis 1Ch アルゴリズムにマッピングすることもできます。

6.5.9.1.2 XTS Distance Integrator 1Ch



*XTS Distance Integrator 1Ch*は、XTS可動子の移動距離を計算します。このアルゴリズムは、総距離、正方向の距離、負方向の距離を提供します。このためは、XTSの長さをミリメートル単位で宣言し、入力信号として可動子の位置が必要です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **XTS Length [mm]:** XTSシステムの長さ（ミリメートル単位）。

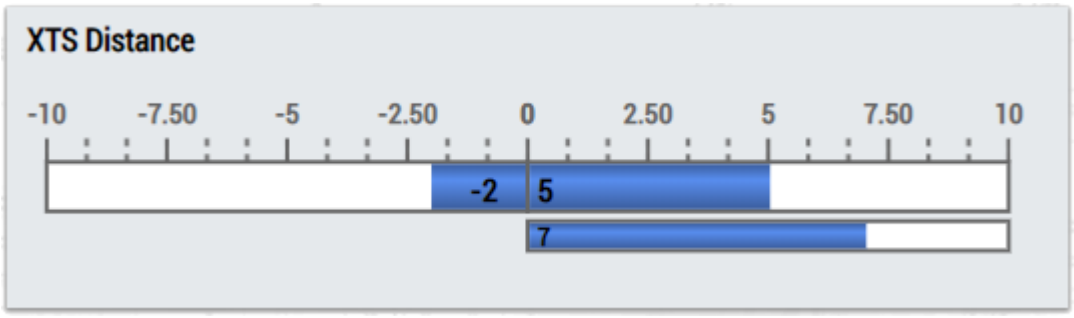
出力値

- **Distance:** XTS可動子の総移動距離。
- **Distance Positive:** XTS可動子が正方向に移動した距離（前進）。
- **Distance Negative:** XTS可動子が負方向に移動した距離（後進）。

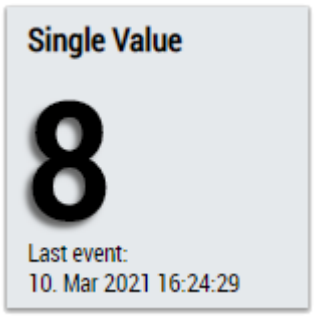
HMIコントロール

XTS Distance Integrator 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.XTSDistanceコントロールは、出力値Distance、Distance Positive、Distance Negativeを表示します。



2.SingleValueコントロールは、出力値Distanceを表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：出力値Distance、Distance Positive、Distance Negative。

Data Table Vertical

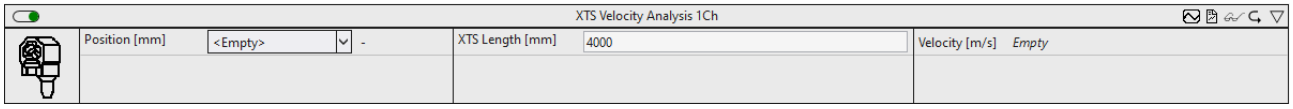
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、XTS Distance Integrator 1Ch アルゴリズムに顧客固有の HMIコントロールをマッピングすることもできます。

6.5.9.1.3 XTS Velocity Analysis 1Ch



*XTS Velocity Analysis 1Ch*は、XTS可動子の現在の速度を計算します。このためは、XTSの長さをミリメートル単位で宣言し、入力信号として可動子の位置が必要です。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **XTS Length [mm]:** XTSシステムの長さ（ミリメートル単位）。

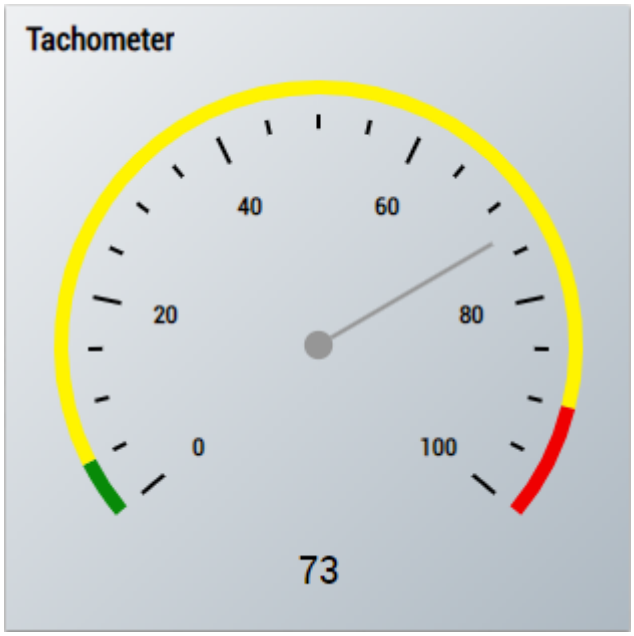
出力値

- **Velocity:** XTS可動子の現在の速度。

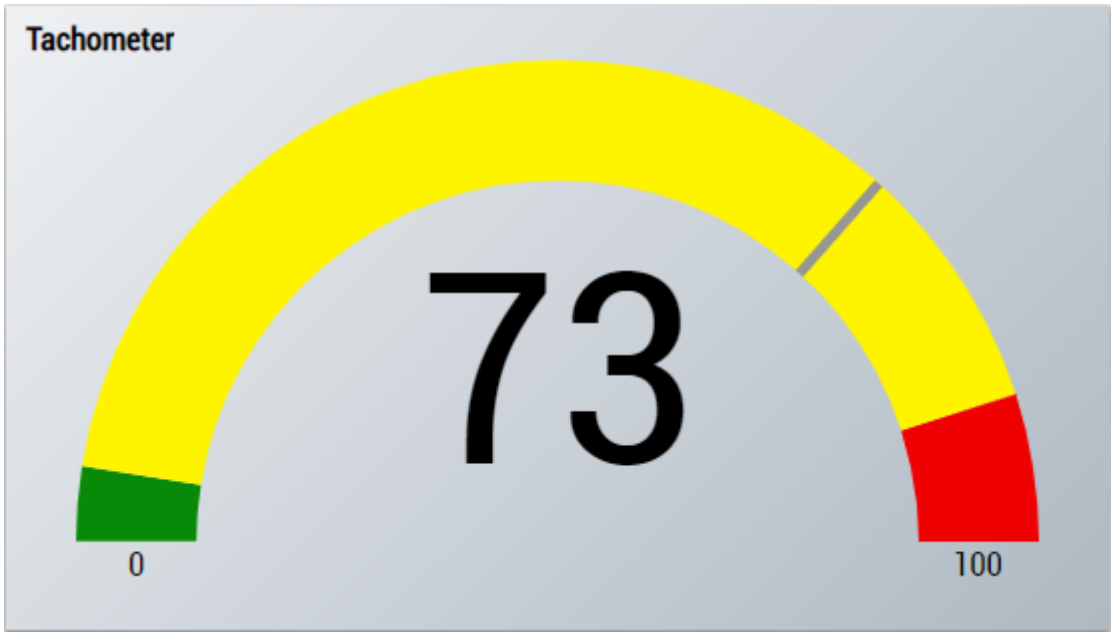
HMIコントロール

XTS Velocity Analysis 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

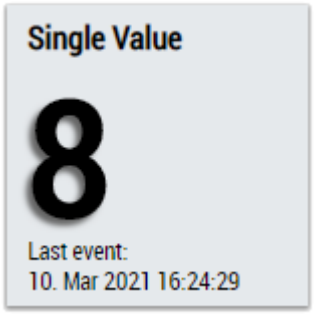
1.Tachometerコントロールは、出力値Velocityを表示します。



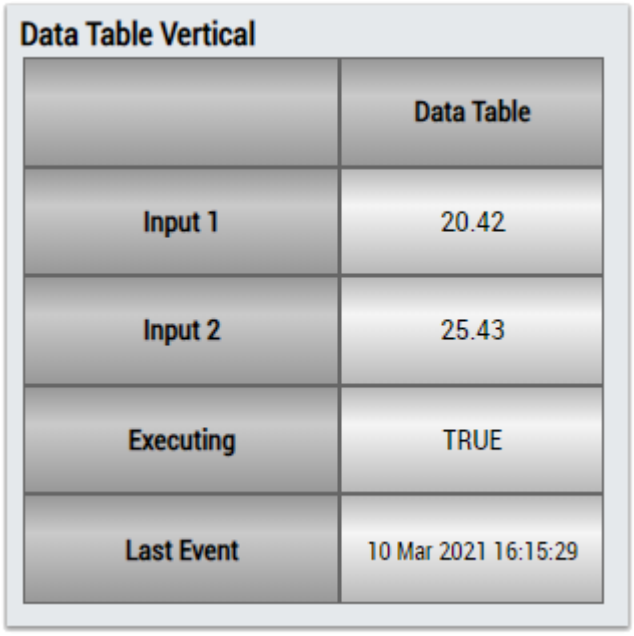
2.Radial Gaugeコントロールは、出力値Velocityを表示します。

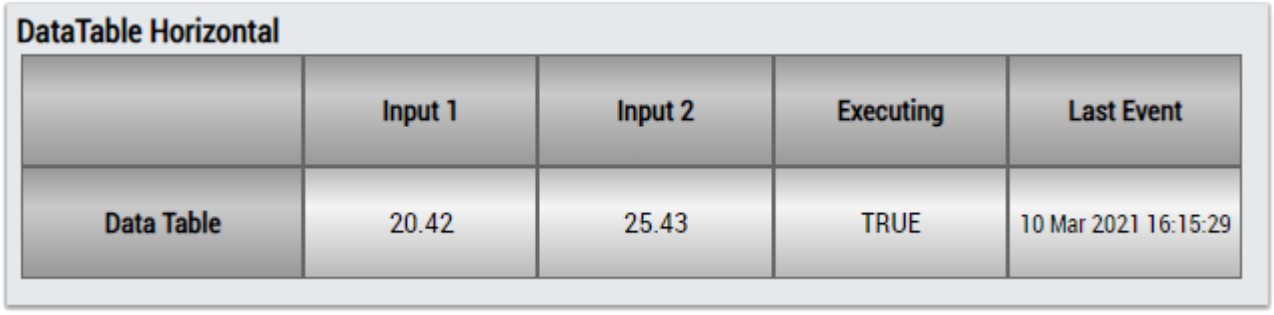


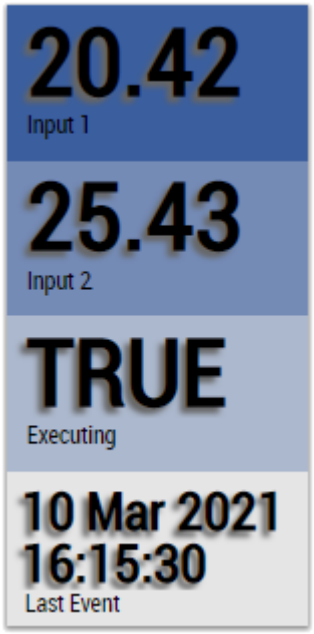
3.SingleValueコントロールは、出力値Velocityを表示します。



4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Velocity。





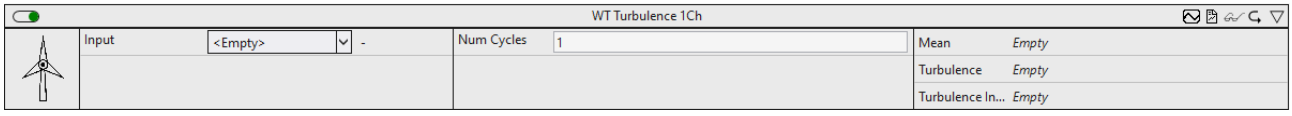


また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールを XTS Velocity Analysis 1Ch アルゴリズムにマッピングすることもできます。

6.5.9.2 Wind Turbine

*Analytics - WT*カテゴリのアルゴリズムは、風力発電業界向けに特別な機能を提供します。例えば、平均風速、乱気流、乱気流強度の解析などです。

6.5.9.2.1 WT Turbulence 1Ch



*WT Turbulence 1Ch*は、*EN 61400-1*規格に基づいた、風速、乱気流、乱気流強度の平均値を計算します。入力信号としては、風速が必要です。出力値は10分周期で更新されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Cycles 10Min :** EN 61400-1に基づき、10分間の時間間隔に収まるサイクル数を示します。

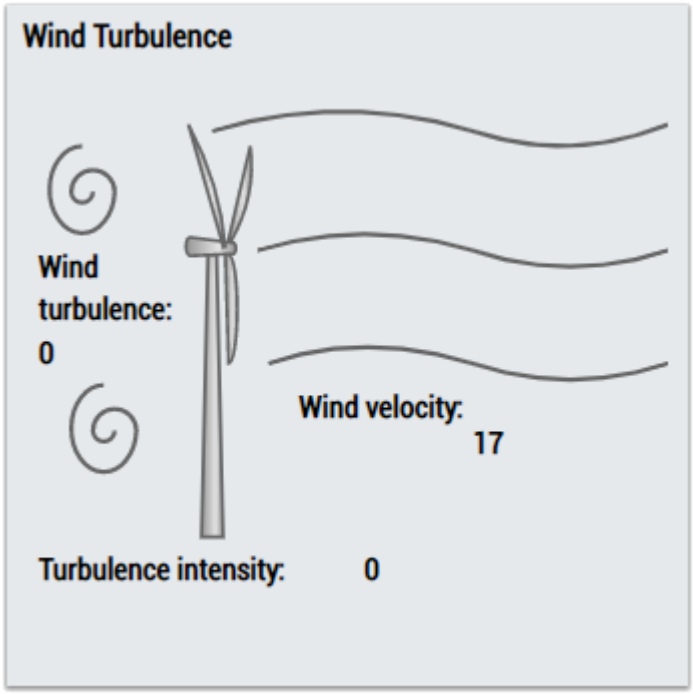
出力値

- **Mean:**風速の平均値。
- **Turbulence:** 風の乱れ（乱気流）。EN規格によれば、これは10分間の時間間隔における風速の標準偏差を指します。
- **Turbulence Intensity:** 乱気流強度。

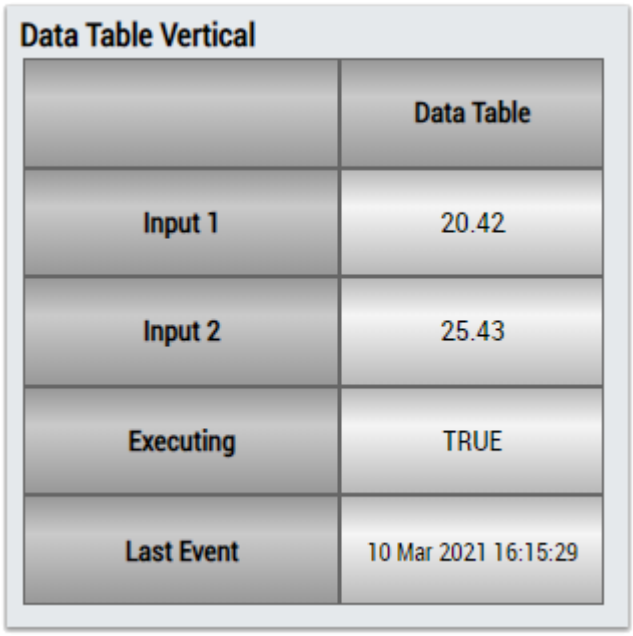
HMIコントロール

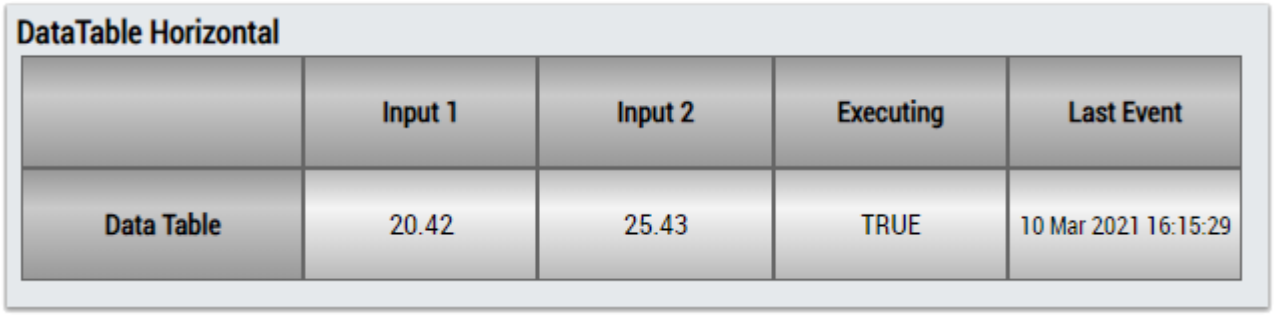
WT Turbulence 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

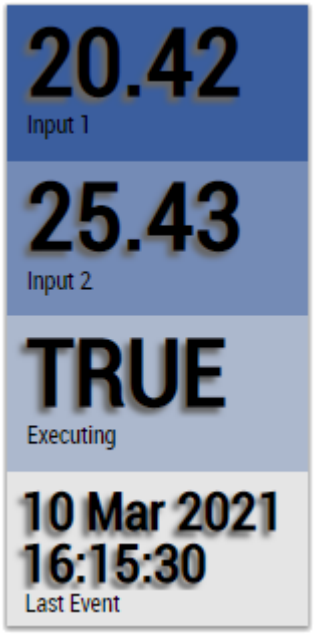
1.WindTurbulence コントロールは、出力値 Mean、Turbulence、および Turbulence Intensity を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Mean, Turbulence, Turbulence Intensity。





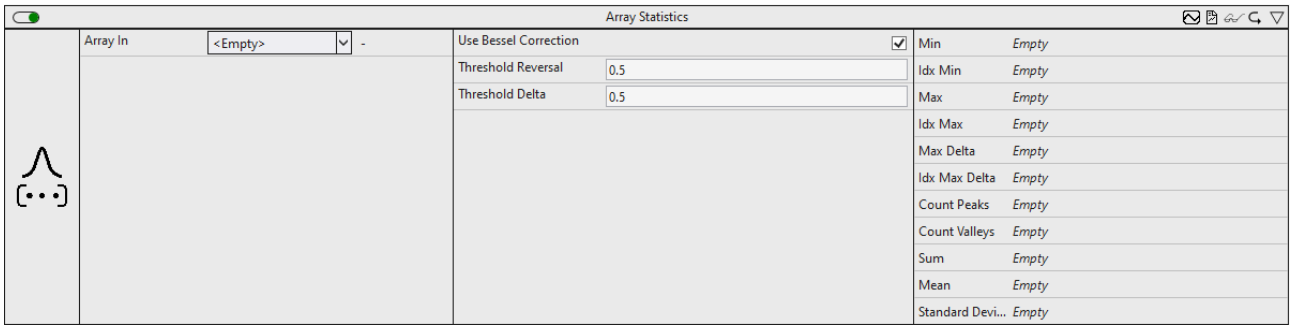


また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールを WT Turbulence 1Ch アルゴリズムにマッピングすることもできます。

6.5.10 Analytics - Statistics（解析 - 統計）

*Statistics*カテゴリのアルゴリズムは、統計的なデータ解析手法の機能を提供します。これには例えば、信号相関の計算や回帰解析が含まれます。

6.5.10.1 Array Statistics（配列統計）



*Array Statistics*アルゴリズムは、入力配列に基づいて各種の統計量を計算します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Use Bessel Correction:** チェックボックスをオンにすると、ベッセル補正が適用されます。ランダムなサンプルに対して期待値通りの結果を得るためには、このパラメータを有効にする必要があります。このパラメータは標準偏差の計算にのみ関係します。

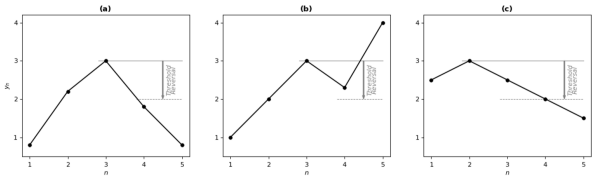
経験的標準偏差（ベッセル補正なし）

$$s' = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x[n] - \overline{x})^2}$$

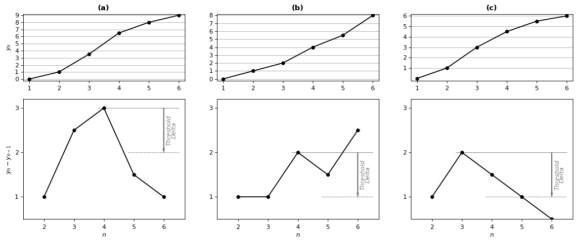
経験的標準偏差（ベッセル補正あり）

$$s = \sqrt{\frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$$

- **Threshold Reversal :** 反転を識別するための閾値 。Threshold Reversal の値を超える差がなければ反転は検出されません。
  以下に、パラメータ *Threshold Reversal* を用いたピーク識別の3つの例を示します。
  (a) 値 y_3 と y_4 の差が *Threshold Reversal* より大きいため、y_4 の処理直後に y_3 はピークとして識別されます。
  (b) y_3 と y_4 の差が *Threshold Reversal* 未満であり、y_4 の後に曲線が再び上昇し始めるため、y_3 はピークとして識別されません。
  (c) y_2 と y_5 の差が *Threshold Reversal* を超過するため、y_5 を処理した後に y_2 はピークとして識別されます。y_2とy_3（またはy_4）との差が *Threshold Reversal* 以下であり、その後も値が減少し続けるかどうかが不明なため、y_2 を事前にピークとして識別することはできません。



- **Threshold Delta:** Deltaの最大値を識別するための閾値。連続する2つの値の差の絶対値（Delta）の最大値は、連続するDelta同士の差が *Threshold Delta* を超えた場合にのみ検出されます。
  以下に、パラメータ *Threshold Delta* を用いたDelta最大値の識別例を3つ示します。上段の図は元の入力信号を、下段の図はそれに対応するDeltaを示しています。
  (a) 2つのDeltaの差が *Threshold Delta* を超えているため、値 y_5 を処理した後に y_4 が最大値として識別されます。
  (b) Deltaの差が *Threshold Delta* 未満のため、最大値は識別されません。
  (c) 値 y_6 を処理した後、y_3 が最大値として識別されます。



*Threshold Delta*に関係なく、2つの反転のDeltaのうち少なくとも1つの最大値が検出されます。

出力値

- **Min :** 入力配列の最小値。
- **Idx Min:** *Min*の配列インデックス。インデックスは 1 から始まります。
- **Max :** 入力配列の最大値。
- **Idx Max:** *Max*の配列インデックス。インデックスは 1 から始まります。
- **Max Delta :** 入力配列における、連続する2つの値の差の絶対値のうち、最大のもの。
- **Idx Max Delta:** *Max Delta*の配列インデックス。インデックスは1から始まります。
- **Count Peaks :** 確認されたpeak（極大値）の総数。
- **Count Valleys:**確認されたvalley（極小値）の総数。
- **Sum:**入力配列全体の合計。
- **Mean:**入力配列全体の平均値。
- **Standard Deviation:**入力配列全体の標準偏差。

HMIコントロール

Array Statistics アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.SingleValueコントロールは出力値 Sumを表示します。

2.SingleValueコントロールは出力値Meanを表示します。

3.SingleValueコントロールは、出力値 Standard Deviationを表示します。

4.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Min、Max、Max Delta、CountPeaks Count Valleys、Sum、Mean、Standard Deviation。

Data Table Vertical

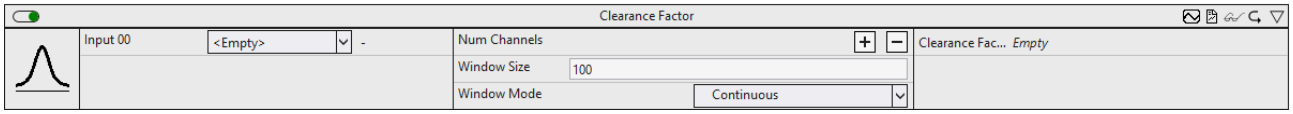
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをStandard Deviationアルゴリズムにマッピングすることもできます。

6.5.10.2 Clearance Factor



*Clearance Factor*アルゴリズムは、入力値から同名の信号特徴量を計算します。出力値は、入力信号のピーク値を、入力信号の絶対値の平方根の2乗平均で割った比率として計算されます。

$$\text{Clearance Factor} = \frac{\max(|x|)}{\left(\frac{1}{N}\sum_{n=1}^{N}\sqrt{|x[n]|}\right)^2}$$

計算に含める入力データの数や計算の種類を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 独立した入出力チャンネル数の設定。
- **Window Mode:**使用するウィンドウモード。計算に含まれる入力データの数と、計算が実行されるタイミングに影響します。
  Continuous：アルゴリズム開始以降のすべての入力値が計算に含まれます。計算はサイクルごとに実行されます。
  Fix Window：直近の *N* 個の入力値のみが計算に含まれます。計算は *N* 回呼び出されるごとに実行されます。
  Sliding Window：直近の *N*個の入力値のみが計算に含まれます。計算は周期的に行われます。
- **Window Size:** 使用するウィンドウモードに応じて、計算に含める値の数*N* を設定できます。
  *Continuous*のウィンドウモードではこのパラメータは無視されます。

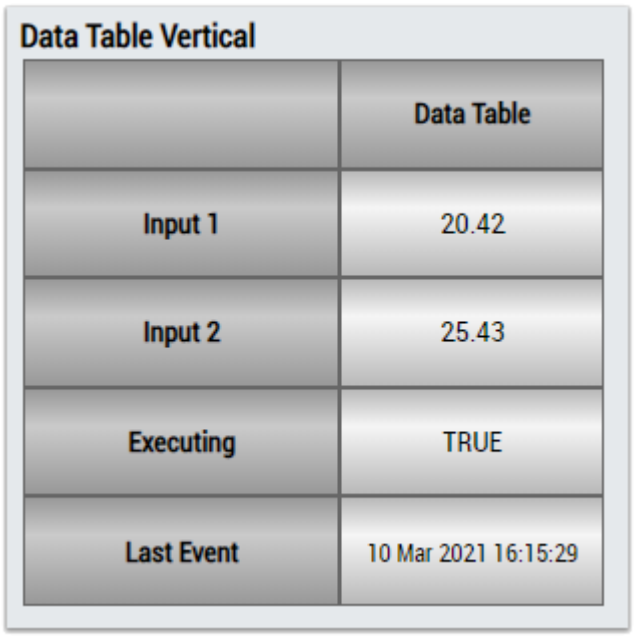
出力値

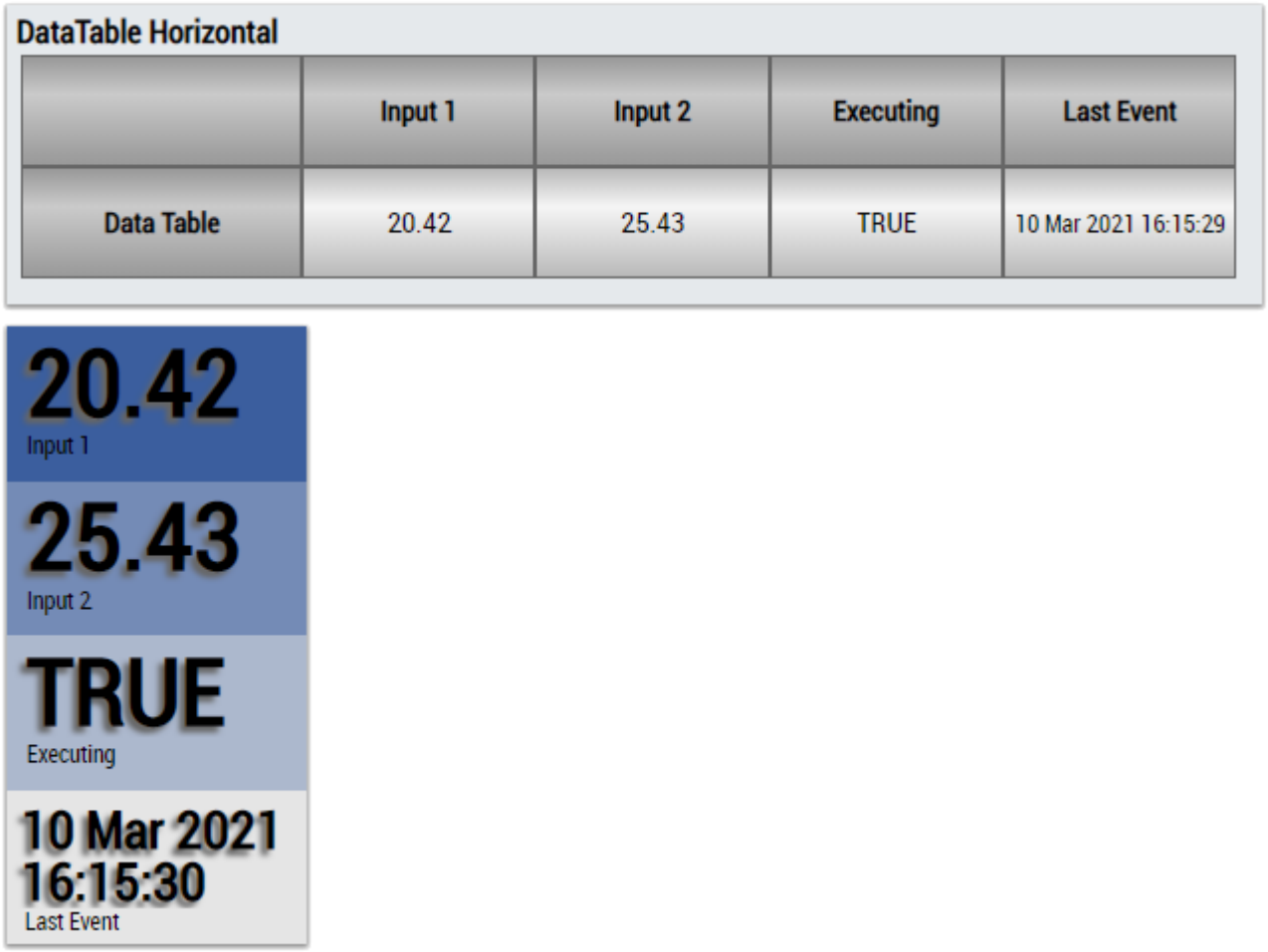
- **Clearance Factor 00..n:**入力チャンネルのクリアランス係数。

HMIコントロール

*Clearance Factor*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

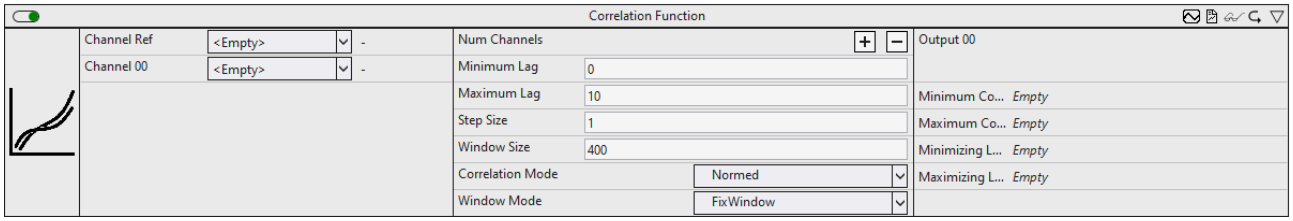
1.TableコントロールまたはMultivalueコントロールは、全ての出力値を表示します。





また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをマッピングすることもできます。

6.5.10.3 Correlation Function



*Correlation Function*ファンクションブロックは、基準信号（Channel Ref）と、1つ以上の他の信号（Channel 00, ..., Channel 0n）の間の離散相関関数を計算します。相関係数は、2つの信号間における *m* サイクル分の時間シフトに対して計算されます。*m* の最小値と最大値は、それぞれパラメータ *Minimum Lag*（負の整数）および *Maximum Lag*（正の整数）によって制限されます。

Step Size パラメータは、2つの連続する相関係数を計算する際に、信号をどれだけのサイクル数だけシフトさせるかを決定します。すなわち、*m*は常に*StepSize*の倍数です。したがって、*Minimum Lag*と*Maximum Lag*では、*StepSize*の倍数のみが許可されます。*StepSize*が1に設定され、*Minimum Lag*が-6、*Maximum Lag*が+4に設定された場合、相関係数は-6、-5、-4、-3、-2、-1、0、+1、+2、+3、+4サイクルのシフトに対して計算されます。*StepSize*を 2 に設定すると、-6、-4、-2、0、+2、+4 サイクルのシフトに対して係数が計算されます。

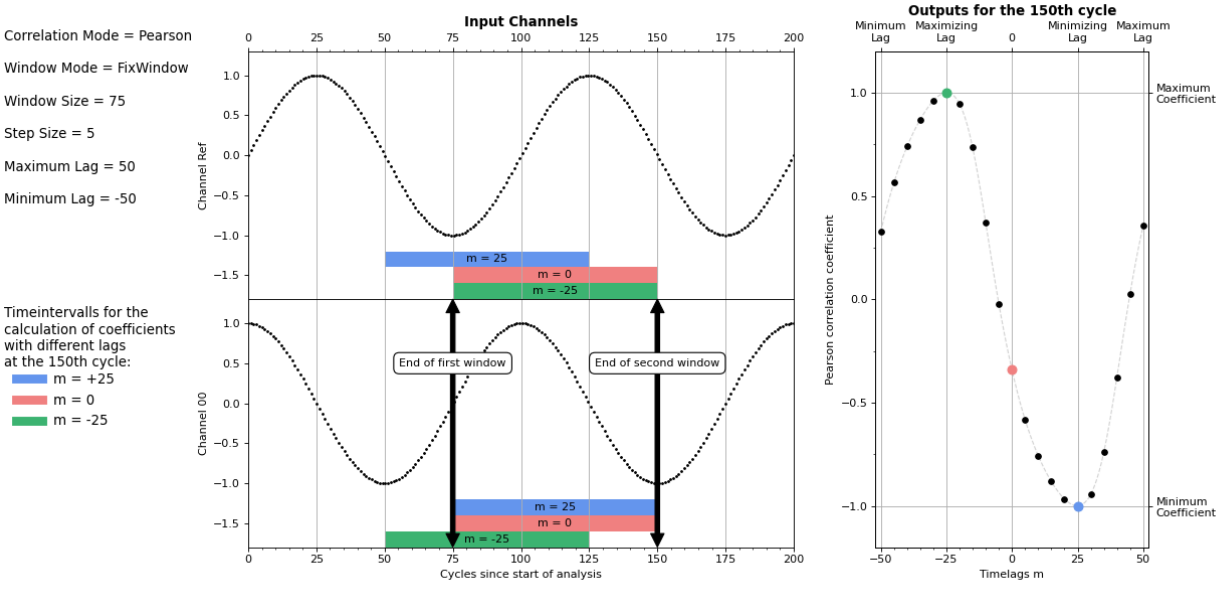
係数は、*Window Mode*パラメータで設定した異なる時間枠で計算できます。*Continuous*モードでは、解析開始以降の全ての値が計算に含まれます。*SlidingWindow*モードでは、計算は*Window Size*で指定された直近のサイクル数に対して継続的に実行されます。*FixWindow*モードでは、*Window Size*で設定したサイクル数でも計算が行われます。ただし、計算は*Window Size*ごとに再実行され、出力値はウィンドウ内の最後のサイクルが処理されたタイミングでのみ更新されます。

*m*が0でない場合、対応する信号のウィンドウは*m*サイクル分シフトします。係数の計算に含まれる値の数は、どの*m*の値でも同じです。唯一の例外は、解析開始後の最初のサイクルの結果です。経過サイクル数が｜*m*｜より小さい場合、それに対応して計算に含まれる値も少なくなります。

*m*が正の値の場合、基準信号（Channel Ref）の値はリングメモリに保存され、*m*サイクル前に受信された基準信号のそれぞれの値を、Channel 00からChannel 0nまでの現在の値と比較できます。これは、基準信号が過去にシフトすることになります。*m*が負の値の場合、基準信号はそれに応じて未来に移動する必要があります。これは明らかに不可能なので、2番目の信号（Channel 00, .. Channel0n）が保存され、代わりに後方に移動します。

相関係数は、様々な計算ルールにしたがって計算できます。これは*Correlation Mode*で設定します。*Base*モードと*Normed*モードでは、係数は信号処理における定義にならい、畳み込みを用いた相関の計算に基づいて算出されます。*Normed*モードでは、係数は加算項の数でも割られます。*Covariance*と*CovarianceBessel*は、それぞれベッセル補正なし/ありで共分散を計算します。*Pearson*モードは、統計学で一般的に使用されるピアソン相関係数の定義を使用しています。正確な計算ルールは、設定オプション内に数式を用いて記載されています。ここで、x_n は基準信号の値、y_nはn番目のサイクル（*Enable Execution* = FALSE のサイクルを除く）における2番目の信号（Channel00 ～ Channel0n）の値を示します。タイムスタンプt_nは解析の開始またはリセット以降のサイクル位置に対応します。*N*の値は選択した*WindowMode*によって異なります。*SlidingWindow*モードおよび*FixWindow*モードでは、解析開始から対応するサイクル数がすでに経過しており、x_{n-N-m}（または y_{n-N+m}）が記録されている場合に限り、*N*は*WindowSize*と等しくなります。そうでない場合、*N*はそれぞれ *n–m+1* または *n+m+1* に減少します。*FixWindow*モードでは、出力値は*WindowSize*サイクルごとにしか更新されないことに注意してください。*Continuous*モードでは、*N* = *n+1*が常に適用されます。

図では、あるサイクル（n = 150）における2つの信号（Channel RefとChannel 00）のファンクションブロックの出力値が、設定例として示されています（*Correlation Mode=Pearson*、*Window Mode=FixWindow*、*Window Size*= 75、*Step Size*= 5、*Maximum Lag*= 50、*Minimum Lag*= -50）。左の2つのプロットでは、入力信号Channel RefとChannel 00が経時的に示されています。右のプロットは離散相関ファンクション（mに対するピアソン相関係数）を示しています。係数は、シフトm = -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, +5, +10, +15, +20, +25, +30, +35, +40, +45, +50の場合を示します。これらはファンクションブロックの配列として出力されます。横軸には、Minimum LagとMaximum Lagの2つのパラメータに加えて、それぞれの出力値が記されています。縦軸には、対応する係数であり、ファンクションブロックの出力でもあるMinimum CoefficientとMaximum Coefficientが記されています。入力チャンネルのプロット（左）において、シフトm＝-25、m＝0、m＝＋25の場合、それぞれの係数の計算に含まれる時間範囲がハイライトされています。右のプロットでは、対応するポイントに色付けされています。



オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 基準信号と相関関係にあるチャンネル数。これは、Add/Remove Channelで設定できます。
- **Maximum Lag:**2つの信号間の相関を計算する際に適用する、最大のシフトサイクル数を指定します。これは正の整数です。
- **Minimum Lag:**2つの信号間の相関を計算する際に適用する、最小のシフトサイクル数を指定します。これは負の整数です。
- **Step Size:**2つの連続する相関係数を計算する際に、信号をどれだけのサイクル数だけシフトさせるかを決定します。
- **Window Size:**ウィンドウモードSlidingWindowおよびFixWindowの場合、係数を計算するサイクル数を指定します。*Continuous*ウィンドウモードでは、ウィンドウの効果はなく、設定できません。*SlidingWindow*モードでは、Reference ChannelからのMaximum Lag値、Channel 00～Channel 0nのMinimum Lag値に加え、全てのチャンネルのWindow Size値がバッファリングされます。したがって、これらのパラメータを設定する際には、ルーターのメモリサイズを考慮する必要があります。
- **Correlation Mode:**以下のいずれかの定義にしたがって相関係数が計算されます：
- *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 :** 係数の計算に使用するウィンドウのタイプを指定します。

  Continuous：解析開始以降のすべての値が、等しい重みで分析に含まれます。

  SlidingWindow：計算にはウィンドウの大きさWindow Sizeを使用します。現在値は常に解析に含まれ、出力はサイクルごとに更新されます。

  FixWindow：出力はWindow Sizeごとに更新され、Window Sizeの長さのウィンドウを用いて計算されます。

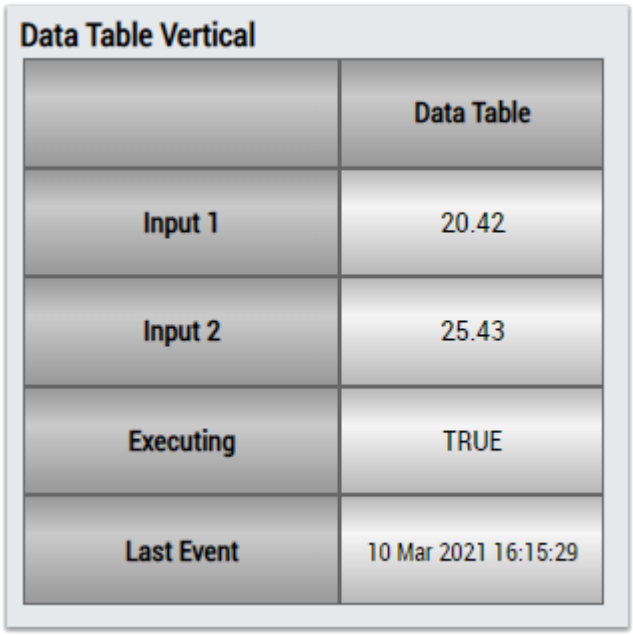
出力値

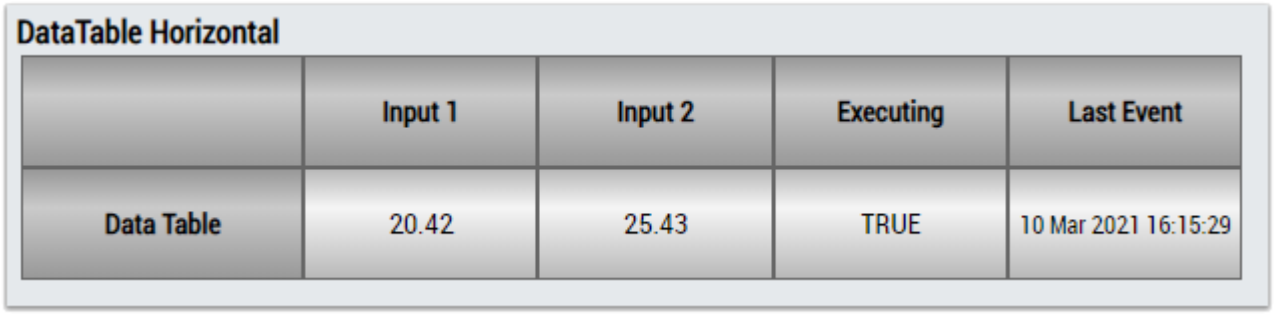
- **Output00 .. Output0n :** 各入力(Channel00, ..., Channel0n)に対して、*m=Minimum Lag*,*m=Minimum Lag+Step Size*, ...,*m= -Step Size*,*m=* 0,*m=+StepSize*, ..., m =*Maximum Lag*のシフト係数を配列として示します。
- **Minimizing Lag00..Minimizing Lag0n:** 各チャンネルについて、係数が最小になるシフトを指定します。
- **Minimum Coefficient00..Minimum Coefficient0n:** 各チャンネルの最小係数を指定します。
- **Maximizing Lag00...Maximizing Lag0n :** 各チャンネルについて、係数が最大になるシフトを指定します。
- **Maximum Coefficient 00..00.. Maximum Coefficient0n:**各チャンネルの最大係数を指定します。

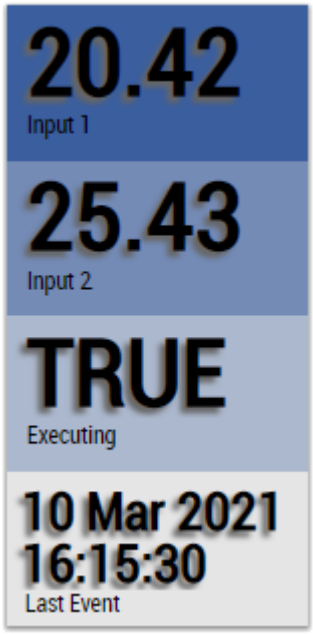
HMIコントロール

Correlation Functionアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：minimum coefficient, maximum coefficient, minimum lag, maximum lag and the coefficient array。

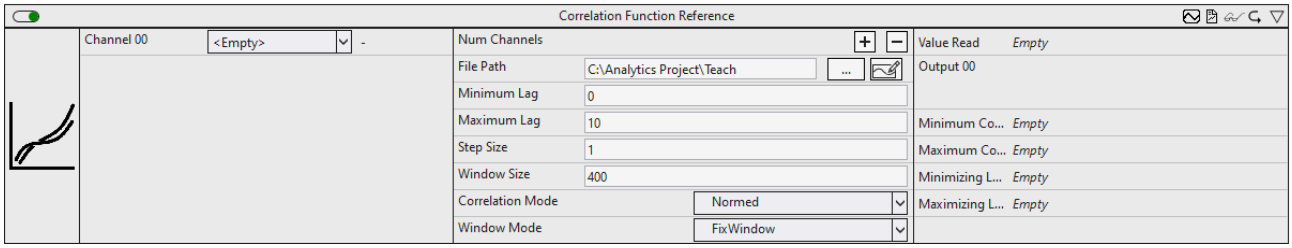






また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMI コントロールをCorrelation Functionアルゴリズムにマッピングすることもできます。

6.5.10.4 Correlation Function Reference



*Correlation Function Reference*ファンクションブロックは、tcabファイルから読み込まれた記録信号（以下、基準信号と呼ぶ）と、1つ以上の入力信号（Channel 00, ..., Channel 0n）との離散相関関数を計算します。

相関係数は、2つの信号間における *m* サイクル分の時間シフトに対して計算されます。*m* の最小値と最大値は、それぞれパラメータ *Minimum Lag*（負の整数）および *Maximum Lag*（正の整数）によって制限されます。

Step Size パラメータは、2つの連続する相関係数を計算する際に、信号をどれだけのサイクル数だけシフトさせるかを決定します。すなわち、*m*は常に*Step Size*の倍数です。したがって、*Minimum Lag*と*Maximum Lag*についても、*Step Size*の倍数のみが許容されます。例えば、*Step Size*を1、*Minimum Lag*を-6、*Maximum Lag*を+4に設定した場合、-6、-5、-4、-3、-2、-1、0、+1、+2、+3、+4サイクルのシフトに対して係数が計算されます。*Step Size*を 2 に設定すると、-6、-4、-2、0、+2、+4 サイクルのシフトに対して係数が計算されます。

解析開始時から、1サイクルごとに読み込まれた信号の値が処理されます。ファイルの末尾に達すると、ファイルの最初の値から処理を再開します。したがって、読み込まれる信号は周期的であると仮定されます。入力信号の特定の期間だけを基準信号と相関させたい場合は、*Enable Execution*入力と*Reset* 入力、および*New Result*出力で制御できます。

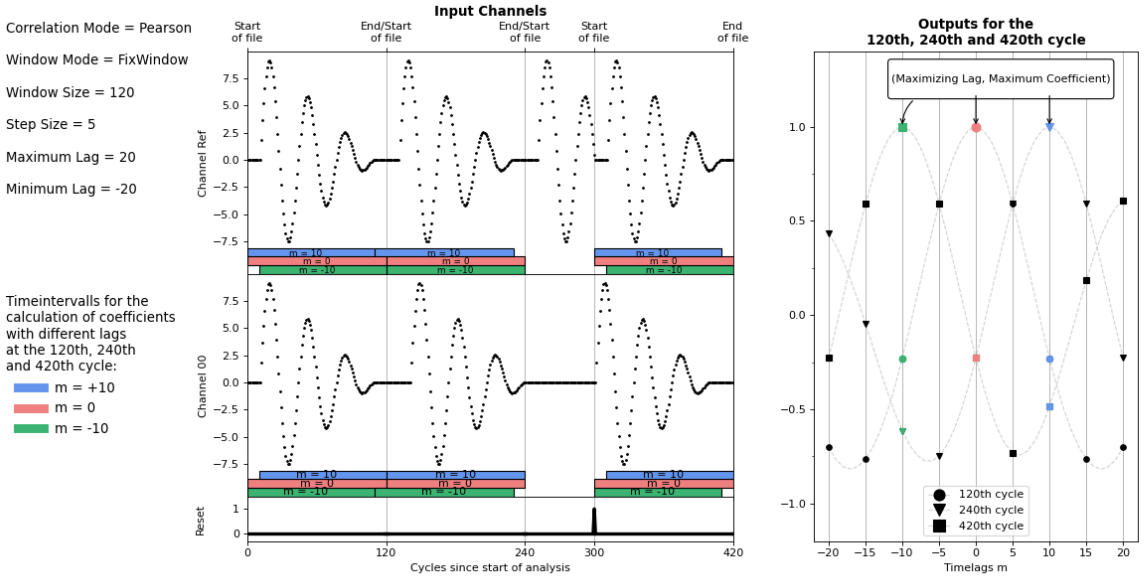
相関係数は、*Window Mode*パラメータで設定された異なる時間枠で計算できます。*Continuous*モードでは、解析開始以降の全ての値が計算に含まれます。*SlidingWindow*モードでは、計算は*Window Size*で指定された直近のサイクル数に対して継続的に実行されます。*FixWindow*モードでは、計算は基準信号の長さにわたって行われ、出力値は常に記録されたシーケンスの最後で更新されます。

m が0でない場合、対応する信号のウィンドウは*m*サイクル分シフトします。係数の計算に含まれる値の数は、どの*m*の値でも同じです。唯一の例外は、解析開始後の最初のサイクルの結果です。経過サイクル数が｜*m*｜より小さい場合、それに対応して計算に含まれる値も少なくなります。

*m*が正の値の場合、基準信号（Channel Ref）の値はリングメモリに保存され、*m*サイクル前に受信された基準信号のそれぞれの値を、Channel 00からChannel 0nまでの現在の値と比較できます。これは、基準信号が過去にシフトすることになります。*m*が負の値の場合、基準信号はそれに応じて未来に移動する必要があります。これは明らかに不可能なので、2番目の信号（Channel 00, .. Channel0n）が保存され、代わりに後方に移動します。

相関係数は、様々な計算ルールにしたがって計算できます。これは*Correlation Mode*で設定します。*Base*モードと*Normed*モードでは、係数は信号処理における定義にならい、畳み込みを用いた相関の計算に基づいて算出されます。*Normed*モードでは、係数は加算項の数でも割られます。*Covariance*と*CovarianceBessel*は、それぞれベッセル補正なし/ありで共分散を計算します。*Pearson*モードは、統計学で一般的に使用されるピアソン相関係数の定義を使用しています。正確な計算ルールは、設定オプション内に数式を用いて記載されています。ここで、x_n は基準信号の値、y_nはn番目のサイクル（*Enable Execution* = FALSE のサイクルを除く）における入力信号（Channel00 ～ Channel0n）の値を示します。タイムスタンプt_nは解析の開始またはリセット以降のサイクル位置に対応します。*N*の値は選択した*WindowMode*によって異なります。*SlidingWindow*モードおよび*FixWindow*モードでは、解析開始から対応するサイクル数がすでに経過しており、x_{n-N-m}（または y_{n-N+m}）が記録されている場合に限り、*N*は*WindowSize*と等しくなります。そうでない場合、*N*はそれぞれ *n–m+1* または *n+m+1* に減少します。FixWindowモードでは、ウィンドウサイズは手動で設定するのではなく、読み込まれた信号セクション（基準信号）の長さに対応し、出力値は信号セクションの終了時に更新されます。*Continuous* モードでは、*N = n+1* が常に適用されます。

図は、ある設定（*Correlation Mode = Pearson*、*Window Mode = FixWindow*、*Window Size* = 120（＝ファイル内の値の数）、*Step Size* = 5、*Maximum Lag* = 20、*Minimum Lag* = -20）および入力チャンネルに対する、ファンクションブロック各種入出力値を示しています。左側には、入力信号であるChannel 00および*Reset*が下から2つのプロットに示されており、その上には処理にしたがって読み込まれたシーケンスが、同一のタイムライン上に表示されています。解析開始時に信号の読み込みが開始されます。ファイルの最後の値が120番目のサイクル（または240番目のサイクル）に処理された場合、処理は最初の値から再開されます。300番目のサイクルでは、*reset*が実行され、プロセスはファイルの最初の値から再開します。さらに、内部メモリから全ての値が削除され、係数の計算が再開されます。例えば、ここではChannel 00が直前のサイクルで有効な値を含んでいなかったことが検出され、現在は振動が再通電されていることが確認できます。このエリアでは、*Enable Execution*= FALSEによっても解析が中断される可能性があります。*WindowMode=FixWindow*なので、出力は120番目、240番目、420番目のサイクルでのみ更新されます。（Output00からの）対応する係数が右側のプロットに示されています。値のペア（*Maximizing Lag*、*Maximum Coefficient*）から、入力信号が基準信号に対してどれだけシフトしているかを読み取ることができます。例：420番目のサイクル（*Maximizing Lag*、*Maximum Coefficient*）＝（-10.1）。つまり、*Reset*が10サイクル前に行われていれば、基準信号とChannel 00はこのウィンドウで正確に一致したことになります。



オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 基準信号と相関関係にあるチャンネル数。これは、Add/Remove Channelで設定できます。
- **File Path:** データファイルへのパス。
- **Maximum Lag:** 2つの信号間の相関を計算する際に適用する、最大のシフトサイクル数を指定します。これは正の整数です。
- **Minimum Lag:** 2つの信号間の相関を計算する際に適用する、最小のシフトサイクル数を指定します。これは負の整数です。
- **Step Size:** 2つの連続する相関係数を計算する際に、信号をどれだけのサイクル数だけシフトさせるかを決定します。
- **Window Size:**ウィンドウモード*SlidingWindow*および*FixWindow*の場合、係数を計算するサイクル数を指定します。*Continuous* ウィンドウモードでは、ウィンドウの効果はなく、設定できません。*SlidingWindow*モードでは、Reference ChannelからのMaximum Lag値、Channel 00～Channel 0nのMinimum Lag値に加え、全てのチャンネルのWindow Size値がバッファリングされます。したがって、これらのパラメータを設定する際には、ルーターのメモリサイズを考慮する必要があります。
- **Correlation Mode:**以下のいずれかの定義にしたがって相関係数が計算されます：
- *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 :** 係数の計算に使用するウィンドウのタイプを指定します。

  Continuous：解析開始以降のすべての値が、等しい重みで分析に含まれます。

  SlidingWindow：計算にはウィンドウの大きさWindow Sizeを使用します。現在値は常に解析に含まれ、出力はサイクルごとに更新されます。

  FixWindow：係数は、読み込まれた信号セクションの全長にわたって計算され、最後の値を処理した後に出力が更新されます。

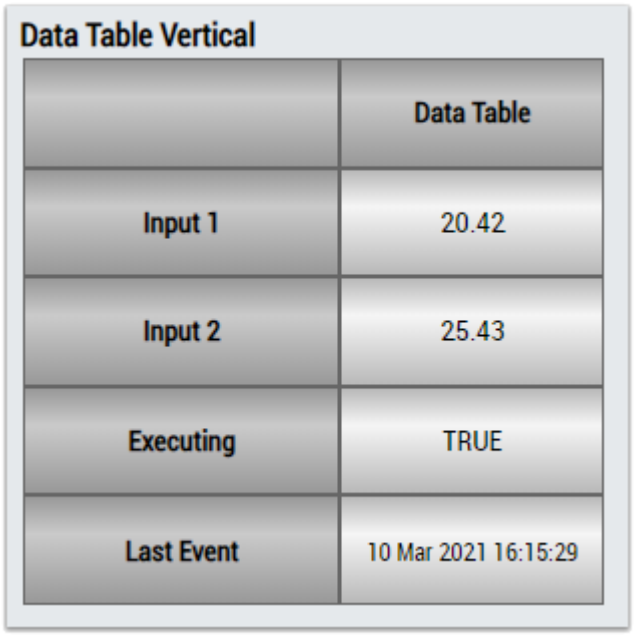
出力値

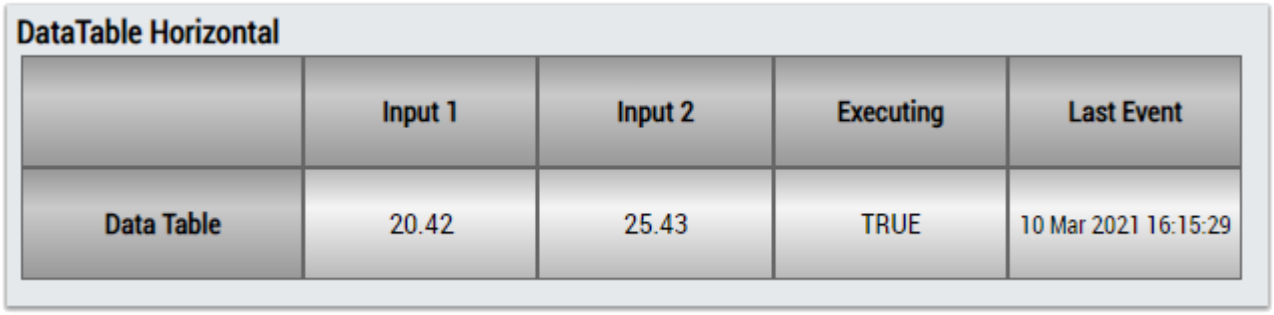
- **Output00 .. Output0n :** 各入力(Channel00, ..., Channel0n)に対して、*m=Minimum Lag*,*m=Minimum Lag+Step Size*, ...,*m= -Step Size*,*m= 0*,*m=+StepSize*, ..., m =*Maximum Lag*のシフト係数を配列として示します。
- **Minimizing Lag00..Minimizing Lag0n:** 各チャンネルについて、係数が最小になるシフトを指定します。
- **Minimum Coefficient00..Minimum Coefficient0n:** 各チャンネルの最小係数を指定します。
- **Maximizing Lag00...Maximizing Lag0n :** 各チャンネルについて、係数が最大になるシフトを指定します。
- **Maximum Coefficient 00..00.. Maximum Coefficient0n:**各チャンネルの最大係数を指定します。

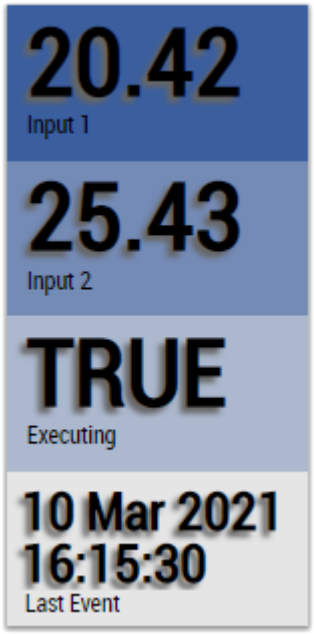
HMIコントロール

Correlation Function Referenceアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：read value, minimum coefficient, maximum coefficient, minimum lag, maximum lag and the coefficient array。

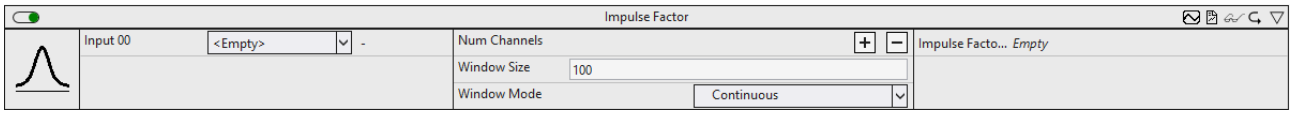






また、Mappingウィザード [▶ 438]を使用して、Correlation Function Referenceアルゴリズムに顧客固有の HMI コントロールをマッピングすることもできます。

6.5.10.5 Impulse Factor



アルゴリズム*Impulse Factor*は、入力値から同名の信号特徴量を計算します。出力値は、入力信号の平均値に対する入力信号のピーク値の比率から計算されます。

$$\text{Impulse Factor} = \frac{\max(|x|)}{\frac{1}{N}\sum_{n=1}^{N}|x[n]|}$$

計算に含める入力データの数や計算の種類を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 独立した入出力チャンネル数の設定。
- **Window Mode:**使用するウィンドウモード。計算に含まれる入力データの数と、計算が実行されるタイミングに影響します。
  Continuous：アルゴリズム開始以降のすべての入力値が計算に含まれます。計算はサイクルごとに実行されます。
  Fix Window：直近の *N* 個の入力値のみが計算に含まれます。計算は *N* 回呼び出されるごとに実行されます。
  Sliding Window：直近の *N*個の入力値のみが計算に含まれます。計算は周期的に行われます。
- **Window Size:** 使用するウィンドウモードに応じて、計算に含める値の数*N* を設定できます。
  *Continuous*のウィンドウモードではこのパラメータは無視されます。

出力値

- **Impulse Factor 00..n:**入力チャンネルのインパルス係数。

HMIコントロール

*Impulse Factor*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.TableコントロールとMultivalueコントロールは、全ての出力値を表示します。

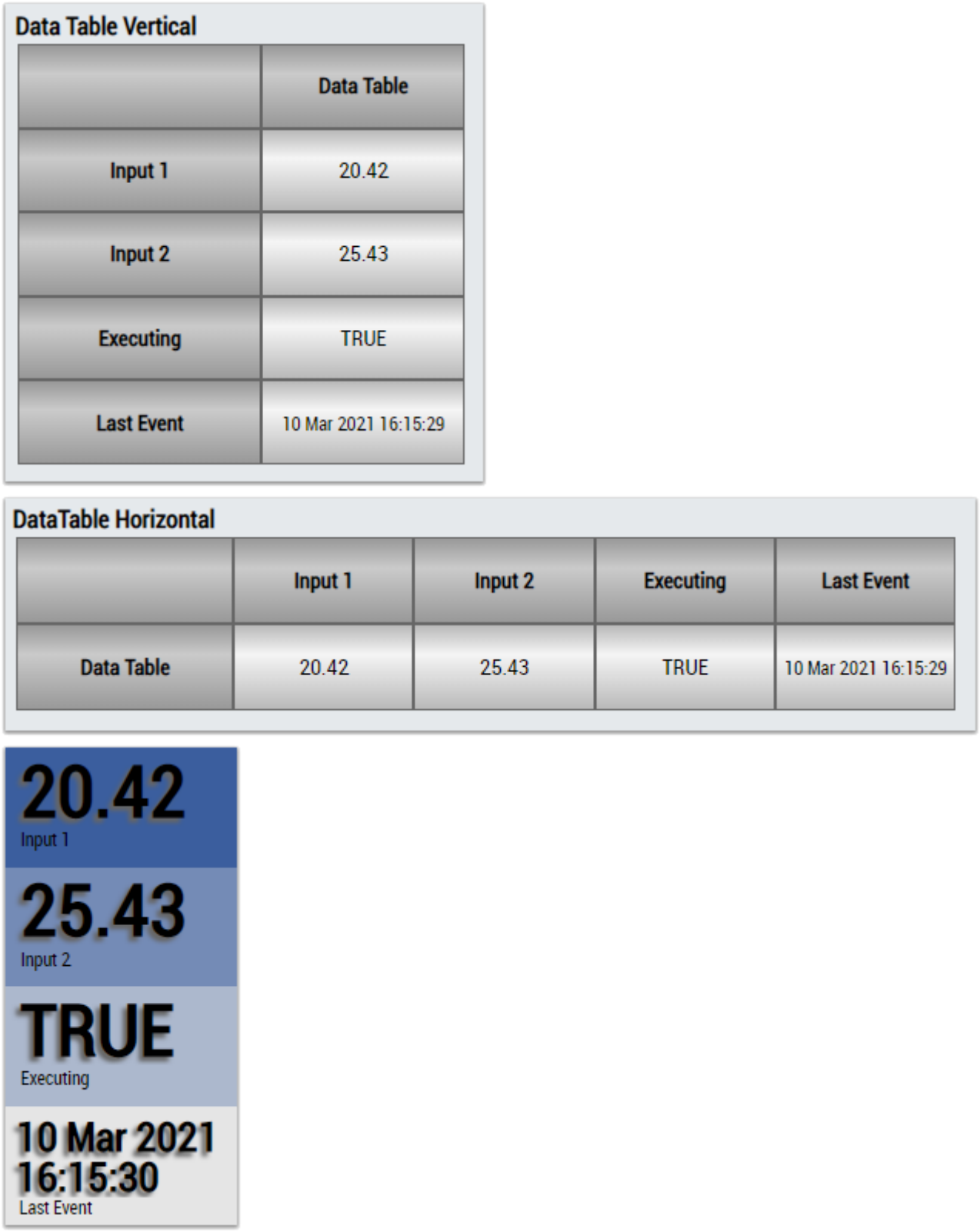
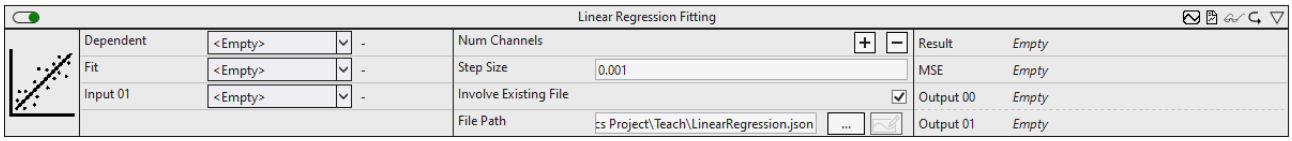


図 3:

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをマッピングすることもできます。

6.5.10.6 Linear Regression Fitting



Linear Regression Fittingファンクションブロックは，1つの変数（Dependent入力）を，他の複数の変数（Input 01 ... の線形結合で近似する． これはインクリメンタル確率的勾配法によって行われます。解析が終わると、計算された係数がファイルに書き出されます。

線形結合は以下の式で提示されます：

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i \times \text{Input 0i}$$

各サイクルにおいて、 β_0 から β_n の値は以下のルールで再計算されます：

$$\beta_i = \begin{cases} \beta_i - \gamma \times \text{Input 0i} \times (y - \text{Dependent}), & i = 1, 2, \ldots, n \\ \beta_i - \gamma \times (y - \text{Dependent}), & i = 0 \end{cases}$$

これは、対応するDependent入力値と、ファンクションブロックによって結果として出力される計算値 y

との二乗誤差を最小化することに相当します。パラメータ γ はstep sizeに相当し、パラメータの調整の度合いを指定します。この値が大きいほど、係数は局所最適値へと速やかに収束します。ただし、この値が大きすぎると、アルゴリズムが収束しないことがあります。

通常，Linear Regression Fittingファンクションブロックは，最初にターゲット変数の回帰のための重みを調整するために使用されます。そして、Linear Regression Inferenceファンクションブロックと調整済みの重みを用いることで、入力変数に基づいてターゲット変数を予測することができます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 基準信号と相関関係にあるチャンネル数。これは、Add/Remove Channelで設定できます。
- **File Path:** 近似処理の最後に係数を保存するファイルのパスを指定します。ファイル形式：*json*または*tas*。
- **Involve Existing File :** TRUEを設定すると、解析開始時にファイルから係数値が読み込まれ、さらに調整されます。FALSEの場合、開始時に全ての係数はゼロに設定されます。
- **Step Size:** 新しい計算のたびに、どのくらい係数を調整するかを指定します。
- **Bias（オプション）：** FALSEの場合、Bias Output 00はゼロに設定され、それ以上近似されません。
- **Mini Batch Size（オプション）：**MSE（平均二乗誤差）に基づいて係数を調整する前に、何サイクルを対象にMSEを計算するか指定します。

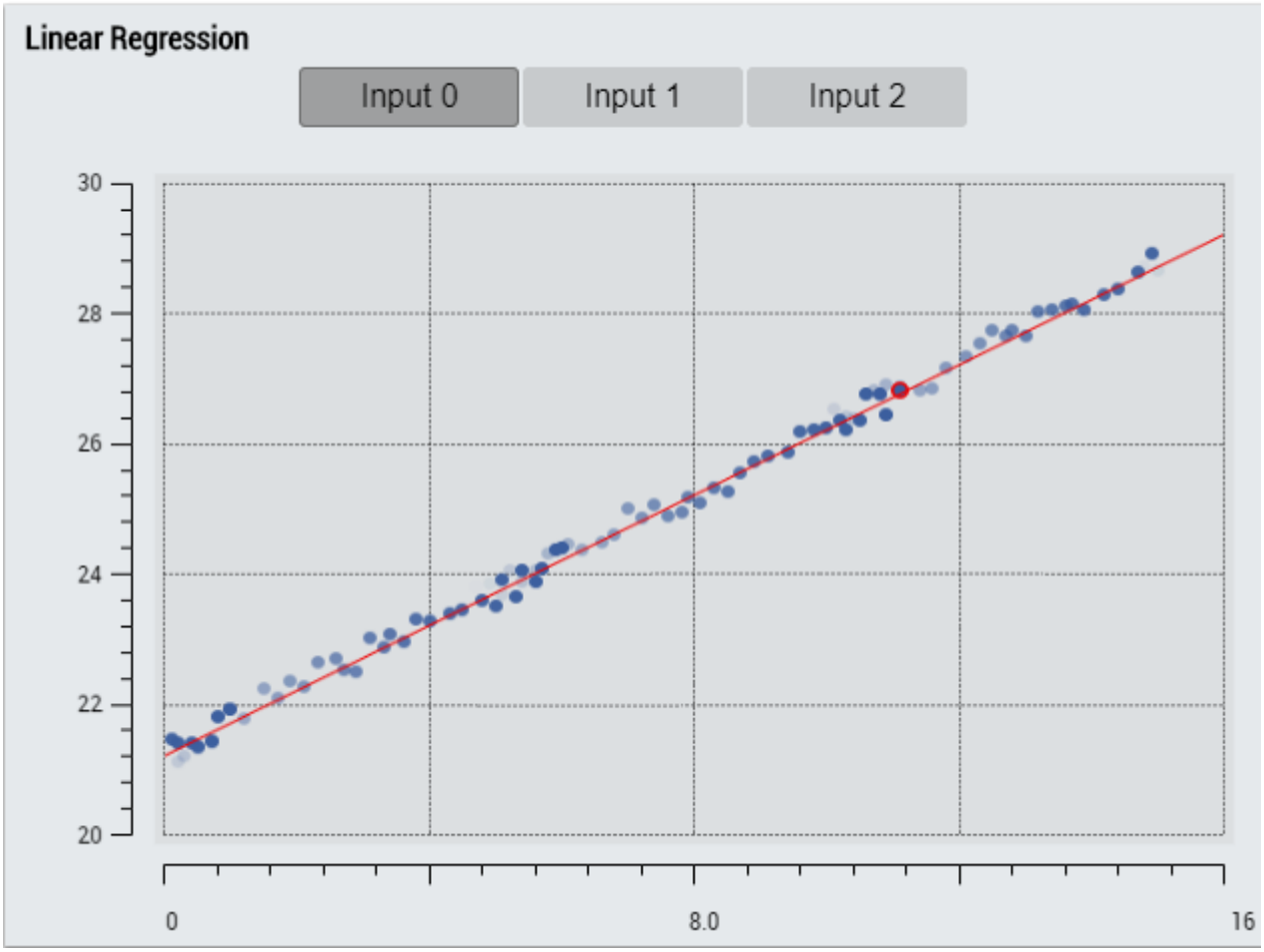
出力値

- **MSE :** 計算された*Result*値と*Dependent*入力値との間のMSE（平均二乗誤差）を示します。
- **Result:** 現在のサイクルの入力の係数を更新した近似値を出力。
- **Output00 .. Output0n :** 計算された係数を表示します。

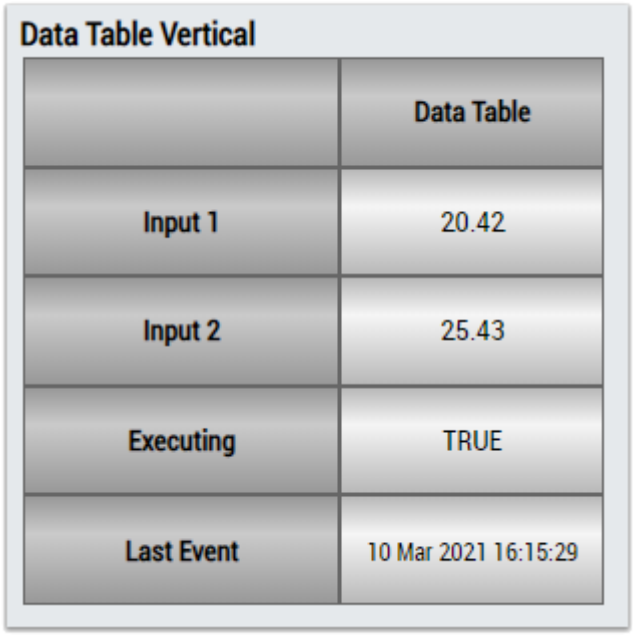
HMIコントロール

Linear Regression Fittingアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.Linear Regressionコントロールは，入力と計算された回帰直線を表示します。ボタンを使って、X軸に表示する入力チャンネルを選択できます。Y軸は回帰の目標値を示します。新しい点は赤い輪郭で表示され、古い点は徐々に薄く表示されます。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：MSE, result, output coefficients.。

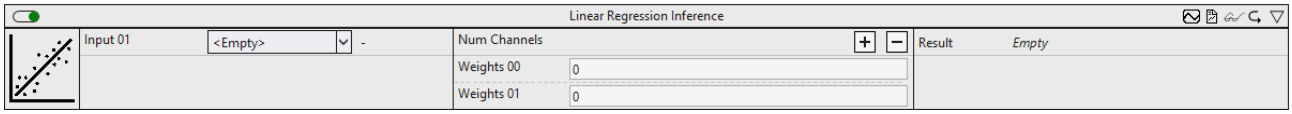


DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをLinear Regression Curve Fittingアルゴリズムにマッピングすることもできます。

6.5.10.7 Linear Regression Inference



Linear Regression Inferenceファンクションブロックは、入力 (Input 01 ... Input 0n）の線形結合を計算します。計算には係数（Weights 00.. Weights 0n)を使用します。

$$\text{Result} = \text{Weights 00} + \sum_{i=1}^{n} \text{Weights 0i} \times \text{Input 0i}$$

パラメータWeights 00からWeights 0nは、手動で設定するか、Linear Regression Fittingファンクションブロックによって生成されたファイルをパラメータフィールドにドラッグ/ドロップして自動で設定できます。通常，Linear Regression Fittingファンクションブロックは，最初にターゲット変数の回帰のための重みを調整するために使用されます。そして、Linear Regression Inferenceファンクションブロックと調整済みの重みを用いることで、入力変数に基づいてターゲット変数を予測することができます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 基準信号と相関関係にあるチャンネル数。これは、Add/Remove Channelで設定できます。

- **Weights00 .. Weights 0n:**計算される線形結合を決定する係数を指定します。
- **File Path（オプション）：** 事前にパラメータ設定したファイルのファイルパス。このファイルは、*Linear Regression Fitting*モジュールにより作成されます。ファイル形式：*json*または *tas*。

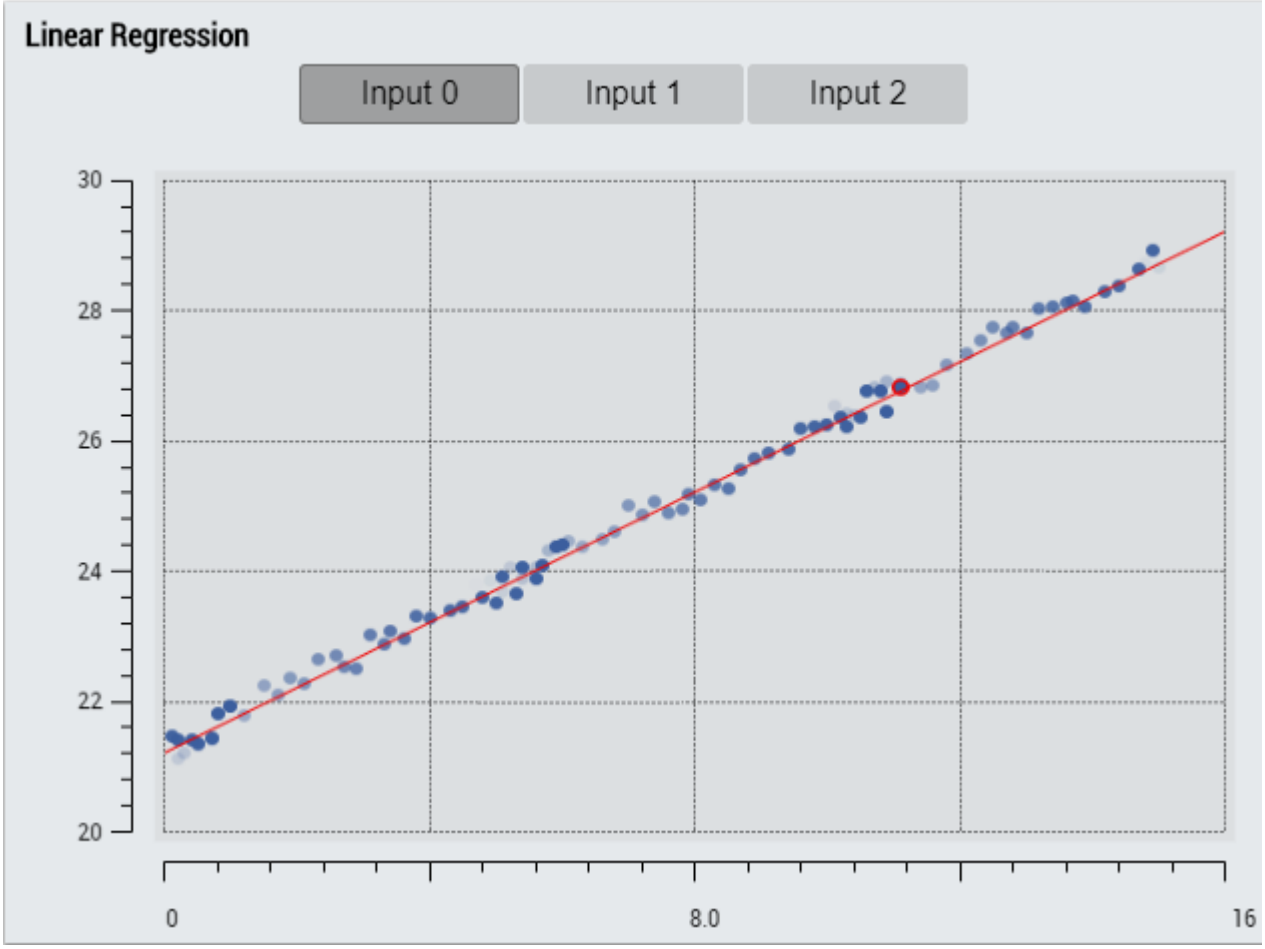
出力値

- **結果：** 線形結合から計算された値を指定します。

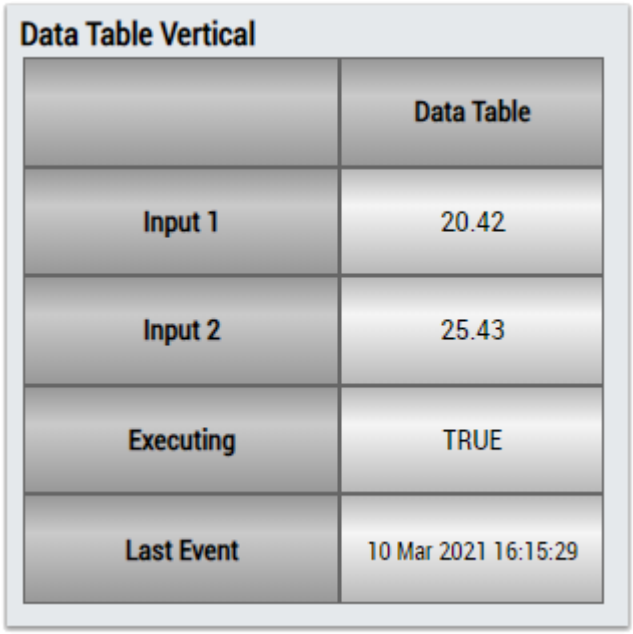
HMIコントロール

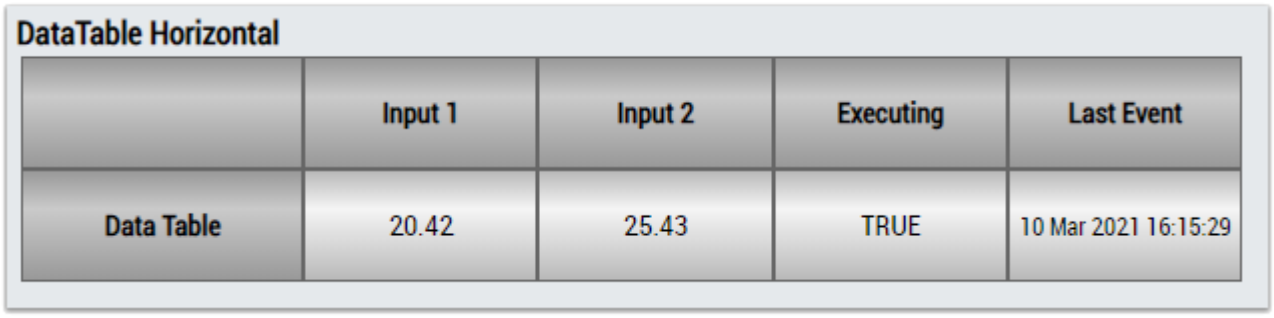
Linear Regression Inferenceアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

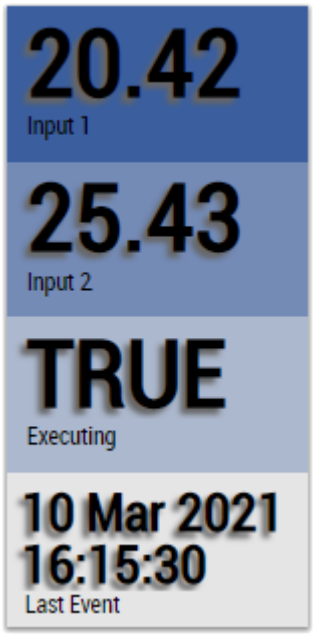
1.Linear Regressionコントロールは，入力と計算された回帰直線を表示します。ボタンを使って、X軸に表示する入力チャンネルを選択できます。Y軸は回帰の目標値を示します。新しい点は赤い輪郭で表示され、古い点は徐々に薄く表示されます。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：MSE, result, output coefficients.。

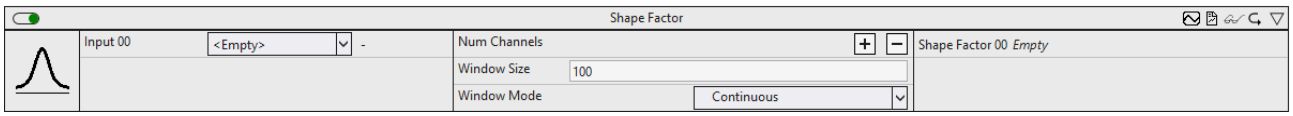






また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをLinear Regression Inferenceアルゴリズムにマッピングすることもできます。

6.5.10.8 Shape Factor



アルゴリズム*Shape Factor*は、入力値から同名の信号特徴量を計算します。出力値は、入力信号の二乗平均平方根（RMS）と、その絶対値の平均値との比から計算されます。

$$\text{Shape Factor} = \frac{x_{rms}}{\frac{1}{N}\sum_{n=1}^{N}|x[n]|}$$

計算に含める入力データの数や計算の種類を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 独立した入出力チャンネル数の設定。
- **Window Mode:**使用するウィンドウモード。計算に含まれる入力データの数と、計算が実行されるタイミングに影響します。
  Continuous：アルゴリズム開始以降のすべての入力値が計算に含まれます。計算はサイクルごとに実行されます。
  Fix Window：直近の *N* 個の入力値のみが計算に含まれます。計算は *N* 回呼び出されるごとに実行されます。
  Sliding Window：直近の *N*個の入力値のみが計算に含まれます。計算は周期的に行われます。
- **Window Size:** 使用するウィンドウモードに応じて、計算に含める値の数*N* を設定できます。
  *Continuous*のウィンドウモードではこのパラメータは無視されます。

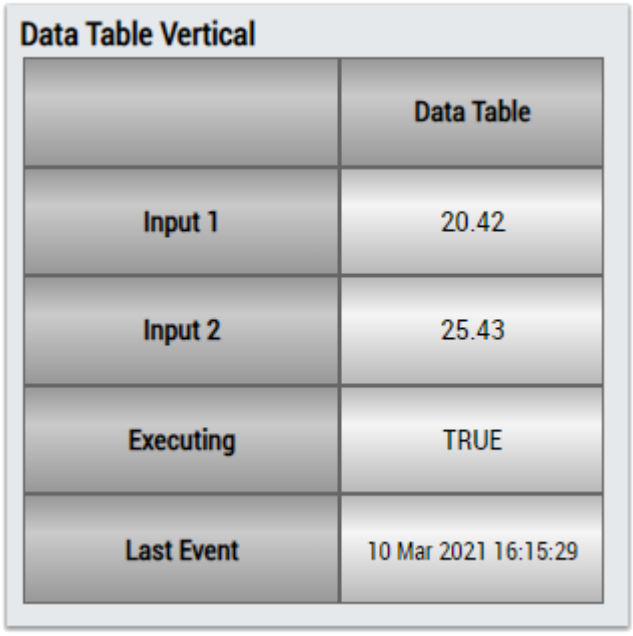
出力値

- **Shape Factor 00..n:**入力チャンネルの形状係数。

HMIコントロール

*Shape Factor*アルゴリズムでは、Analyticsダッシュボードを生成するために以下のHMIコントロールが利用できます：

1.TableコントロールとMultivalueコントロールは、全ての出力値を表示します。



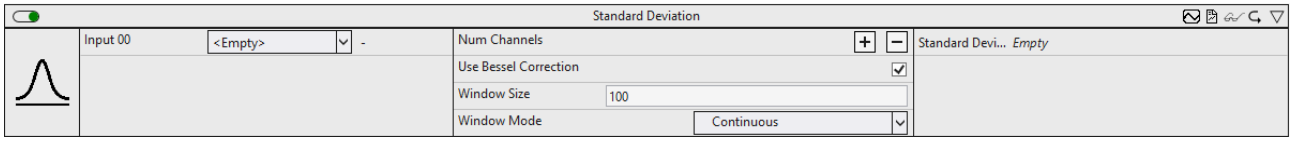
DataTable Horizontal

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

図 4:

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをマッピングすることもできます。

6.5.10.9 Standard Deviation



*Standard Deviation*アルゴリズムは、設定した入力チャンネル数の経験的標準偏差を計算します。計算に含める入力データの数や計算の種類を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Num Channels :** 独立した入出力チャンネル数の設定。
- **Use Bessel Correction:** チェックボックスをオンにすると、ベッセル補正が適用されます。ランダムなサンプルで期待する結果を得るには、このパラメータを有効にする必要があります。

経験的標準偏差（ベッセル補正なし）

$$s' = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x[n] - \overline{x})^2}$$

経験的標準偏差（ベッセル補正あり）

$$s = \sqrt{\frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\bar{x})^2}$$

- **Window Mode:**使用するウィンドウモード。計算に含まれる入力データの量と計算のタイミングに影響します。
  Continuous: アルゴリズムの開始以降の全ての入力値が計算に含まれます。計算は周期的に実行されます。
  Fix Window：最後の*N* 個の入力値のみが計算に含まれます。計算は *N* 回呼び出されるごとに実行されます。
  Sliding Window：直近の *N*個の入力値のみが計算に含まれます。計算は周期的に行われます。
- **Window Size:** 計算に含まれる値の数*N*は、使用するウィンドウモードに応じて設定できます。ウィンドウモード*Continuous*では、このパラメータは無視されます。

出力値

- **Standard Deviation 00..n:**入力チャンネルの経験的標準偏差。

HMIコントロール

標準偏差アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

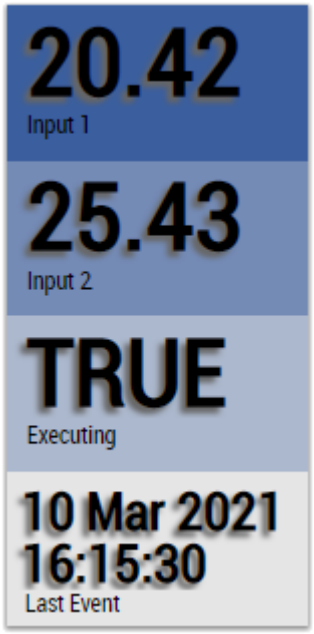
1.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Standard Deviation Results。

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

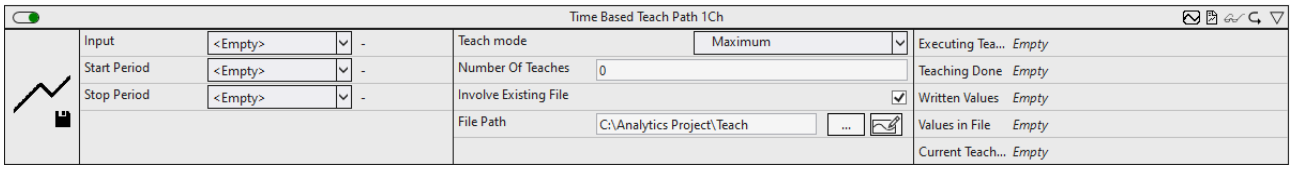


また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMIコントロールをStandard Deviationアルゴリズムにマッピングすることもできます。

6.5.11 Analytics - Training Base（解析 - 教示ベース）

*Analytics - Training Base*カテゴリのアルゴリズムは、周期信号を教示し、そのデータをファイルに書き込むための機能を提供します。後で別の入力信号と比較し、最適な動作との違いを解析できます。

6.5.11.1 Time Based Teach Path 1Ch



*Time Based Teach Path 1Ch*は、設定された教示回数にしたがって、入力データを定期的にファイルに書き込みます。これは、各期間の値が順次書き込まれるのではなく、新しい期間の値が既存の値と比較されることを意味します。周期は、入力値*Start Period*と *Stop Period* により定義できます（ブーリアン信号が必要）。教示モードにしたがって、各値は上書きまたは保持され、その結果、教示された入力信号となり、後にTime Based Envelope 1Ch [▶ 157] アルゴリズムの基準信号などとして使用できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

ファイルアクセスが競合するため、Time Based Teach Path 1ChをTime Based Envelope 1Ch [▶ 157]と同時に使用しないことをお勧めします。代わりに、まず基準信号を学習し、それから初めてTime Based Envelope 1Ch [▶ 157] による評価を実行すべきです。

設定オプション

- **Teach mode:** 教示モード。いずれの基準値（*Minimum*、*Maximum*、*Mean*）で値を比較するかを定義します。*Mean*（平均）の場合は、後の期間の値が総結果に占める比重が大きくならないようにするため、加重平均が計算されます。
- **Number of Teaches:**教示処理を自動的に停止するサイクル数。0に設定すると、連続的に教示処理されます。
- **Involve Existing File:**チェックボックスがチェックされており、既にデータのあるファイルが存在する場合、既存ファイルの値が教示処理に含まれます。チェックされていない場合、既存ファイルは無視され、上書きされます。

- **File Path:** データファイルへのパス。
- **Negate Start Period:** チェックボックスをオンにすると、*Start Period*のブール入力信号が反転されます。
- **Negate Stop Period:**チェックボックスをオンにすると、*Stop Period*のブール入力信号が反転されます。

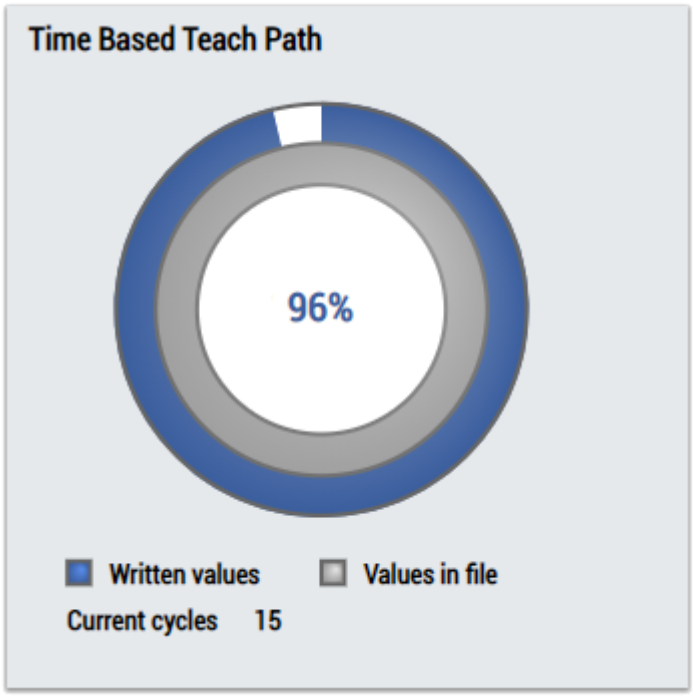
出力値

- **Executing Teach:** 教示処理が実行中かどうかを表示します（開始フラグから停止フラグまでの時間範囲）。
- **Written Values:** 教示処理中に書き込まれた値の合計を表示します。教示サイクルごとに上書きされるファイル内の値の合計と混同しないように注意してください。
- **Values in File :** 現在ファイルに書き込まれている値の数を示します（1サイクル経過後、値は一定になります）。
- **Current Teach Cycles:** ファイル内の教示サイクル数を表示します。

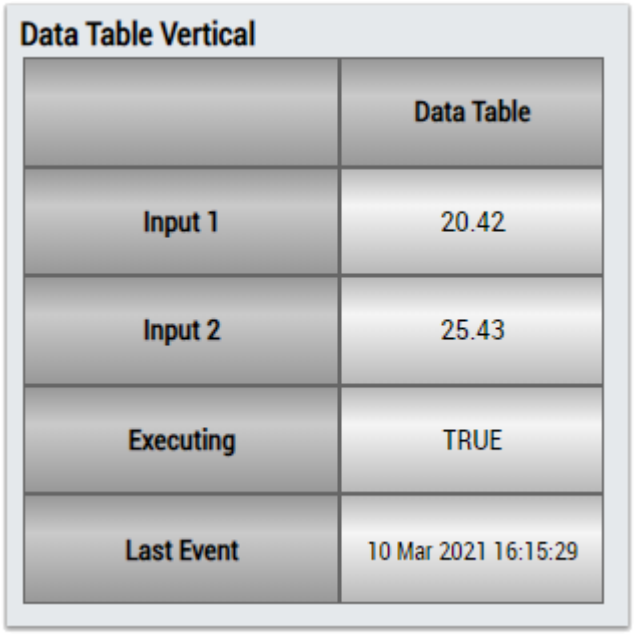
HMIコントロール

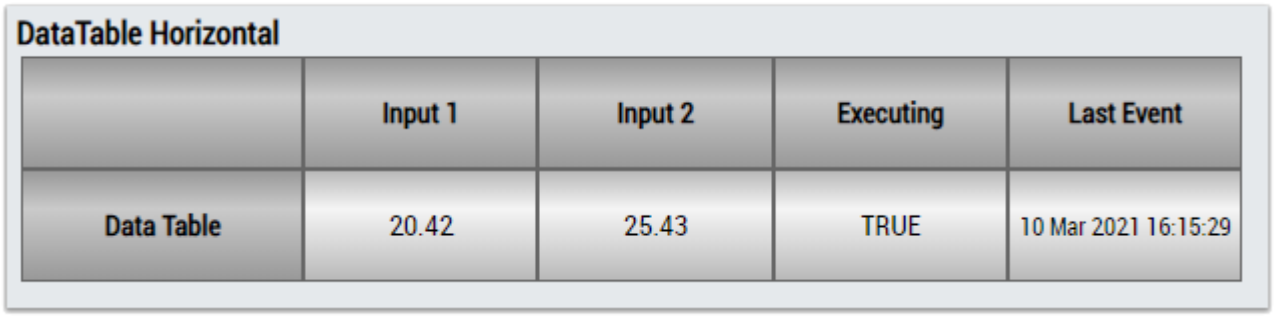
Time Based Teach Path 1Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

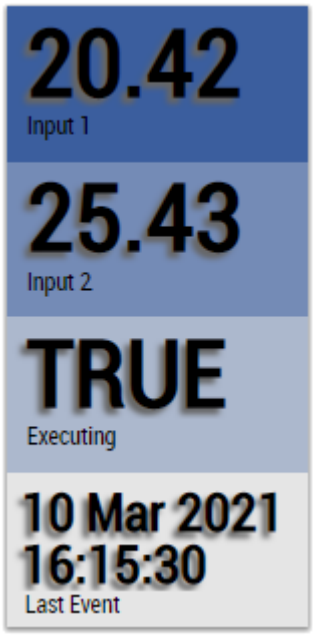
1.TimeBasedTeachPath コントロールは、Written Values、Values in File、および Current Teach Cycles の出力値を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Written Values, Values in File, Current Teach Cycles。







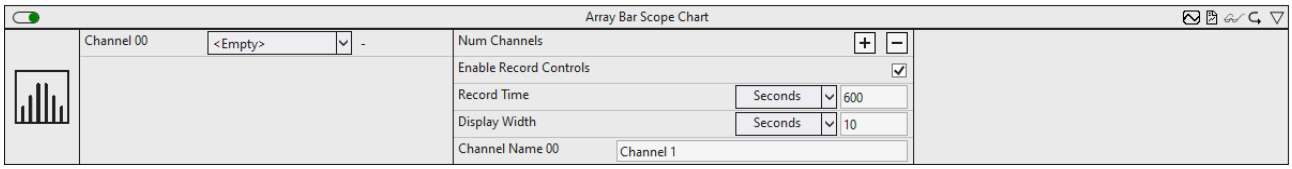
また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをTime Based Teach Path 1Chアルゴリズムにマッピングすることもできます。

6.5.12 Analytics - Visualization Only（解析 - 可視化のみ）

*Visualization Only*カテゴリのアルゴリズムは、HMIダッシュボード上での可視化に使用される補助的なファンクションブロックを表し、後続の可視化処理に必要な機能を提供します。

6.5.12.1 Array Bar Scope Chart

HMIワンクリックダッシュボード用のHMI Scopeコントロールは、HMIバージョン1.12.752.0以降でのみ生成可能です。



*Array Bar Scope Chart*ファンクションブロックは、HMIダッシュボードに入力を表示するために使用します。入力はHMI Scopeコントロールの配列バーチャートに表示されます。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

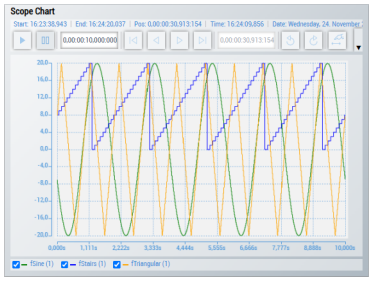
設定オプション

- **Num Channels :** HMI Scopeコントロールに表示する入力の数。
- **Channel Name 00/XX:**入力の表示名。これらは、HMI Scopeコントロールに凡例として表示されます。
- **Enable Record Controls:**チェックボックスをオンにすると、HMIからの記録を開始および停止できます。
- **Record Time :** 記録時間。設定時間に達した後、スコープチャートはリングバッファで実行されます。
- **Display Width:** 選択した画像の表示幅。このプロパティは、HMI Scopeコントロールでも変更できます。

HMIコントロール

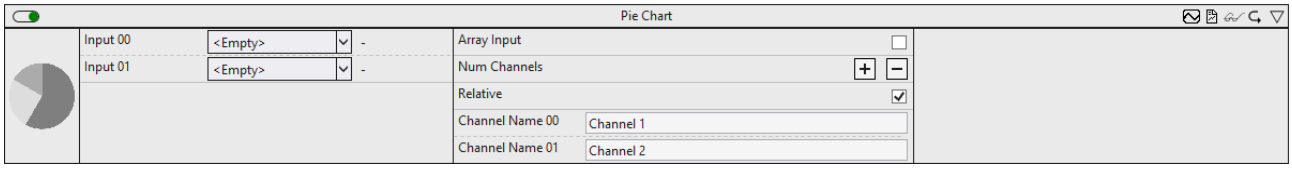
*Array Bar Scope Chart*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

スコープチャートを表示するためのHMI Scopeコントロール。



このファンクションブロックは、HMI Scopeコントロールの表示専用であるため、他の HMI コントロールは追加できません。

6.5.12.2 Pie Chart



*Pie Chart*ファンクションブロックは、後で HMI ダッシュボードで使用するための円グラフを作成できます。円グラフのセグメント数は、パラメータ*Num Channels*で設定できます。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

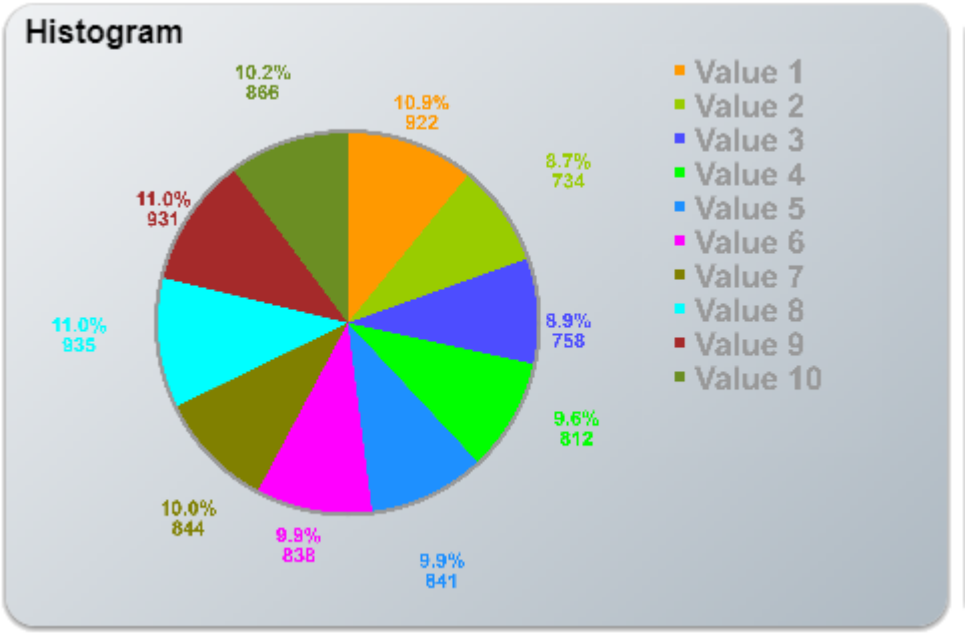
設定オプション

- **Array Input:** 1つの配列を入力として使用するか、複数の独立した入力を使用するかを選択します。配列が最初の入力にドラッグされるとすぐに自動的に設定されます。配列がまだリンクされている限り、パラメータをリセットできません。
- **Num Channels**：設定可能なチャンネル数を指定します。これは円グラフを構成するセグメントの数に相当します。
- **Channel Name 00:**1つ目チャンネルの名前を指定します。ここで割り当てられた名前は、後にHMIダッシュボードでチャンネル名として使用されます。
- **Channel Name 01:** 2つ目のチャンネル名を指定します。ここで割り当てられた名前は、後にHMIダッシュボードでチャンネル名として使用されます。
- **Channel Name xx :** 他の全てのチャンネルについても同様です。

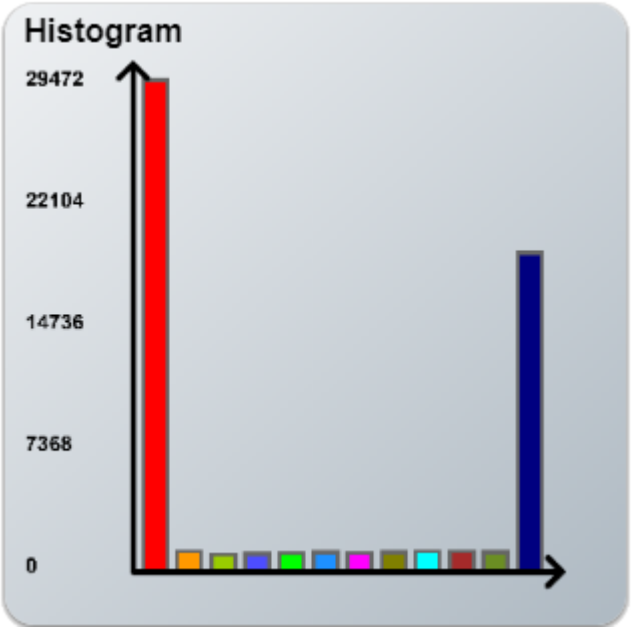
HMIコントロール

*Pie Chart*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

Pie Chartは、円グラフでデータを表示します。

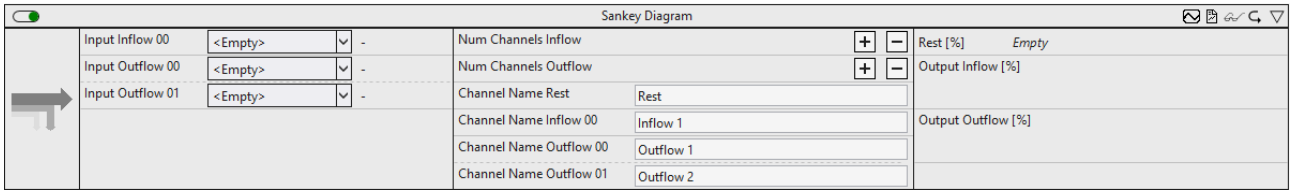


Histogramは、ヒストグラムでデータを表示します。



また、Mappingウィザード [▶ 438]を使用して、顧客固有の HMI コントロールを*Pie Chart*アルゴリズムにマッピングすることもできます。

6.5.12.3 Sankey diagram



*Sankey Diagram*ファンクションブロックは、後で HMI ダッシュボードで使用するためのサンキーダイアグラムを作成できます。流入と流出は区別されます。流入および流出のチャンネル数は、*Num Channels Inflow*および*Num Channels Outflow*パラメータで個別に設定可能です。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

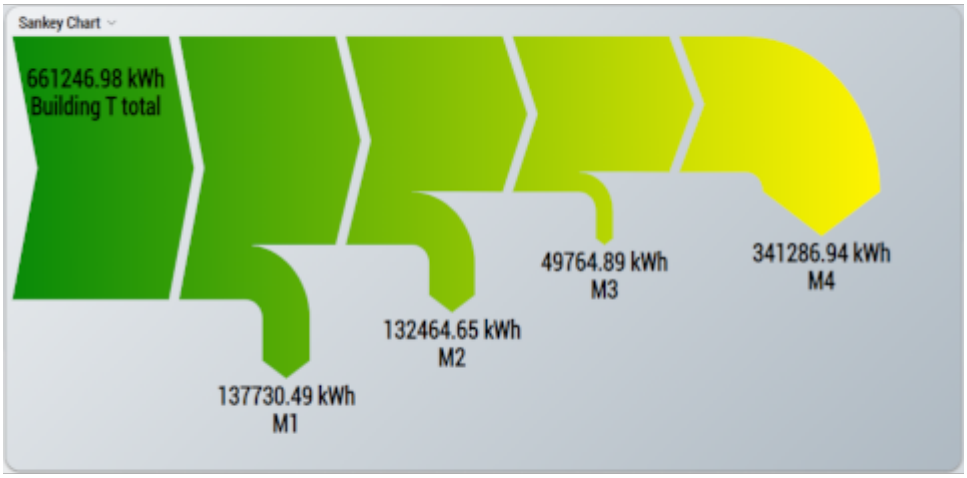
設定オプション

- **Num Channels Inflow**：サンキーダイアグラムの流入チャンネル数を指定します。
- **Num Channels Outflow**：サンキーダイアグラムの流出チャンネル数を指定します。
- **Channel Name Rest:**残りの値（Rest値）を処理するためのチャンネル名を指定します。ここで割り当てられた名前は、後で HMI ダッシュボードで流出チャンネル名として使用されます。
- **Channel Name Inflow 00:** 1つ目の流入チャンネル名を指定します。ここで割り当てられた名前は、後でHMIダッシュボードで流入チャンネル名として使用されます。
- **Channel Name Inflow xx:** 他の全ての流入チャンネル名についても同様に指定します。
- **Channel Name Outflow 00:** 1つ目の流出チャンネル名を指定します。ここで割り当てられた名前は、後で HMI ダッシュボードで流出チャンネル名として使用されます。
- **Channel Name Outflow 01:**2つ目の流出チャンネル名を指定します。ここで割り当てられた名前は、後で HMI ダッシュボードで流出チャンネル名として使用されます。
- **Channel Name Outflow xx:** 他の全ての流出チャンネル名についても同様に指定します。

HMIコントロール

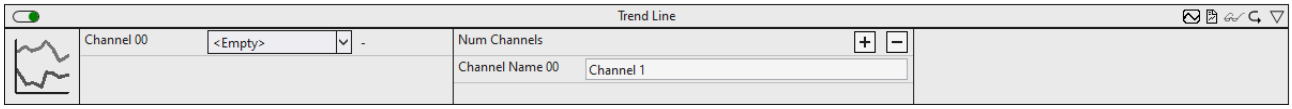
*Sankey Diagram*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

Sankey chartはデータを表示します。



また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールを*Sankey Diagram*アルゴリズムにマッピングすることもできます。

6.5.12.4 Trend Line



*Trend Line*ファンクションブロックは、後で HMI ダッシュボードで使用するための入力信号の傾向線を提供します。NumChannelsでチャンネル数を設定し、任意のチャンネルを選択できます。各チャンネルについて、1分毎に1つの値が保存されます。つまり、このファンクションブロックは信号の転送には適しておらず、ゆっくりと変化する信号の傾向を示すのに適しています。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

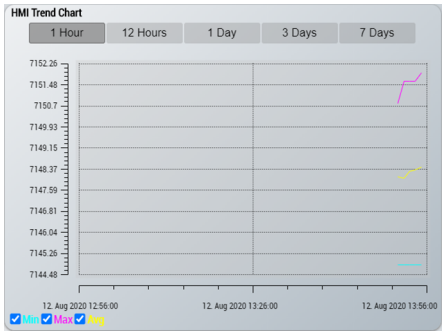
- **Num Channels**：入力信号を転送するために設定可能なチャンネル数を示します。
- **Channel 00:**転送される*1*つ目のチャンネル名を示します。ここで割り当てられた名前は、後にHMIダッシュボードでチャンネル名として使用されます。

- **Channel 01:**転送する2つ目のチャンネル名を示します。ここで割り当てられた名前は、後にHMIダッシュボードでチャンネル名として使用されます。
- **Channel 02:**転送される3つ目のチャンネル名を示します。ここで割り当てられた名前は、後にHMIダッシュボードでチャンネル名として使用されます。
- **Channel Name xx:**他の全てのチャンネルについても同様です。

HMIコントロール

*Trend Line*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

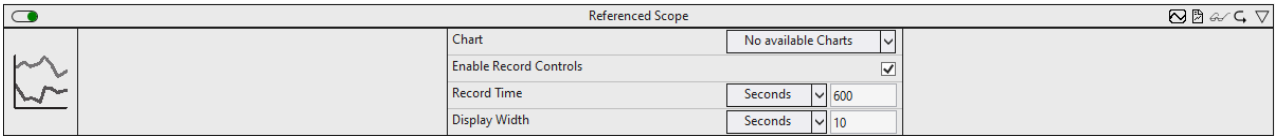
Trend Lineは時間の経過に伴うデータの変化を表示します。



このファンクションブロックは、Trend Lineコントロールの表示専用であるため、他の HMI コントロールは追加できません。

6.5.12.5 Referenced Scope

HMIワンクリックダッシュボード用のHMI Scopeコントロールは、HMIバージョン1.12.752.0以降でのみ生成可能です。



*Referenced Scope*ファンクションブロックは、HMI ダッシュボードに参照スコープからスコープチャートを表示するために使用します。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

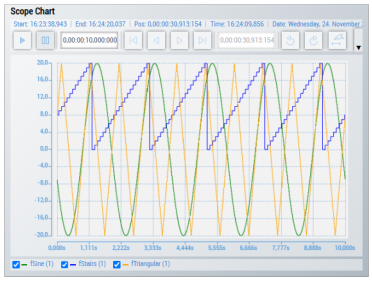
設定オプション

- **Chart:**HMI に表示する参照スコープのチャート。現在、以下のチャートがサポートされています：YT Chart, XY Chart, Array Bar Chart and Single Bar Chart。
- **Enable Record Controls:**チェックボックスをオンにすると、HMIからの記録を開始および停止できます。
- **Record Time :** 記録時間。設定時間に達した後、スコープチャートはリングバッファで実行されます。
- **Display Width:** 選択した画像の表示幅。このプロパティは、HMI Scopeコントロールでも変更できます。

HMIコントロール

*Referenced Scope*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

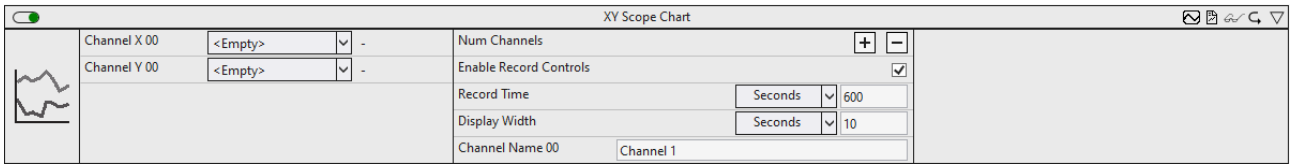
スコープチャートを表示するためのHMI Scopeコントロール。



このファンクションブロックは、HMI Scopeコントロールの表示専用であるため、他の HMI コントロールは追加できません。

6.5.12.6 XY Scope Chart

HMIワンクリックダッシュボード用のHMI Scopeコントロールは、HMIバージョン1.12.752.0以降でのみ生成可能です。



*XY Scope Chart*ファンクションブロックは、HMIダッシュボードに入力を表示するために使用されます。入力はHMI ScopeコントロールのXYチャートに表示されます。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

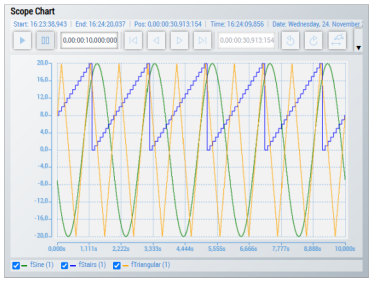
設定オプション

- **Num Channels :** HMI Scope・コントロールに表示するチャンネルまたは入力ペアの数。
- **Channel Name 00/XX:**チャンネルの表示名。これらは、HMI Scopeコントロールに凡例として表示されます。
- **Enable Record Controls:**チェックボックスをオンにすると、HMIからの記録を開始および停止できます。
- **Record Time :** 記録時間。設定時間に達した後、スコープチャートはリングバッファで実行されます。
- **Display Width:** 選択した画像の表示幅。このプロパティは、HMI Scopeコントロールでも変更できます。

HMIコントロール

*XY Scope Chart*アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

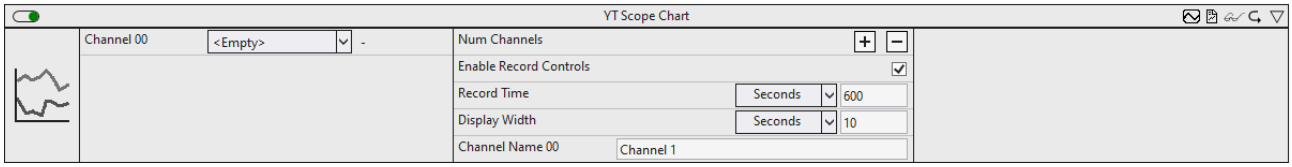
スコープチャートを表示するためのHMI Scopeコントロール。



このファンクションブロックは、HMI Scopeコントロールの表示専用であるため、他の HMI コントロールは追加できません。

6.5.12.7 YT Scope Chart

HMIワンクリックダッシュボード用のHMI Scopeコントロールは、HMIバージョン1.12.752.0以降でのみ生成可能です。



*YT Scope Chart*ファンクションブロックは、HMI ダッシュボードに入力を表示するために使用されます。入力はHMI ScopeコントロールのYTチャートに表示されます。

このファンクションブロックの機能は解析自体には無関係であるため、「Visualization Only（可視化のみ）」のカテゴリに属します。

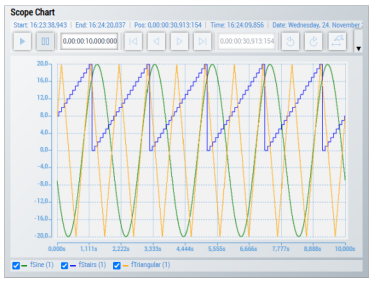
設定オプション

- **Num Channels :** HMI Scopeコントロールに表示する入力の数。
- **Channel Name 00/XX:**入力の表示名。これらは、HMI Scopeコントロールに凡例として表示されます。
- **Enable Record Controls:**チェックボックスをオンにすると、HMIからの記録を開始および停止できます。
- **Record Time :** 記録時間。設定時間に達した後、スコープチャートはリングバッファで実行されます。
- **Display Width:** 選択した画像の表示幅。このプロパティは、HMI Scopeコントロールでも変更できます。

HMIコントロール

アルゴリズム*YT Scope Chart*では、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます。

スコープチャートを表示するためのHMI Scopeコントロール。

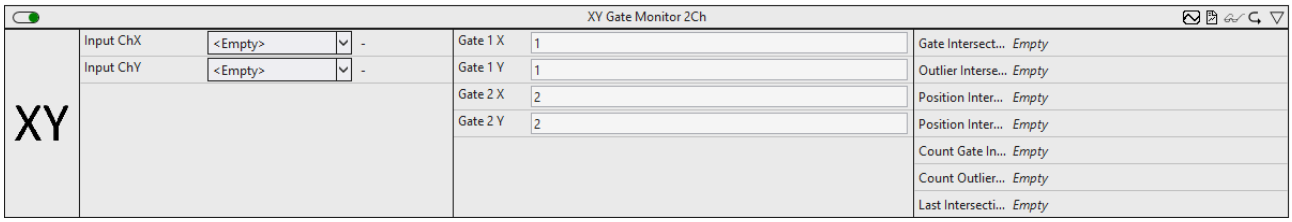


このファンクションブロックは、HMI Scopeコントロールの表示専用であるため、他の HMI コントロールは追加できません。

6.5.13 Analytics - XY Path Analysis（解析 - XYパス解析）

*XY Path Analysis*カテゴリのアルゴリズムは、XYチャンネルの座標評価のための機能を提供します。例えば、入力チャンネルによって決定された座標が、特定の境界や形状の内部にあるかどうか、また境界を越える現象がどの程度の頻度で発生しているかを解析することが可能です。

6.5.13.1 XY Gate Monitor 2Ch

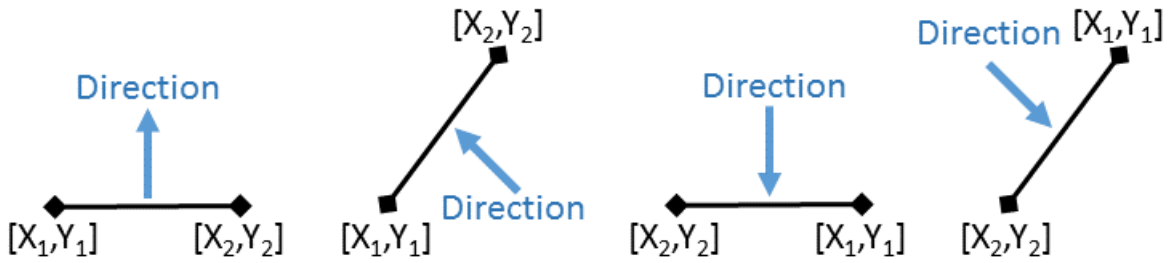


*XY Gate Monitor 2Ch*は、設定されたゲートモードに応じて、XY入力が指定されたゲートまたはその投影（ゲートポイント間の直線）と交差した回数をカウントします。解析期間は、入力*Start*と *Stop*で開始できます。このアルゴリズムは方向を認識します。つまり正しい方向の交点だけがカウントされます。方向の解釈は、ゲートポイント（X1/Y1）と（X2/Y2）の順番に依存します。可能性のある交点の方向を以下に示します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

交点の方向：

青い矢印は信号の方向を表し、黒い線はゲートポイント（X1/Y1）と（X2/Y2）を持つゲートを示しています。交点は、信号が最初のゲートポイント（X1/Y1）を中心に反時計回りの方向に回転するときにカウントされます。



設定オプション

- **Gate Mode:** Gate Monitorのモード：

  Intersect Gate:XY信号が設定された方向でゲートと交差するかどうかを判定します。解析期間中に交点があれば*OK*、なければ*NOK*に分類されます。

  Not Intersect Gate:解析期間中、XY信号が設定された方向でゲートと交差しないかどうかを監視します。交点がなければ*OK*、あれば*NOK*となります。

  Intersect Projection:XY信号が設定された方向でゲートの投影と交差するかどうかを判定します。解析期間中に交点があれば*OK*、なければ*NOK*に分類されます。

  Not Intersect Projection：解析期間中、XY信号が設定された方向でゲートの投影と交差しないかどうかを監視します。交点がなければ*OK*、あれば*NOK*となります。

  Intersect Gate Or Projection：XY信号が設定された方向でゲートまたはその投影と交差するかどうかを判定します。解析期間中に交点があれば*OK*、なければ*NOK*に分類されます。

- **Gate 1 X:**1番目のゲートポイントのX座標。
- **ゲート1 Y：**1番目のゲートポイントのY座標。
- **ゲート2 X：**2番目のゲートポイントのX座標。
- **ゲート2 Y：**2番目のゲートポイントのY座標。

出力値

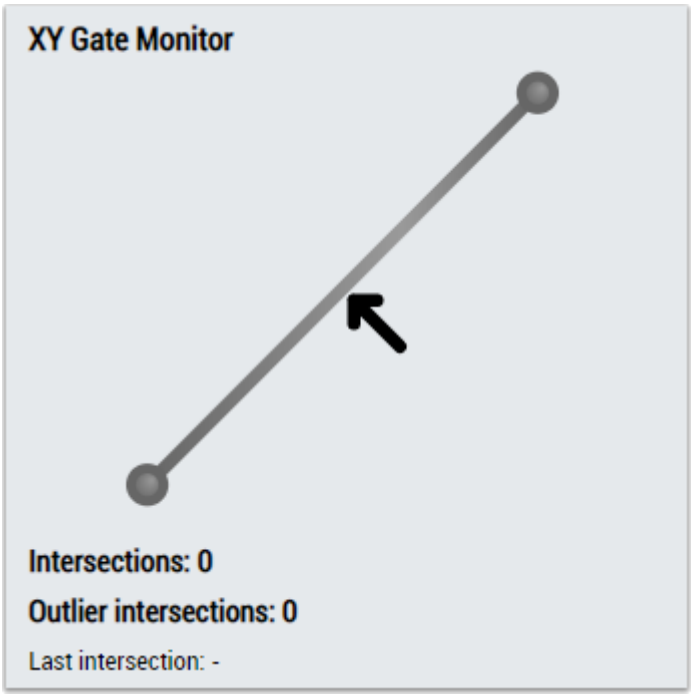
- **Gate Intersection:**現在、ゲートとの交点があるかどうかを示します。
- **Outlier Intersection:**現在、外れ値との交点（ゲートの投影との交点）があるかどうかを示します。
- **Position Intersection X:**最後の交点のX座標。

- **Position Intersection Y:**最後の交点のY座標。
- **Count Gate Intersections:**ゲートとの交点の総数を示します。
- **Count Outlier Intersections:**外れ値との交点の総数を示します。
- **Time Intersection:**最後に交差するイベントが発生した時間を示します → イベントをスコープチャートにドラッグ&ドロップすることで、トリガーイベントとして表示できます。
- **New Result : 新しい結果が計算されたかどうかを示します。**
- **Executing:**アルゴリズムが有効か無効かを示します。
- **Classification:**分類結果を示します。*OK*または *NOK*。上記の通り、分類はゲートモードによって異なります。

HMIコントロール

XY Gate Monitor 2Chアルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

1.XYGateMonitor コントロールは、出力値 Gate Intersection、Outlier Intersection、Count Gate Intersections、Count Outlier Intersections、Last Intersection、および交点の方向を表示します。



2.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Executing, Gate Intersection, Outlier Intersection, Position Intersection X, Position Intersection Y, Count Gate Intersections, Count Outlier Intersections, Last Intersection, Classification。

Data Table Vertical

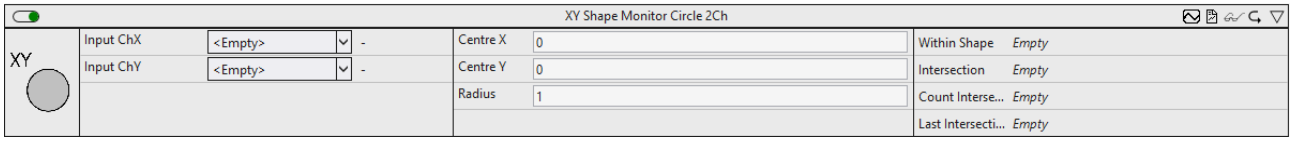
	Data Table
Input 1	20.42
Input 2	25.43
Executing	TRUE
Last Event	10 Mar 2021 16:15:29

DataTable Horizontal

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

また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをXY Gate Monitor 2Chアルゴリズムにマッピングすることもできます。

6.5.13.2 XY Shape Monitor Circle 2Ch



*XY Shape Monitor Circle 2Ch*は、XY 入力と指定した円形状との交点の数をカウントします。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Centre X:**円の中心のX座標。
- **Centre Y:**円の中心のY座標。
- **Radius:**円の半径。

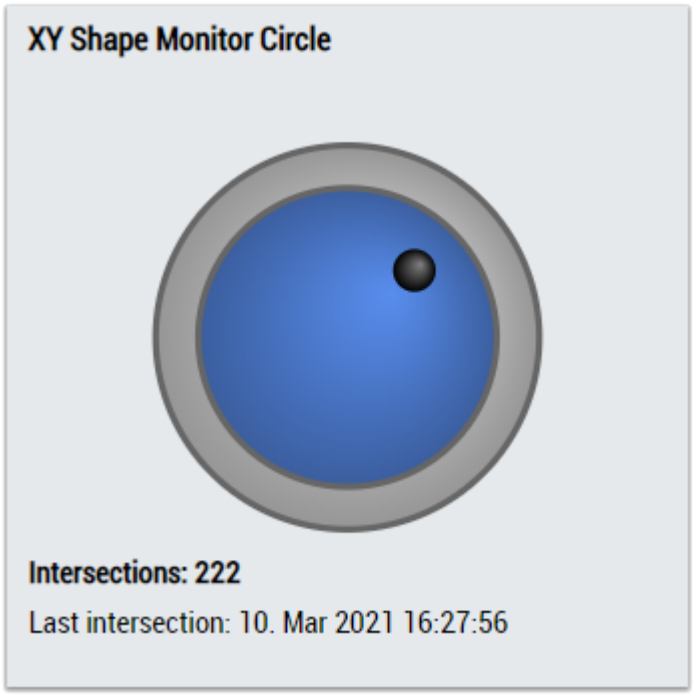
出力値

- **Within Shape :** 入力信号が現在指定された形状内にあるかを示します。
- **Intersection:**入力信号が現在指定された形状と交差するかを示します。
- **Count Intersections:**入力信号と形状の交点の総数をカウントします。
- **Time Intersection:**最後に交差するイベントが発生した時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

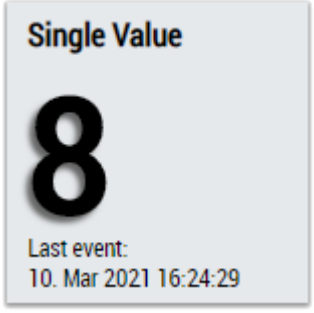
HMIコントロール

XY Shape Monitor Circle 2Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

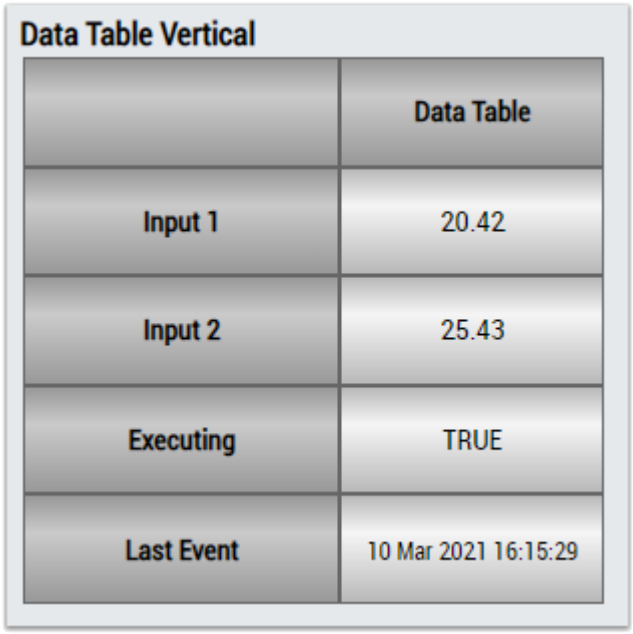
1.XYShapeMonitor コントロールは、出力値Shape、Intersection、Count Intersections、Last Intersection を表示します。

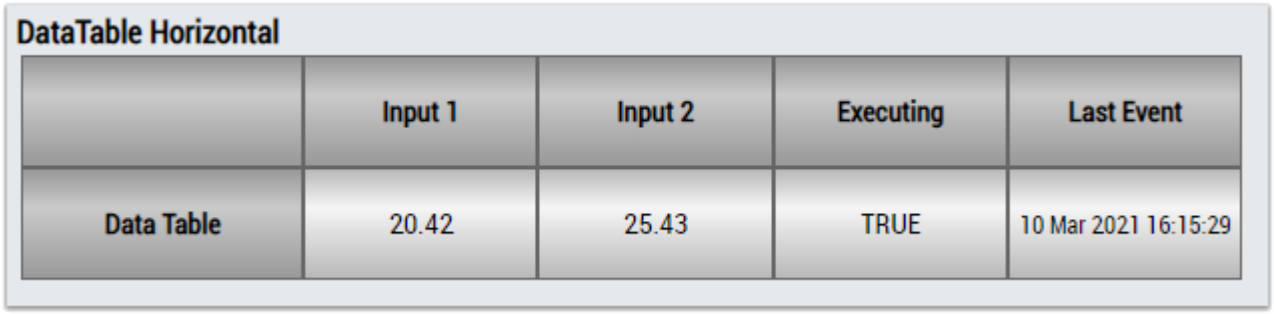


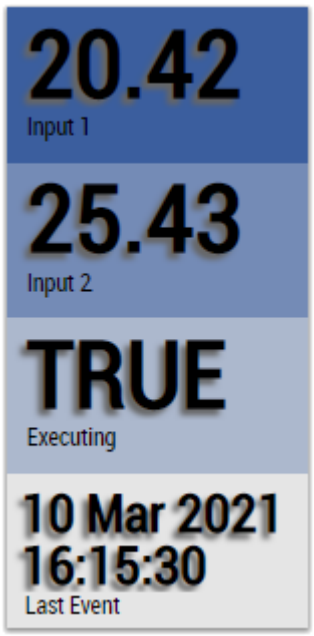
2.SingleValue コントロールは、出力値 Intersection と Last Intersection を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Within Shape, Intersection, Count Intersections, Last Intersection。

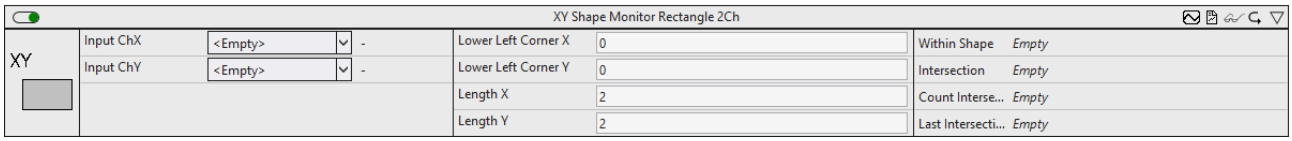






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをXY Shape Monitor Circle 2Chアルゴリズムにマッピングすることもできます。

6.5.13.3 XY Shape Monitor Rectangle 2Ch



*XY Shape Monitor Rectangle 2Ch*は、XY入力が指定された矩形形状と交差した回数をカウントします。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Lower Left Corner X:**矩形左下のX座標。
- **Lower Left Corner Y:**矩形の左下のY座標。
- **Length X:** X方向（正方向）の矩形の長さ。
- **Length Y:**Y方向（正方向）の矩形の長さ。

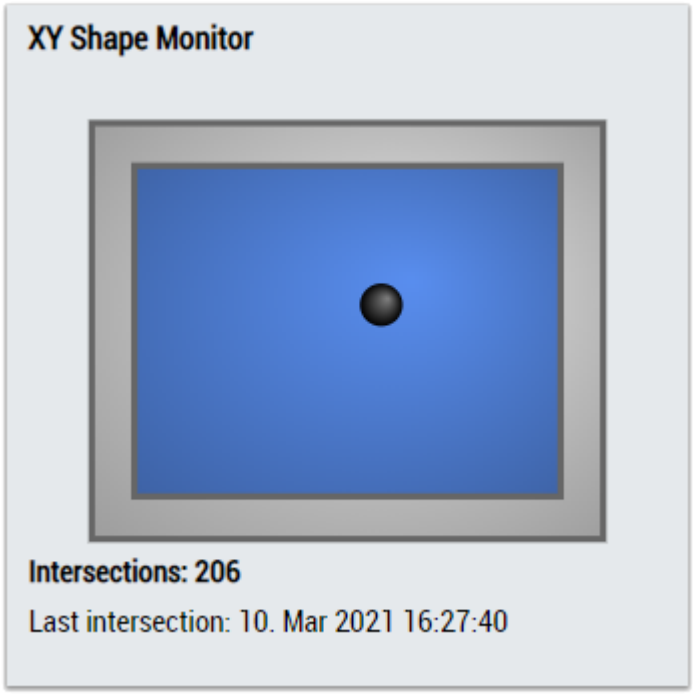
出力値

- **Within Shape :** 入力信号が現在指定された形状内にあるかを示します。
- **Intersection:**入力信号が現在指定された形状と交差するかを示します。
- **Count Intersections:**入力信号と形状の交点の総数をカウントします。
- **Time Intersection:**最後に交差するイベントが発生した時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

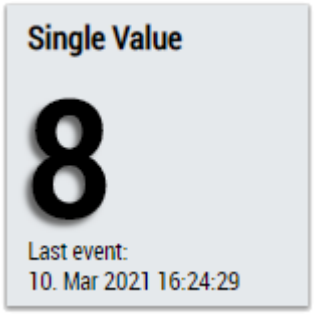
HMIコントロール

XY Shape Monitor Rectangle 2Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

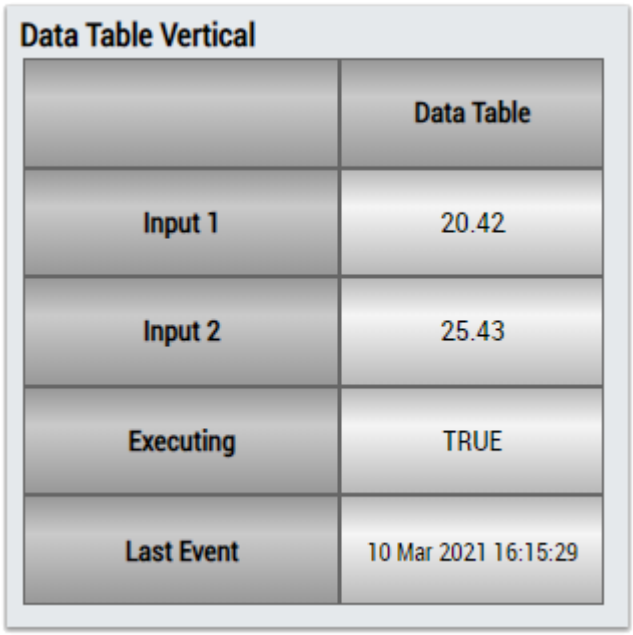
1.XYShapeMonitor コントロールは、出力値Shape、Intersection、Count Intersections、Last Intersection を表示します。

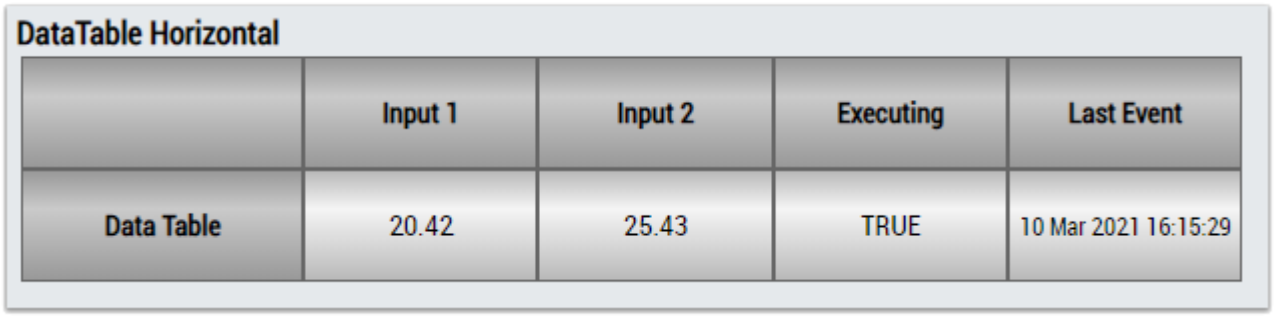


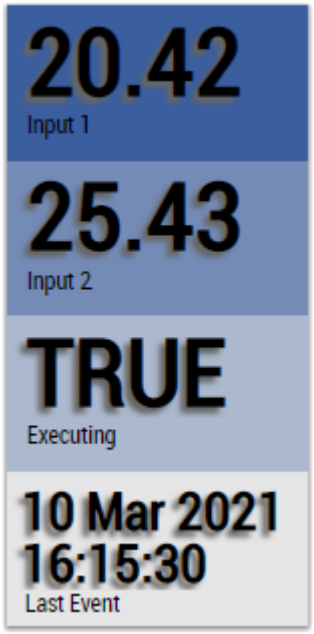
2.SingleValue コントロールは、出力値 Intersection と Last Intersection を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Within Shape, Intersection, Count Intersections, Last Intersection。

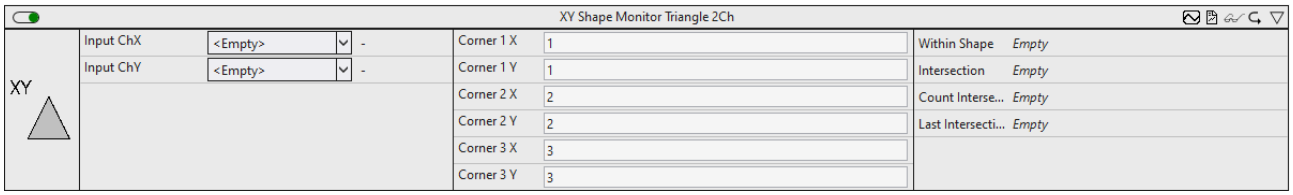






また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをXY Shape Monitor Rectangle 2Chのアルゴリズムにマッピングすることもできます。

6.5.13.4 XY Shape Monitor Triangle 2Ch



*XY Shape Monitor Triangle 2Ch*は、XY 入力と指定された三角形状とが交差した回数をカウントします。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Corner 1 X:**三角形の 1 つ目の頂点のX座標。
- **Corner 1 Y:**三角形の 1 つ目の頂点のY座標。
- **Corner 2 X:**三角形の2つ目の頂点の
- **Corner 2 Y:**三角形の2つ目の頂点のY座標。
- **Corner 3 X:**三角形の3つ目の頂点X座標。
- **Corner 3 Y:**三角形の3つ目の頂点のY座標。

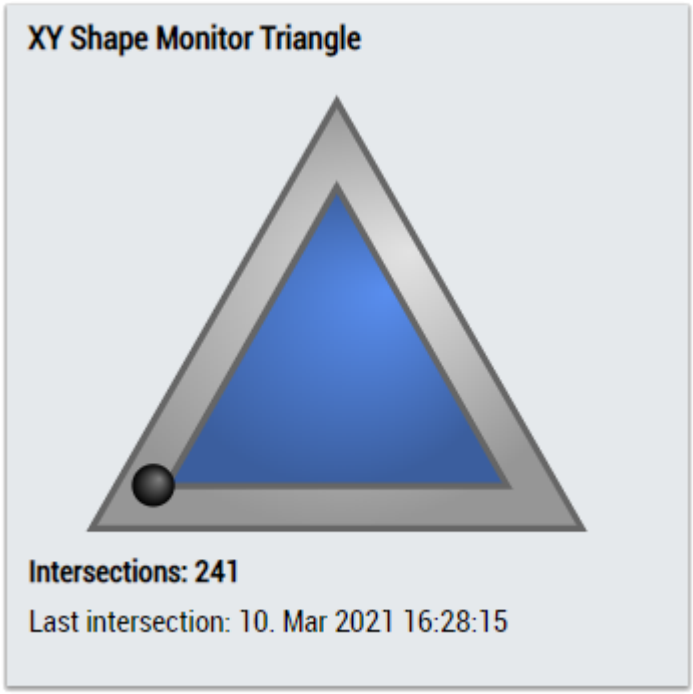
出力値

- **Within Shape :** 入力信号が現在指定された形状内にあるかを示します。
- **Intersection:**入力信号が現在指定された形状と交差するかを示します。
- **Count Intersections:**入力信号と形状の交点の総数をカウントします。
- **Time Intersection:**最後に交差するイベントが発生した時間を示します → イベントをスコープチャートにドラッグ＆ドロップすることで、トリガーイベントとして表示できます。

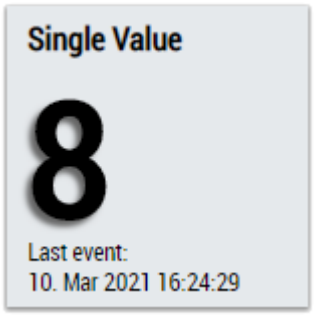
HMIコントロール

XY Shape Monitor Triangle 2Ch アルゴリズムでは、Analyticsダッシュボード作成のために、以下のHMIコントロールを使用できます：

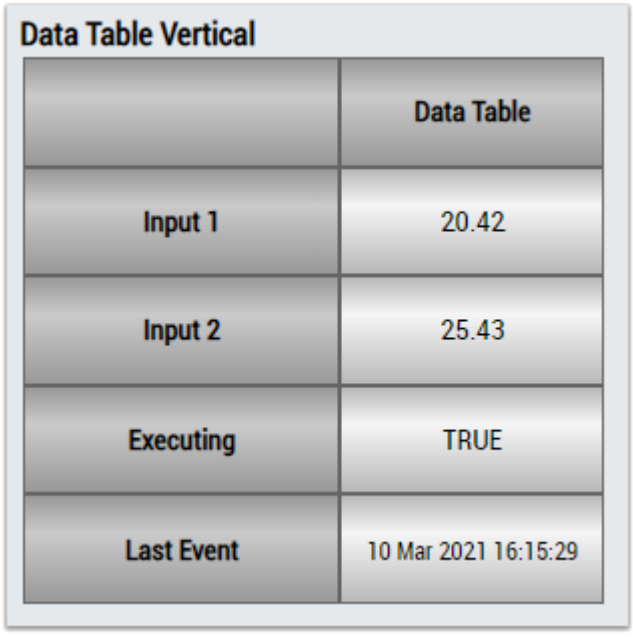
1.XYShapeMonitor コントロールは、出力値Shape、Intersection、Count Intersections、Last Intersection を表示します。



2.SingleValue コントロールは、出力値 Intersection と Last Intersection を表示します。



3.TableコントロールまたはMultivalueコントロールは、以下、全ての出力値を表示します：Within Shape, Intersection, Count Intersections, Last Intersection。



DataTable Horizontal

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

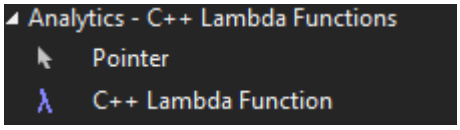
また、Mappingウィザード [▶ 438]を使用して、顧客固有のHMIコントロールをXY Shape Monitor Triangle 2Chアルゴリズムにマッピングすることもできます。

6.5.14 Analytics - extension of the algorithms（解析 - アルゴリズムの拡張）

TwinCAT Analyticsワークベンチのコンフィグレータまたは TwinCAT Analyticsサービスツールで利用できるライブラリのアルゴリズムを拡張するために、2つの拡張オプションが用意されています。 1 つ目に、ラムダ関数を使用して、ユーザ固有のアルゴリズムを開発し、Analyticsワークフローに統合できます。2つ目に、他のライブラリのアルゴリズムを使って既存のアルゴリズムを拡張することもできます。

6.5.14.1 C++ラムダ関数

C++ラムダ関数でユーザ独自のアルゴリズムを開発し、TwinCAT Analyticsワークフローに統合することができます。アルゴリズムの開発ガイドとして、Visual Studio® Toolboxにラムダテンプレートが用意されています。



ラムダテンプレートは、ラムダ関数の設定、プログラミング、パブリッシュに使用できます。ラムダ関数の開発が正しく完了すると、そのアルゴリズムはツールボックスの「Analytics - C++ Lambda Functions」グループに配置されます。さらに、Analyticsワークフローにおける後続の全てのステップが利用可能になります。例えば、ラムダ関数をHMIコントロールにリンクさせることができます。デプロイ後、ラムダ関数はTwinCAT Analyticsランタイムで利用可能です。正しく設定すると、HMIワンクリックダッシュボードに表示されます。

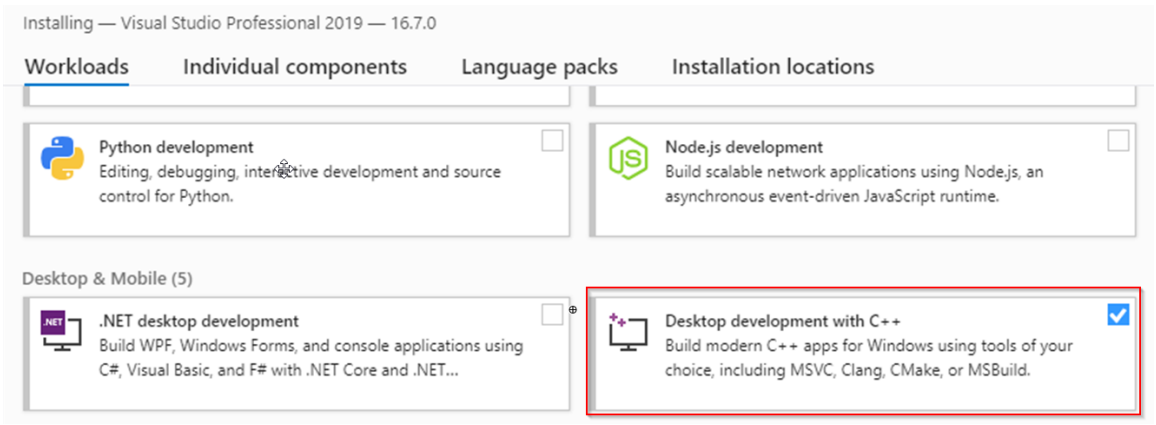
6.5.14.1.1 要件

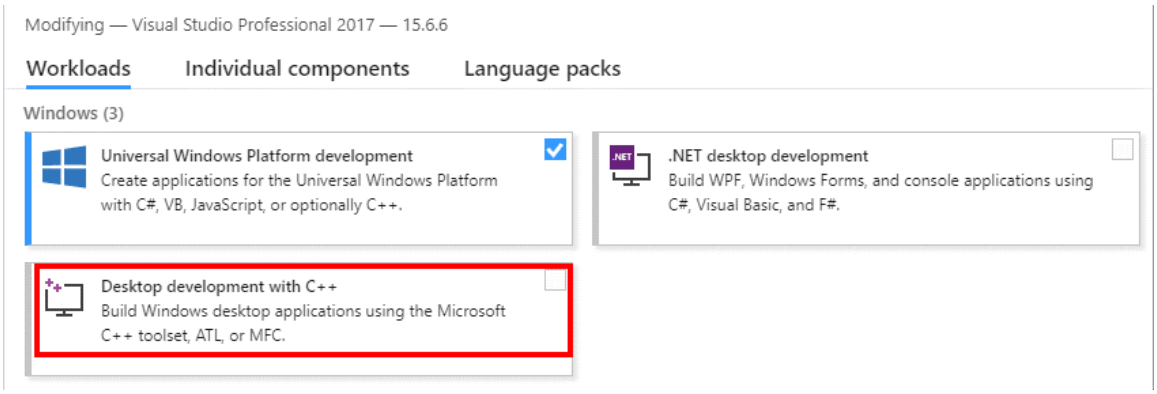
最低要件

TwinCAT 3 C++モジュールの実装とデバッグには、以下の最小要件を満たす必要があります。

開発PCには以下をインストールする必要があります：

- Microsoft Visual Studio® 2017または2019 Professional / Enterprise。
  - Visual Studio® をインストールする際、**Desktop development with C++（C++ によるデスクトップ開発）**オプションを追加で選択する必要があります：





- TwinCAT 3のインストール（XAEエンジニアリング）
- 既存のバイナリC++モジュールをTwinCAT 3 PLC環境に統合して使用する場合、XAE Shell版で対応可能です（Visual Studio®は必要ありません）。

実行用PC（ランタイムPC）

- OS Windows 7 / 10またはTwinCAT/BSDのいずれかを搭載した産業用PCまたは組込み型PC（CX）。32ビット版と64ビット版の両方に対応しています。
- Microsoft Visual Studio®のインストールは**不要**です。
- TwinCAT 3.1のインストール（XARランタイム）

6.5.14.1.2 準備 - 1回のみ

ラムダ関数とTwinCAT C++モジュールを開発するためのPCを準備する必要があります。これらの手順を実行するのは、システムごとに1回のみです：

- TwinCAT Basisとコンフィギュレーションツールバー、プラットフォームツールバーを設定します。
- モジュールに署名するためのテスト証明書を作成します。
- TcSignToolを使用して、証明書のパスワードをプロジェクト外に保存します。
- 証明書名で環境変数を作成します。

C++ラムダ関数をパブリッシュする際に、対応する署名要件の概要がログファイルに出力されます。ログファイルから 「Checking Signing Preconditions」を検索し、関連するエラー処理を行います。ドライバの署名に成功すると、次のように表示されます：

```
== 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.
```

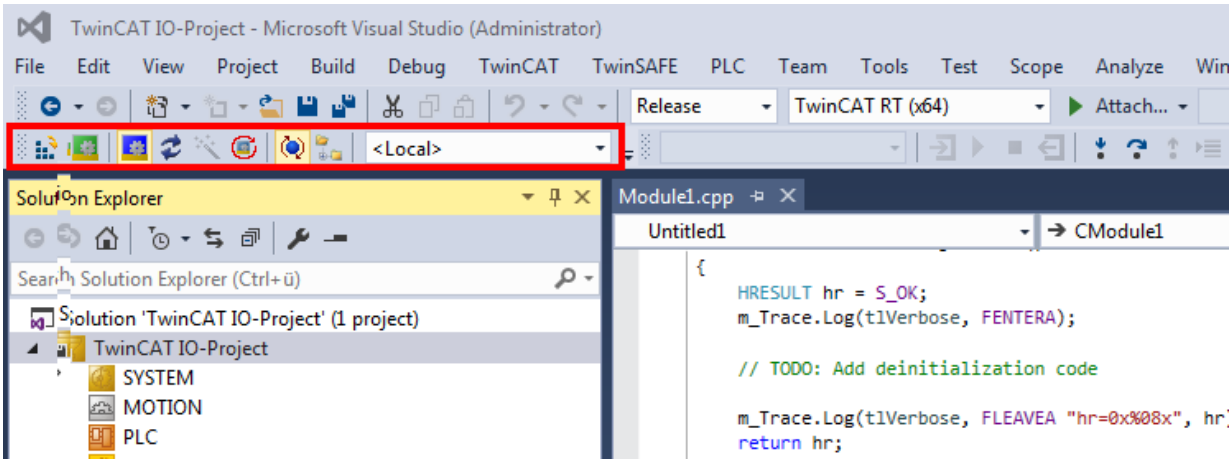
6.5.14.1.2.1 Visual Studio - TwinCAT XAE Baseツールバー

TwinCAT XAE Baseツールバーによる効率的な開発

TwinCAT 3は、Visual Studioのメニューに独自のツールバーを統合し、効率的な開発をサポートしています。ツールバーは、C++プロジェクトの作成を支援します。ツールバーは、TwinCAT 3のセットアップ時に自動的にVisual Studioメニューに追加されます。手動で追加したい場合は、以下の手順で追加できます：

1. **View** メニューを開き、**Toolbars¥TwinCAT XAE Base**を選択します。

⇨ 選択したツールバーがメニューの下に表示されます。



6.5.14.1.2.2 ドライバ署名

TwinCAT C++モジュールを実行するには、証明書で署名する必要があります。

この署名は、出所を追跡できるC++ソフトウェアのみが本番環境で実行されることを保証します。

テスト目的であれば、信頼性が保証されない証明書も署名に使用できます。ただし、これはOSがテストモードの場合にのみ可能で、本番環境でこれらの証明書が使用されることはありません。

エンジニアリング環境では署名は不要です

証明書が必要なのは実行環境だけで、エンジニアリング環境では必要ありません。

モジュールをロードするには2つの方法があります。このため、署名には異なる証明書が使用されます：

- TwinCAT: C++モジュールはTwinCATランタイムシステムでロードし、TwinCATユーザ証明書で署名する必要があります。
  - TwinCAT 3.1. 4024 以上では、この手順も利用できます。
  - この手続きは、 のような新しい機能やを実行するために必要です。
  - TwinCAT/BSDに必要です。

- (4024.0未満のバージョンでは、新規プロジェクトには推奨されなくなりました。既存プロジェクトは移行する必要があります。)
  オペレーティングシステム：C++モジュールは通常のカーネルドライバとしてロードされるため、署名も必要です。
  - TwinCAT 3.1. 4022 以前では、この手順のみ可能です。
  - Windows 7 (Embedded) x86 (32bit)は署名不要です。
  - TwinCAT/BSDでは使用できません。

パブリッシュされたモジュールは、様々なPCで実行可能にするため、必ず署名が必要です。

開発ソフトウェアと本番ソフトウェアの組織的分離

ベッコフは、（少なくとも）2つの証明書を組織的に区別して運用することを推奨しています。

1. 開発プロセスでテストモードが必要な場合に、カウンターサインされていない証明書を使用します。この証明書は、各デベロッパーが個別に発行することもできます。
2. 最終テストに合格したソフトウェアのみが、カウンターサイン付きの証明書によって署名されます。このソフトウェアは、機械にインストールして納品できます。

このように開発と運用を分離することで、テスト済みのソフトウェアだけが生産性の高い本番システム上で動作可能となります。

6.5.14.1.2.2. TwinCAT 1

バージョン管理されたC++プロジェクトは、バイナリ形式でTMXファイル（TwinCAT Module Executeable）に格納されます。

TwinCAT 3 C++モジュールを実装する場合に、TwinCATランタイムでロードするには、コンパイルされた実行可能なTMXファイルをTwinCATユーザ証明書で署名する必要があります。

署名

TwinCAT TMXファイルは、署名に成功した後にのみロードできます。

注記

32ビットおよび64ビットシステムでの署名

オペレーティングシステムの署名とは対照的に、TwinCATの署名は32ビットと64ビットの両方のシステムを対象としています。そのため、32ビットシステムでテスト署名を行う場合も、テストモードが必要となります。

TMXファイルに署名するには、TwinCATユーザ証明書が必要です [▶ 275]。これは、署名用プロジェクトで設定します。

テスト署名

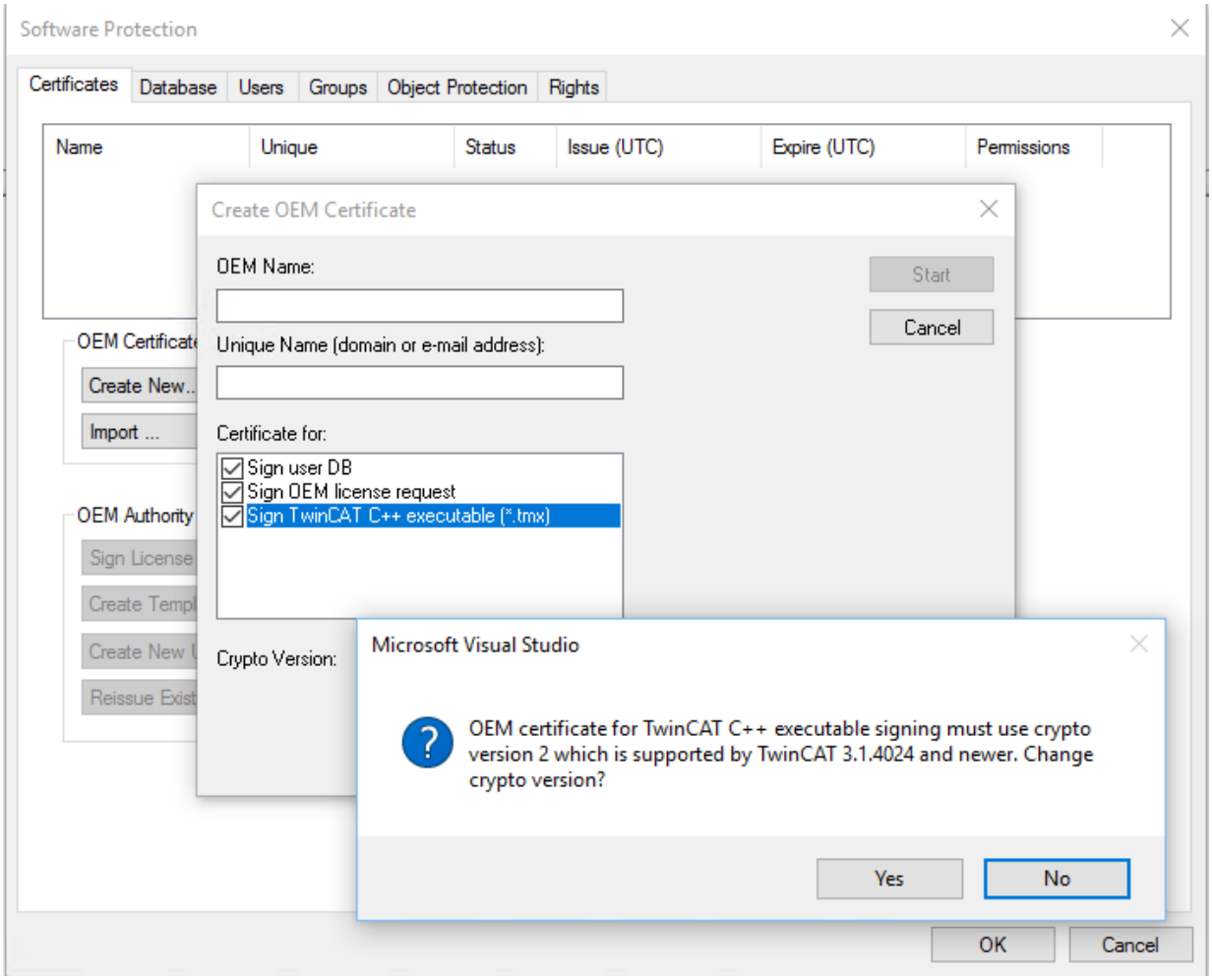
ユーザ証明書は、TwinCATのローカル環境に作成できます。ベッコフによるカウンターサインがない限り、テストモードを有効にする必要があります（こちら [▶ 275]を参照）。

ベッコフがTwinCATユーザ証明書に署名した [▶ 278]時点で、テストモードを終了できます。有効化するのと同じ方法で無効化できます。

6.5.14.1.2.2.1 テスト署名.1

TwinCATのテスト署名は、実際の納品と同じTwinCATユーザ証明書を使用して実施できます（TwinCAT 3 ユーザ証明書の申請 [▶ 278] を参照）。

1. ソフトウェア開発中など、テスト運用を行う場合は、TC0008 Certificate Request Fileの作成 [▶ 279] に記載されているように、TwinCATユーザ証明書を作成するだけで十分です。必ず 「Sign TwinCAT C++ executable (*.tmx) 」を選択してください。この場合、「Crypto version 2 is required」のメッセージが表示されます。

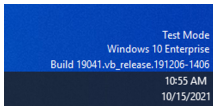


XAR（ローカルテストの場合はXAE）で、オペレーティングシステムが自己署名証明書を承認するようにテストモードを有効にします。これは、エンジニアリング（XAE）とランタイム（XAR）の両方で可能です。

Windowsの場合

管理者プロンプトを使って以下を実行します。
bcdedit /set testsigning yes
ターゲットシステムを再起動します。
このためにBIOSで「SecureBoot」をオフにする必要がある場合があります。

テスト署名モードが有効になると、デスクトップの右下に表示されます。システムは、全ての署名済みドライバの実行を承認するようになりました。



TwinCAT/BSDの場合

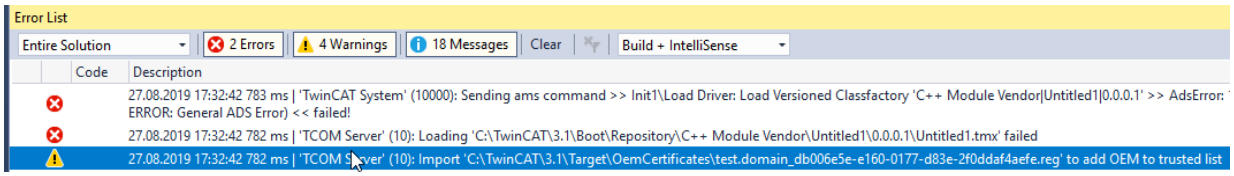
/usr/local/etc/TwinCAT/3.1/TcRegistry.xmlのファイルで、キー 「System 」の下に„<Value Name=″EnableTestSigning″ Type=″DW″>1</Value> を入力します。

```
<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>
```

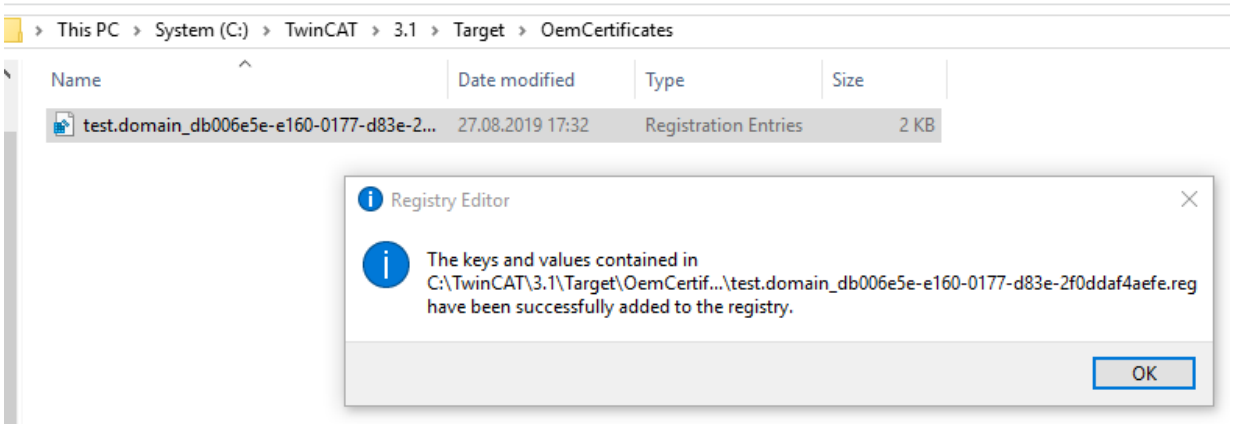
次に、TwinCAT System Serviceを再起動します。
doas service TcSystemService restart

各手順の後、システムは全ての署名済みドライバの実行を承認します。

1. TwinCATユーザ証明書を使用した最初のアクティベーション（Activate Configuration）中に、ターゲットシステムで信頼されていない証明書が検出され、アクティベーションプロセスが中断されます:



Windowsの場合：
管理者権限を持つローカルユーザが、作成したREGファイルを実行するだけで、証明書を信頼できます：



TwinCAT/BSD の場合：
「Tcimportcert 」パッケージがインストールされていない場合は、インストールしてください：pkg install tcimportcert
doas tcimportcert /usr/local/etc/TwinCAT/3.1/Target/OemCertificates/<CreatedFile>.reg を介して証明書を信頼します。
その後、TwinCAT System Serviceを再起動するか、システムを再起動してください：
doas service TcSystemService restart

⇨ このプロセスでは、信頼された TwinCAT ユーザ証明書で署名された C++ モジュールのみが実行可能です。

2. このプロセスの後、オペレーティングシステムのテストモードで署名するため、TwinCATユーザ証明書を使用できます。
これはプロジェクトのプロパティで設定します。
TwinCAT ユーザ証明書のパスワードが、プロジェクト内に保存されてバージョン管理システムに含まれるのを防ぐために、TcSignTool [▶ 289] を使用します。

TestModeを使用せずにTwinCATユーザ証明書を納品する場合は、ベッコフがカウンターサインした証明書 [▶ 278]が必要です。

6.5.14.1.2.2.1 テストモードなしで納品するための署名済みTwinCATユーザ証明書.2

システム要件

- TwinCAT 3.1 ビルド4024以上
- Windows 10以上またはTwinCAT/BSD（ターゲットシステム上）

ベッコフでは、TwinCAT ビルド4024で「TwinCAT 3 OEMユーザ証明書」を発行し、TwinCAT 3で作成したTMXファイルにC++で署名できるようにしました。

- この証明書は、Windows環境で使用されるため、申請者データの安全な検証を必要とします。したがって、TwinCAT 3ユーザ証明書は、住所および連絡先データの検証のために正式に注文する必要があり、ベッコフの既存顧客にのみ発行されます。
- 注文番号： **TC0008**
- このTwinCAT 3ユーザ証明書の発行は無料です。
- 証明書を保存するディレクトリ：**C:¥TwinCAT¥3.1¥CustomConfig¥Certificates**

TwinCAT 3 TMXファイルの使用にはTwinCAT 3ユーザ証明書は必要ありません

TwinCAT 3ユーザ証明書は、TMXファイルの1回限りの署名にのみ使用されます。この証明書で署名されたTMXファイルの使用に、証明書は必要ありません。

TwinCAT 3ユーザ証明書 TC0008が必要となるPC

TwinCAT 3ユーザ証明書は、TMXファイルが署名された開発用コンピュータのみに格納する必要があります。つまり、各ターゲットシステムには**不要**です。

TwinCAT 3ユーザ証明書の有効期限

セキュリティ上の理由から、TwinCAT 3ユーザ証明書の有効期限は2年間に制限されています。

証明書の有効期限が切れた場合

新しいTMXファイルに署名できなくなります。
ただし、既に署名されたTMXファイルは制限なく使用すできます。

2年の有効期限が切れる前であれば証明書の更新を申請できます。（切れた後でも申請できます）。

TwinCAT 3ユーザ証明書を更新するには、新しい証明書を要求する場合と同じプロセスが適用されます。更新の場合も、証明書を注文する必要があります（証明書更新の注文番号は、新規証明書の番号と同じです）。

新規証明書の場合と異なり、新たにOEM Certificate Request Fileは作成せず、既存の証明書をベッコフの証明書部門に送信して更新します。これは証明書の更新であり、新規発行ではない旨をメールでお知らせください。それ以外は、新規証明書の申請と同じEメールの内容が適用されます。

既存の証明書に新しい有効期限が付与され、再署名されて、さらに2年間有効となります。

したがって、新たに署名された証明書は、元のバージョンと完全に互換性があります。

6.5.14.1.2.2.1. TwinCAT 3ユーザ証明書の申請 2.1

注文と認証プロセスの概要

TwinCATユーザ証明書を請求するには、正式な注文が必要です。

- 注文番号**TC0008**（TwinCAT 3 Certificate Extended Validation）
- TwinCAT 3ユーザ証明書の発行（および更新）は無料です。

- TwinCAT 3 ユーザ証明書はデジタル ID カードに相当するため、一般的な市場基準に基づき、申請者の連絡先情報の確認が求められます。
- そのため、TwinCAT 3ユーザ証明書は、ベッコフの既存顧客に対してのみ発行されます。

注文と認証プロセスの概要

メールアドレスは企業のメールアカウントである必要があり（Gmail等のフリーメールは不可）、申請者の企業名と一致する必要があります。

1. ベッコフの担当営業にご連絡のうえ、TwinCAT 3 OEM証明書を申請してください。「TC0007」または「TC0008」をご注文ください。
2. 重要：お問い合わせの際は、送付先（連絡先氏名、Eメールアドレス）、証明書の使用場所（会社名、住所）をお知らせください。
3. 注文書に記載された連絡先を認証し、お客様（送付先に記載された担当者）宛にベッコフの担当営業からご連絡します。
4. 新しいOEM証明書を申請する場合：TC0008 Certificate Request Fileの作成 [▶ 279] 。
5. TwinCAT Engineeringを使用して、OEM証明書ファイルの 「File Fingerprint」を決定します（OEM証明書ファイルのFile Fingerprint 設定 [▶ 282] を参照）。連絡先データ認証のため、このFile Fingerprintをベッコフの担当営業にお知らせください。File Fingerprintの送信は、OEM Certificate Request Fileの送信に使用する通信チャンネルとは別の通信チャンネルを使用する必要があります。
6. ベッコフの担当営業に「OEM Certificate File」を送付してください。
7. ベッコフ本社にて証明書ファイルに署名した後、担当営業よりEメールで送付します。

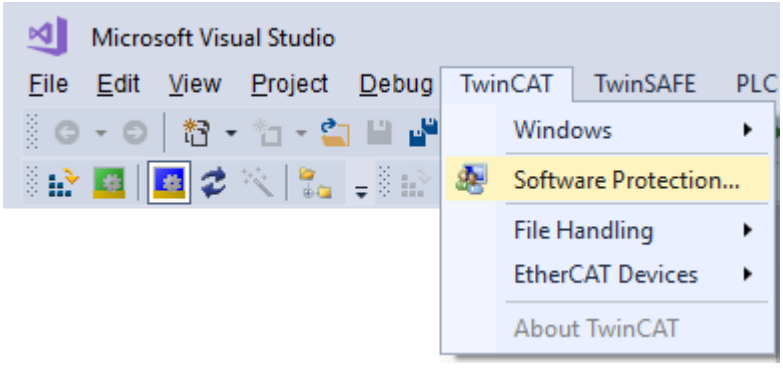
連絡先の認証と証明書の発行には数日かかる場合があります。

6.5.14.1.2.2.1. TC0008 Certificate Request Fileの作成 2.2

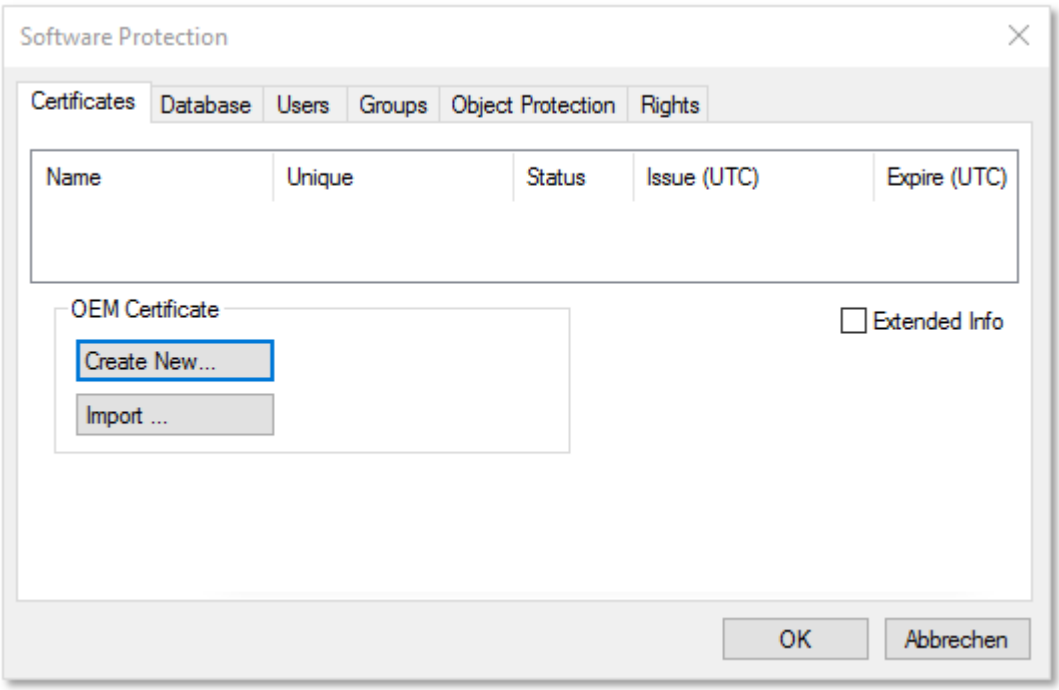
システム要件

- TwinCAT 3.1 ビルド4024以上
- Windows 10以上またはTwinCAT/BSD（ターゲットシステム上）

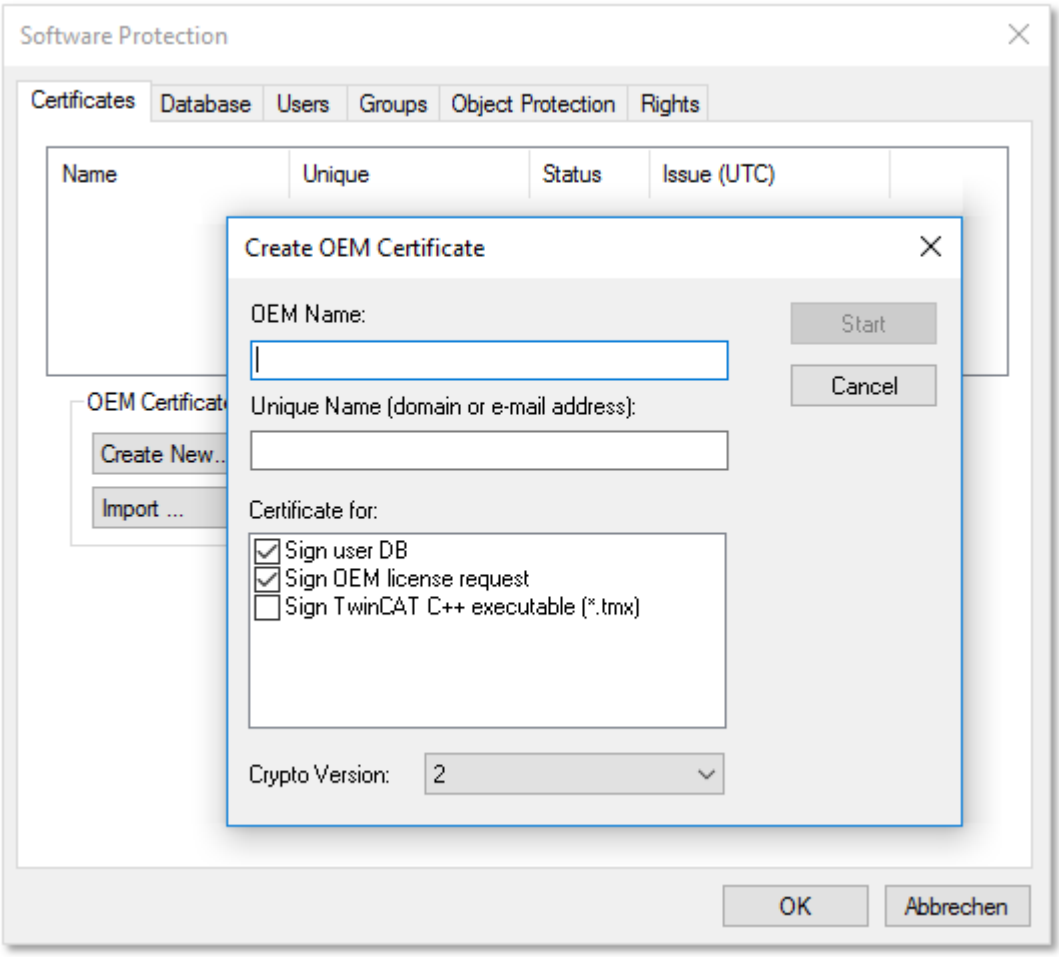
1. Software Protectionコンフィグレータを呼び出します。呼び出しには、メインメニュー**TwinCAT**の下にある**Software Protection**を選択します：



2. 開いたウィンドウの**Certificates**タブを選択します。
Create New...をクリックします：

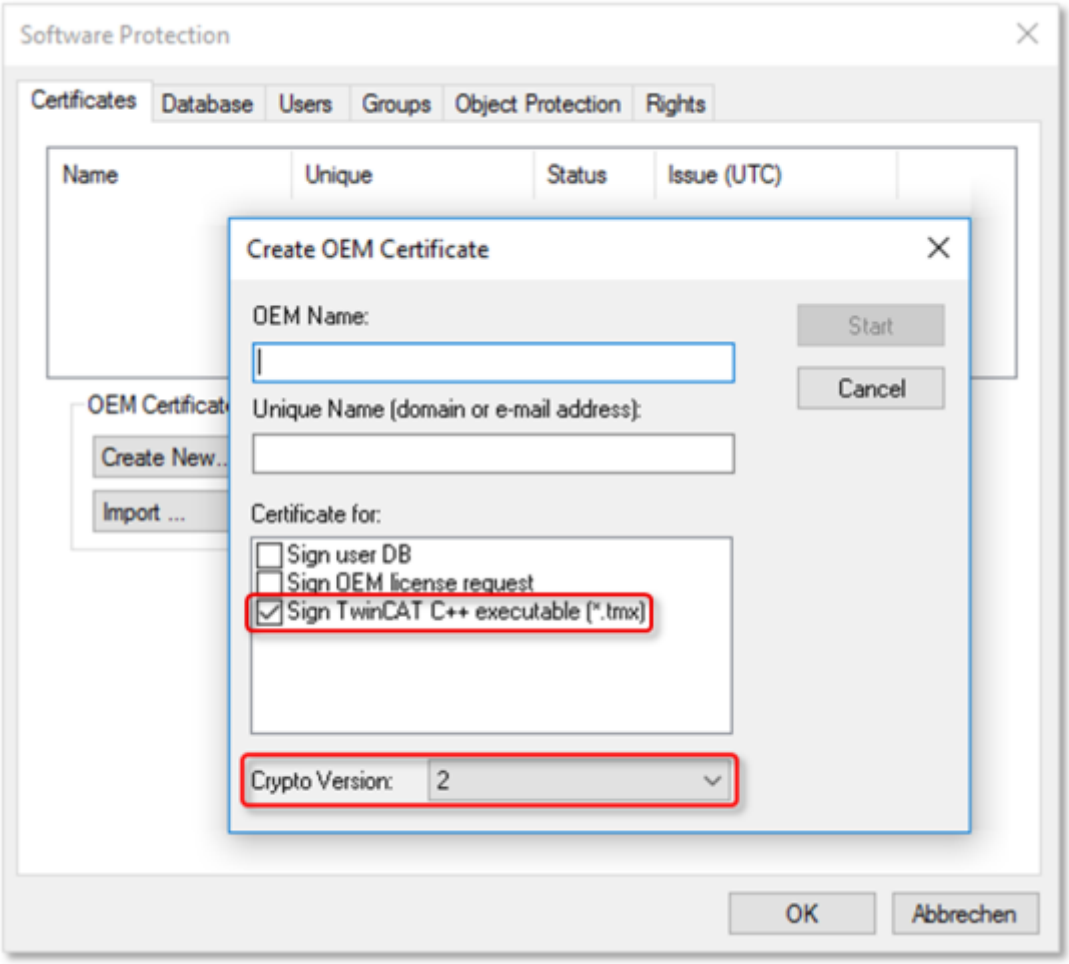


3. **Create OEM Certificate**入力ウィンドウが開きます：



4. 必要なデータを入力します：

- **OEM Name**テキストフィールドに会社名を入力してください。申請者の会社または事業部門の正確な名称を入力してください。
- **Unique Name**を入力してください。「OEM Unique Name」は、全世界で証明書の所有者を一意に識別できる固有の名称にしてください。望ましい名称として、会社のウェブサイトURLまたはEメールアドレスがあります。Eメールアドレスは会社のものである必要があります。つまり、そのアドレスから申請者の所属企業を明確に特定できる必要があります。
- チェックボックス **Sign TwinCAT C++ executables**にチェックを入れます:



この証明書でTwinCATドライバソフトウェアのみに署名する場合は、他の2つのチェックボックスのチェックを外します。(これらはPLCエリアでのみ使用されます。)

- Crypto version 2（証明書の暗号化コンテンツ）が設定されていることを確認します。（標準設定）

5. データを入力したら、**Start**をクリックし、ファイルを保存するディレクトリを選択します。
提案されたディレクトリ「c:¥twincat¥3.1¥customconfig¥certificates」をそのまま使用してもかまいません。次のステップで、ファイルの「ファイルフィンガープリント」を読み出す [▶ 282]**ために、このディレクトリに新しく作成されたファイルが必要です。**
   ⇨ OEM Private Keyのパスワードを選択するダイアログが開きます。
6. OEM Private Keyのパスワードを発行します **。**

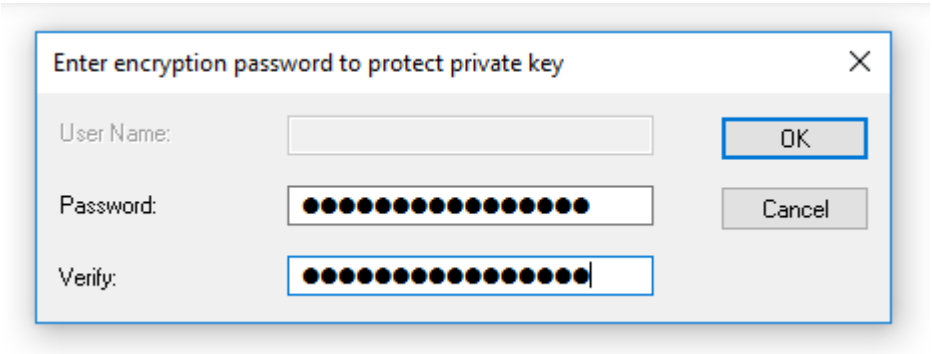
重要：パスワードのセキュリティ

証明書には強力なパスワードを必ず使用してください。
パスワードが不正な第三者に渡らないよう、適切な対策を講じて保護してください。

パスワードを紛失しても復旧できません

ベッコフではパスワードの復旧やリセットはできません。証明書のパスワードを忘れたり紛失したりした場合、証明書は使用できなくなり、新しい証明書を申請する必要があります。

7. パスワードを再入力して確認し、**OK**でダイアログを閉じてください。



⇨ ファイルが保存されます。

このようにして生成された「Certificate Request File」は、ベッコフの証明書部門が署名しなければ有効になりません。手順は証明書の申請 [▶ 278]の章に記載されています。

6.5.14.1.2.2.1. OEM証明書ファイルのFile Fingerprint 設定 2.3

TwinCAT OEM Certificate Extended Validation（TC0008）を要求するために、この機能が必要です。

システム要件
この機能を使用するには、TwinCAT 3.1 ビルド 4024以降が必要です。

OEM Certificate Request Fileは、ベッコフが署名すると、TwinCAT OEM証明書になります。この署名以外、ファイルに相違はありません。このため、以下のセクションでは、両方のファイルバージョンを指して「TwinCAT OEM 証明書ファイル」という用語を使用します。

TwinCAT 3エンジニアリングによるOEM証明書ファイルの「File Fingerprint」の読み取り

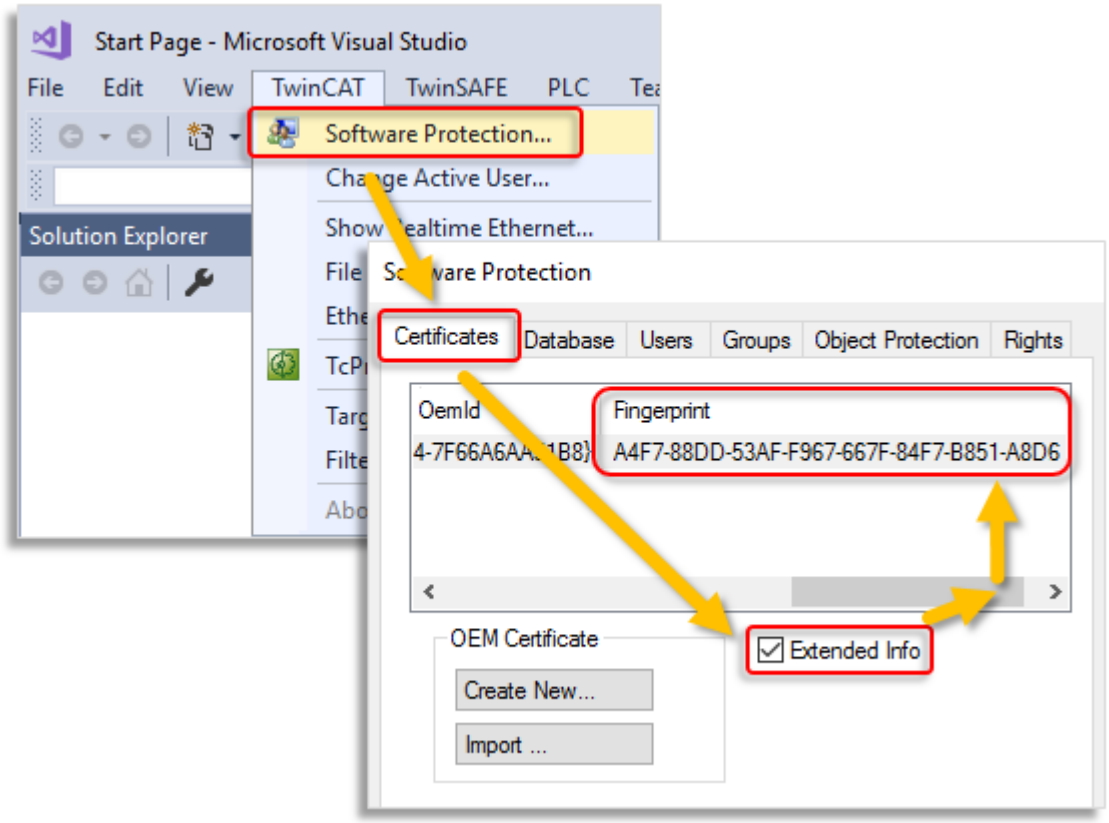
この機能を使用するには、OEM証明書ファイルが以下のディレクトリにある必要があります：「c:¥twincat¥3.1¥customconfig¥certificates」

このディレクトリにはOEM証明書が格納されています（既に証明書を持っていて、更新する場合）。

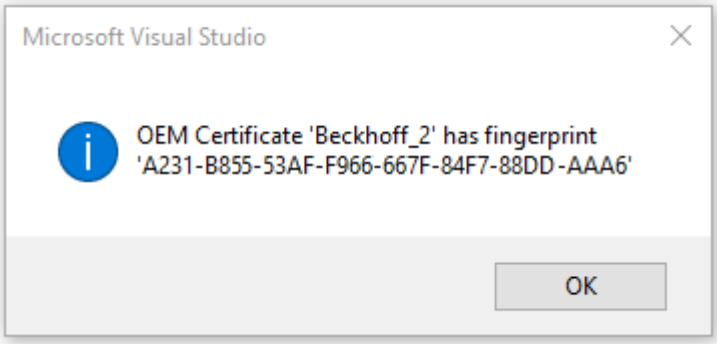
「OEM Certificate Request File」の作成時に推奨ディレクトリを変更しなかった場合、ファイルは既にこのディレクトリにあります。

手順:

1. TwinCAT 3 Software Protectionコンフィグレータを呼び出します。



2. **Certificates**タブを選択します。
3. **Extended Info**チェックボックスにチェックを入れます。
4. ウィンドウを右にスクロールし、**Fingerprint**列を確認します。(別の方法として、証明書の行をダブルクリックすることもできます。） Fingerprint Fileがウィンドウに表示されます：



ショートカット[Ctrl] + [C]を使用すると、メッセージウィンドウから Windows のクリップボードにFingerprintデータをコピーできます。

6.5.14.1.2.2.1. 署名済みTwinCATユーザ証明書の保存 2.4

証明書を保存する推奨ディレクトリ：**C:¥TwinCAT¥3.1¥CustomConfig¥Certificates**

システム要件
- TwinCAT 3.1 ビルド4024以上
- Windows 10以上またはTwinCAT/BSD（ターゲットシステム上）

TwinCAT 3 TMXファイルの使用にはTwinCAT 3ユーザ証明書は必要ありません

TwinCAT 3ユーザ証明書は、TMXファイルの1回限りの署名にのみ使用されます。この証明書で署名されたTMXファイルの使用に、証明書は必要ありません。

TwinCAT 3ユーザ証明書 TC0008が必要となるPC

TwinCAT 3ユーザ証明書は、TMXファイルが署名された開発用コンピュータのみに格納する必要があります。つまり、各ターゲットシステムには**不要**です。

6.5.14.1.2.2. オペレーティングシステム 2

注記

TwinCAT LoaderによるTMXへの移行を推奨

TwinCAT 3.1 4024.0 以降、TwinCAT から直接バイナリをロードできる、バージョン管理された C++ プロジェクトが利用可能です。移行を推奨します。

x64プラットフォームで TwinCAT 3 の C++ モジュールを実装する場合、ドライバ（*.sys ファイル）をオペレーティングシステムがロードするためには、証明書による署名が必要です。

TwinCAT 3ビルドプロセスで自動的に署名され、64ビットWindows OSでドライバの認証に使用されます。

ドライバの署名には証明書が必要です。Microsoft のドキュメントでは、64 ビット Windows オペレーティングシステムで承認されるテストおよびリリース用証明書を取得するための手順と、その背景知識について説明しています。

このような証明書をTwinCAT 3で使用するには、x64ビルドターゲットをコンパイルした後に、「テストモード用のテスト証明書の作成 [▶ 284]」で説明されている手順で設定します。

テスト証明書

テスト目的であれば、自己署名のテスト証明書を作成し、技術的な制限なしに使用できます。

以下のチュートリアルでは、このオプションを有効にする方法を説明します。
実稼働する機械用に実際の証明書を使用したドライバを作成するには、このオプションを無効にする必要があります。

- テストモード用のテスト証明書の作成 [▶ 284]
- （テスト）証明書の削除 [▶ 287]

参考文献

MSDN, MakeCert test certificates (Windows drivers),
https://docs.microsoft.com/en-us/windows-hardware/drivers/install/makecert-test-certificate

6.5.14.1.2.2.2 テスト署名 .1

概要

x64プラットフォーム用のTwinCAT 3 C++モジュールを実装するには、証明書を使用してドライバに署名する必要があります。

この項では、C++ドライバのテスト証明書を作成し、インストールする方法について説明します。

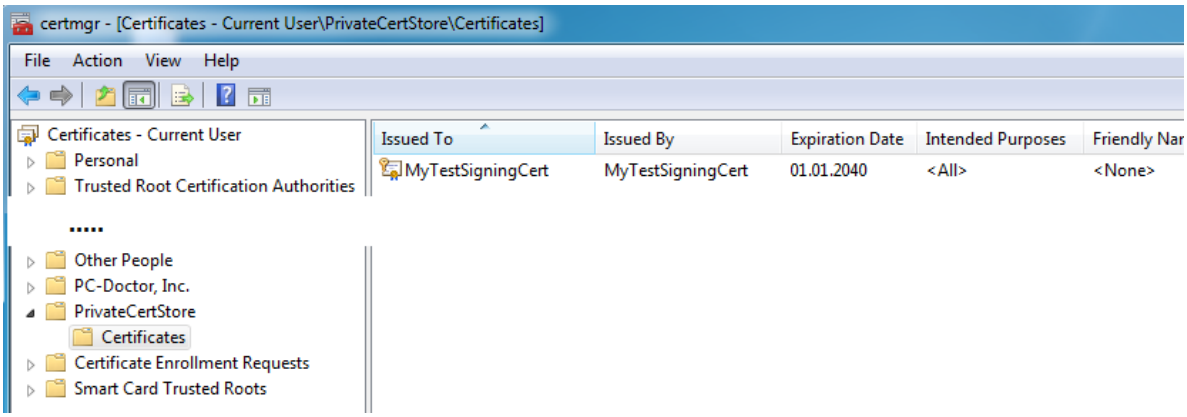
テスト証明書の作成手順

開発者は、証明書を作成するための様々なツールを利用できる場合があります。テスト証明書の仕組みを有効にするには、以下の説明に正確に従ってください。

以下のコマンドは、いずれかの方法で開いたコマンドラインから実行する必要があります：

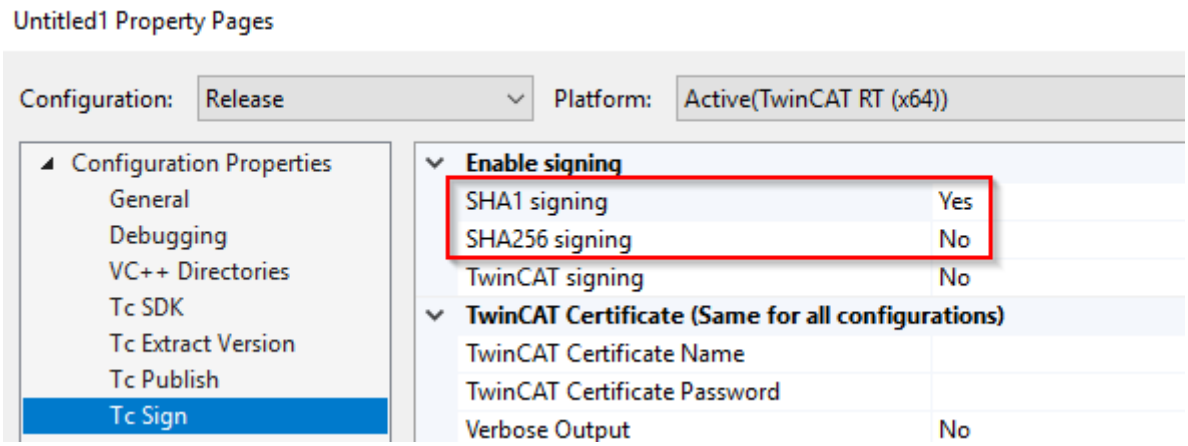
- **管理者権限でVisual Studio 2010 / 2012プロンプトを表示** (手順：**All Programs** -> **Microsoft Visual Studio 2010/2012** -> **Visual Studio Tools** -> Visual Studio Command Prompt、次に、**Run as administrator**を右クリック)
- **管理者権限でVisual Studio 2017 / 2019 の開発者コマンドプロンプトを表示** (手順: **All Programs** -> **Visual Studio 2017** -> Visual Studio Command Prompt for VS 2017/2019、次に、**Run as administrator**を右クリック)
- WINDDKがインストールされている場合のみ：
  管理者権限で通常のプロンプト(**Start**-> Command Prompt)を表示し、対応するツールを含むディレクトリ*%WINDDK7%¥bin¥x86¥*に移動します。

1. XAE:
   エンジニアリングシステムの Visual Studio 2010 / 2012 プロンプトで、管理者権限で次のコマンドを入力します（上記の注釈を参照）：
   makecert –r –pe –ss PrivateCertStore –n CN=MyTestSigningCert MyTestSigningCert.cer
   （PrivateCertStore へのアクセス権がない場合は、別の場所を使用できます。） これは、PostBuild イベントでも使用する必要があります（こちら [▶ 288]を参照。）
   ⇨ 続いて自己署名証明書が作成され、「MyTestSigningCert.cer」ファイルとWindowsCertificate Store に格納されます。
   ⇨ mmcで結果を確認します(**Use File**->**Add/Remove Snap-in**->**Certificates**):



2. XAE:
   証明書がエンジニアリングシステムの TwinCAT XAE で認識されるように設定します。
   エンジニアリングシステムで環境変数 TWINCATTESTCERTIFICATE を "MyTestSigningCert" に設定するか、Debug|TwinCAT RT (x64) および Release|TwinCAT RT (x64) のビルド後のイベントを編集します。
   変数名は証明書ファイル名ではなく、CN 名（この場合は MyTestSigningCert）です。

注記 TwinCAT 3.1 4024.0以降、使用する証明書の設定は、プロジェクトプロパティのTc Signで行います。ここで説明するように、オペレーティングシステム経由で署名を使用するには、プロジェクトの設定に注意してください：

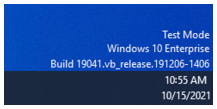


XAR（ローカルテストの場合はXAE）で、オペレーティングシステムが自己署名証明書を承認するようにテストモードを有効にします。これは、エンジニアリング（XAE）とランタイム（XAR）の両方で可能です。

Windowsの場合

管理者プロンプトを使って以下を実行します。
bcdedit /set testsigning yes
ターゲットシステムを再起動します。
このためにBIOSで「SecureBoot」をオフにする必要がある場合があります。

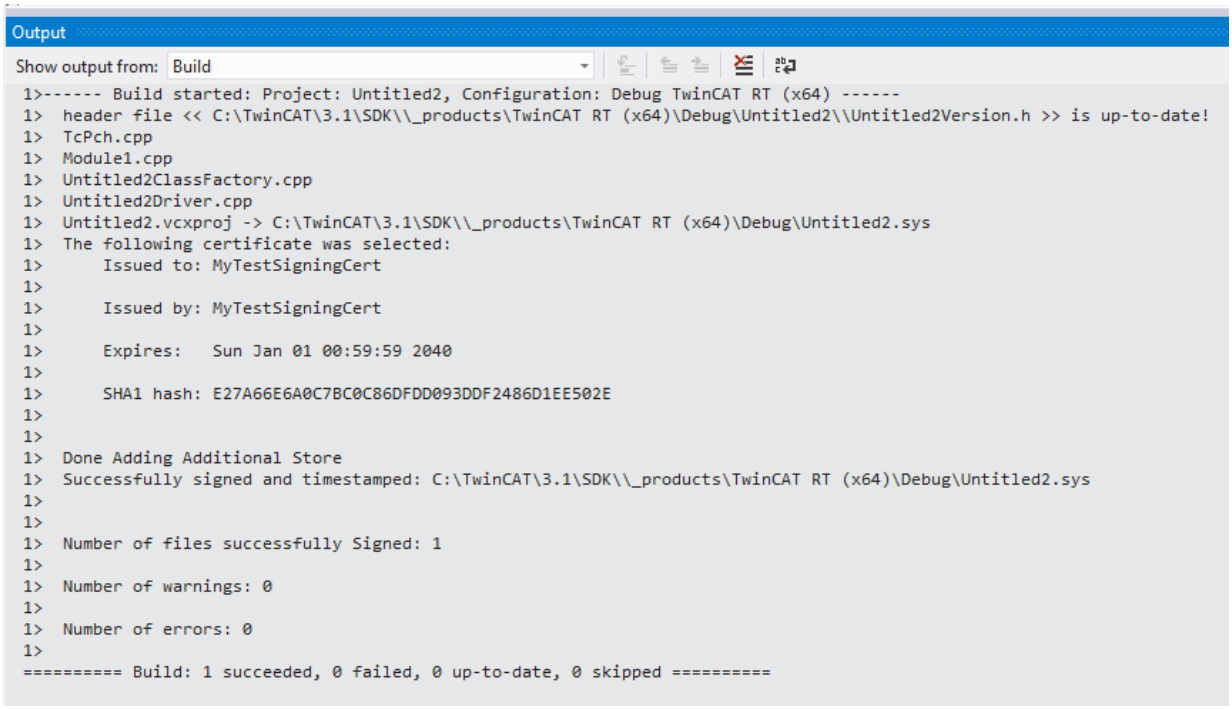
テスト署名モードが有効になると、デスクトップの右下に表示されます。システムは、全ての署名済みドライバの実行を承認するようになりました。



各手順の後、システムは全ての署名済みドライバの実行を承認します。

1. TwinCAT C++ ドライバで実装された TwinCAT モジュールを含む構成が、ターゲットシステム上で有効化および起動できるかどうかをテストします。

⇨ x64ドライバのコンパイルは以下の出力を生成します：



参考文献：
MSDN, MakeCert test certificates (Windows ドライバ)
https://docs.microsoft.com/en-us/windows-hardware/drivers/install/makecert-test-certificate

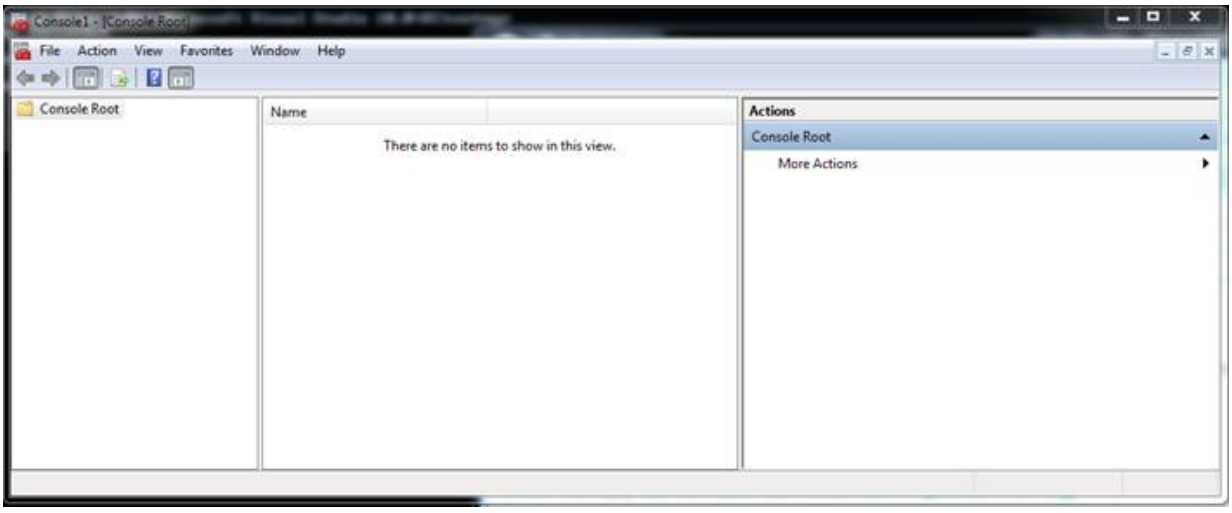
6.5.14.1.2.2.2 テスト証明書の削除 .2

この項では、テスト証明書を削除する方法について説明します。

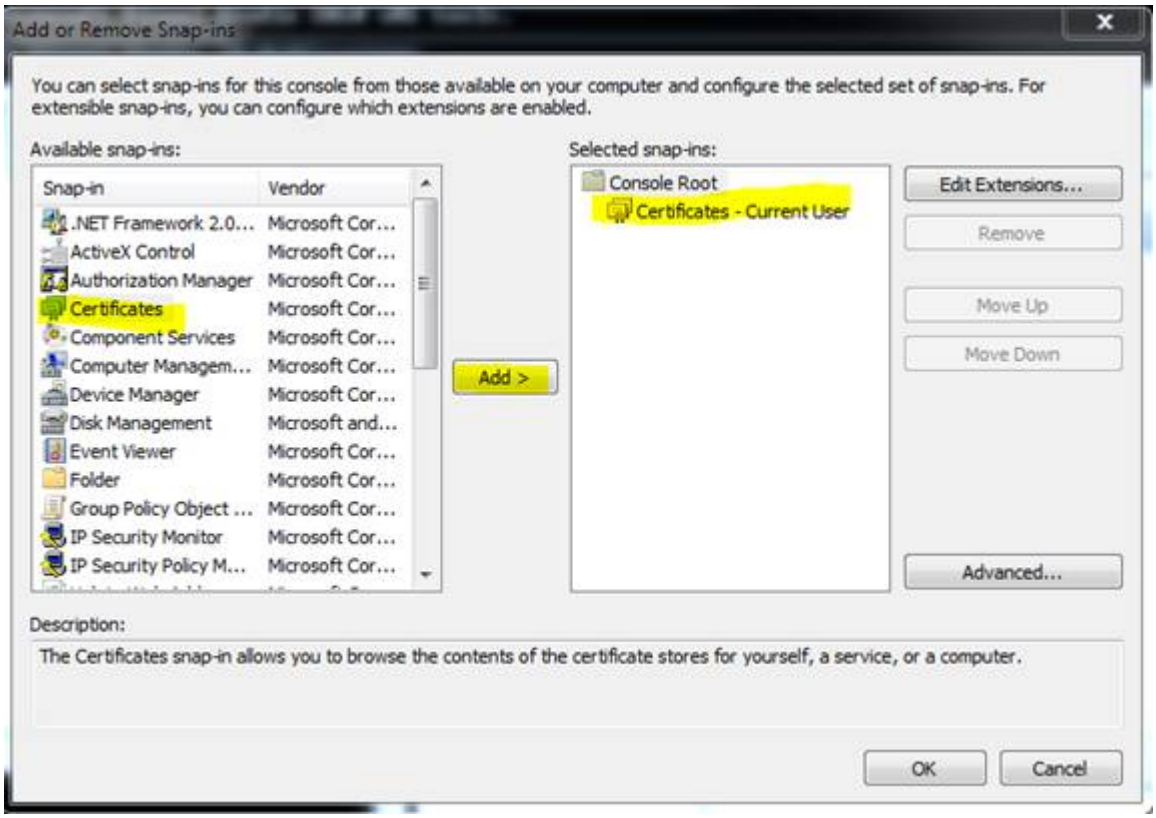
概要

証明書は Microsoft Management Console で削除できます：

1. Startメニューまたはユーザインターフェースから管理コンソール MMC.exeを起動します。

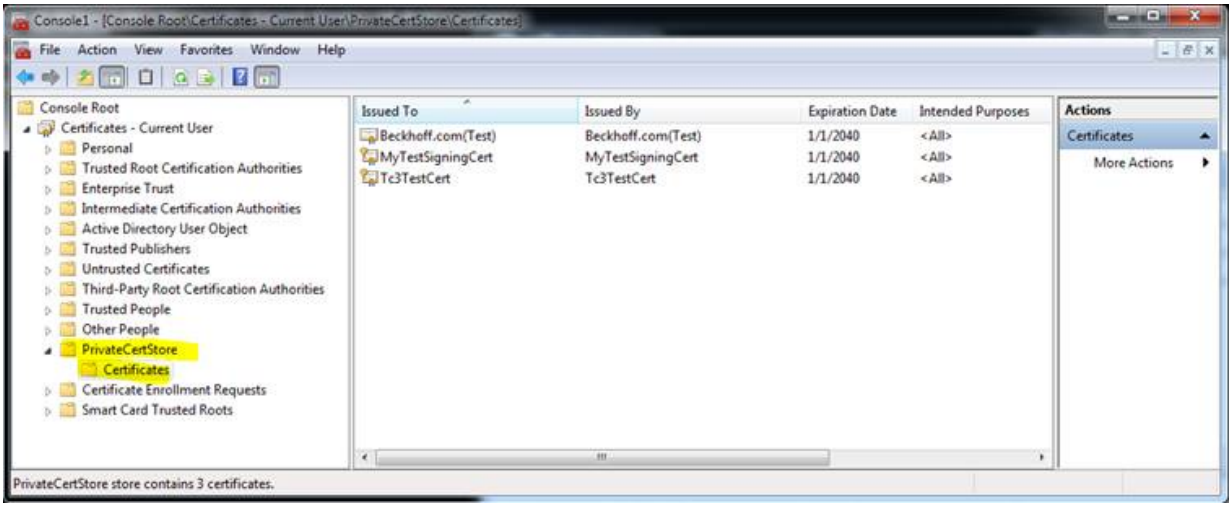


2. メニューの**File->Add/Remove Snap-in...**をクリックし、現在のユーザの証明書スナップインを選択し、**OK**で決定します。



⇨ 証明書は、ノード内の**PrivateCertStore/Certificates** にリストされます。

3. 削除する証明書を選択します。



6.5.14.1.2.2.2 テストモードなしのWindowsドライバ.3

Windows OSの場合、ドライバは「Attestation Signing（認証署名）」によって署名する必要があります。これには、EV証明書が必要です。
Microsoft は、以下のページで関連手順を提供しています：https://docs.microsoft.com/en-us/windows-hardware/drivers/dashboard/attestation-signing-a-kernel-driver-for-public-release
この方法で作成されたドライバは、セキュアブートが有効なデバイスにも対応しています。

Microsoftは、CrossSigning証明書（/acをパラメータとする署名ツール）を使用した従来の手順を2021年7月から廃止すると発表しました。この日付を過ぎると、（個々のCrossSigning証明書の有効期限によって）使用できなくなります（こちらを参照）。

TwinCAT Loader経由でロードされるバージョン管理されたC++プロジェクトは、以前からTwinCAT C++で使用できます。ただし、これらはオペレーティングシステム上ではドライバとは見なされません。ベッコフでは、バージョン管理されたC++プロジェクトの使用を推奨しています。

TwinCAT C++ドライバをバージョン管理されたC++プロジェクトに移行するためのガイドが、How toセクションに用意されています。

6.5.14.1.2.3 TcSignTool - プロジェクト外の証明書パスワード保存

TcSignToolを使用して、TwinCATユーザ証明書のパスワードをレジストリに保存できます。そのため、パスワードがバージョン管理システムに意図せず残ってしまうようなプロジェクトでは、パスワードは必要ありません。

TcSignToolは、パス*C:¥TwinCAT¥3.x¥sdk¥Bin¥*にあるコマンドラインプログラムです。

パスワードの保存は以下のパラメータで行われます：

```
tcsigntool grant /f "C:\TwinCAT\3.1\CustomConfig\Certificates\MyCertificate.tccert" /p MyPassword
```

パスワードは以下のパラメータで削除されます：

```
tcsigntool grant /f "C:\TwinCAT\3.1\CustomConfig\Certificates\MyCertificate.tccert" /r
```

暗号化されていないパスワードはHKEY_CURRENT_USER¥SOFTWARE¥Beckhoff¥TcSignTool¥以下に保存されます。

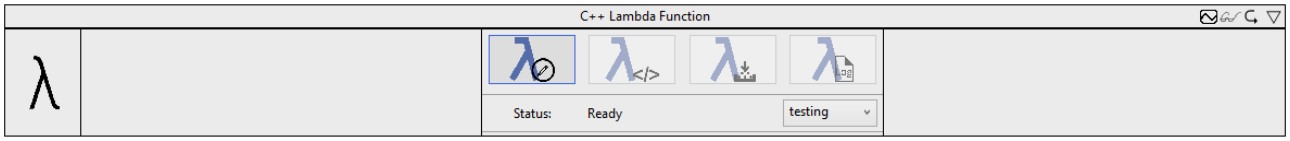
6.5.14.1.2.4 環境変数

最後に、証明書名とともに2つの環境変数を作成する必要があります。これらはラムダ関数の生成時に読み込まれ、対応する証明書でラムダ関数に署名されます。

変数名	変数値
TWINCATTESTCERTIFICATE	<オペレーティングシステム証明書名>（例：MyTestSigningCert）
TWINCATCERTIFICATENAME	<TwinCAT証明書名>（例：TwinCATTestCert）

ファイル拡張子（.certと.tccert）は含めないでください。

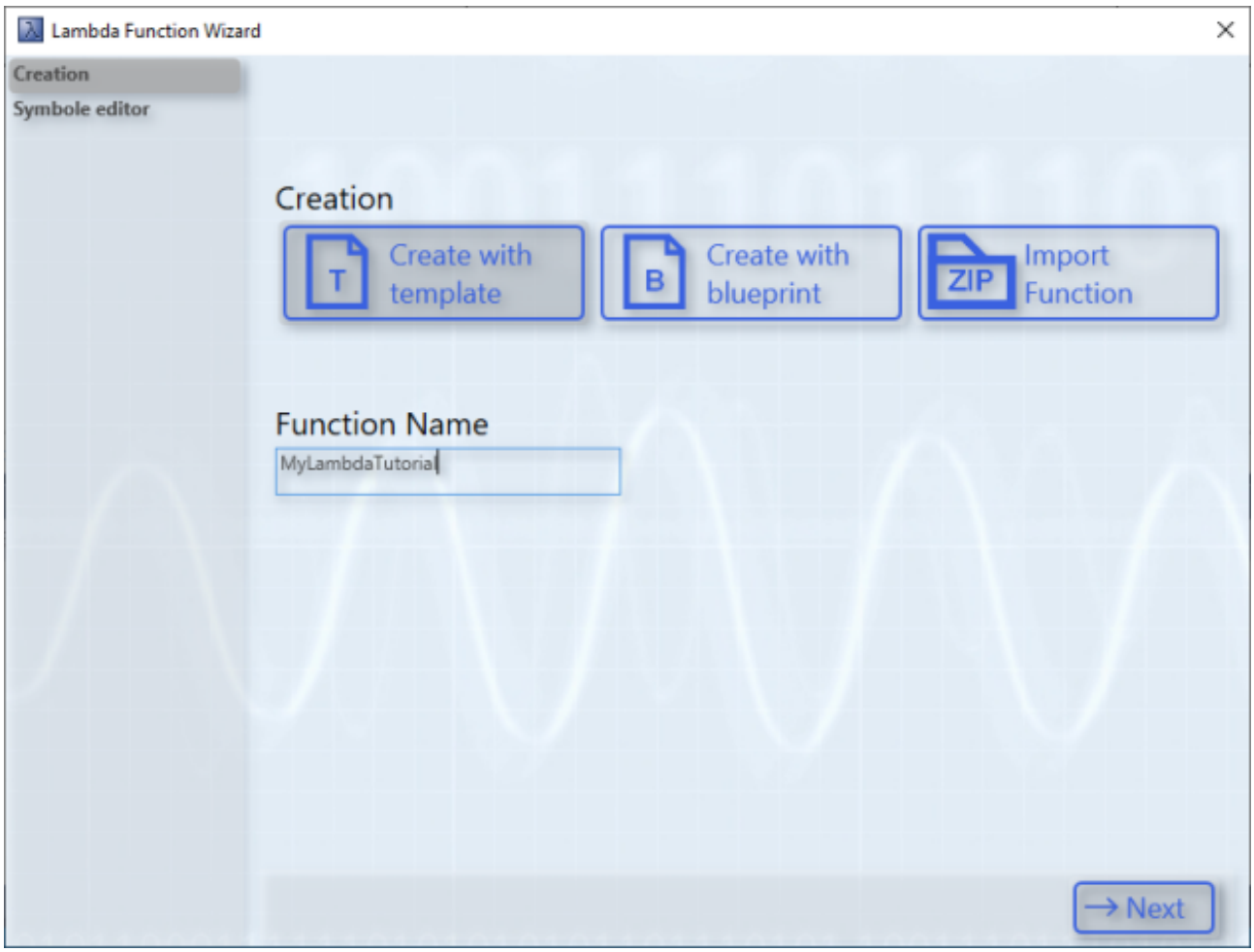
6.5.14.1.3 ラムダテンプレート



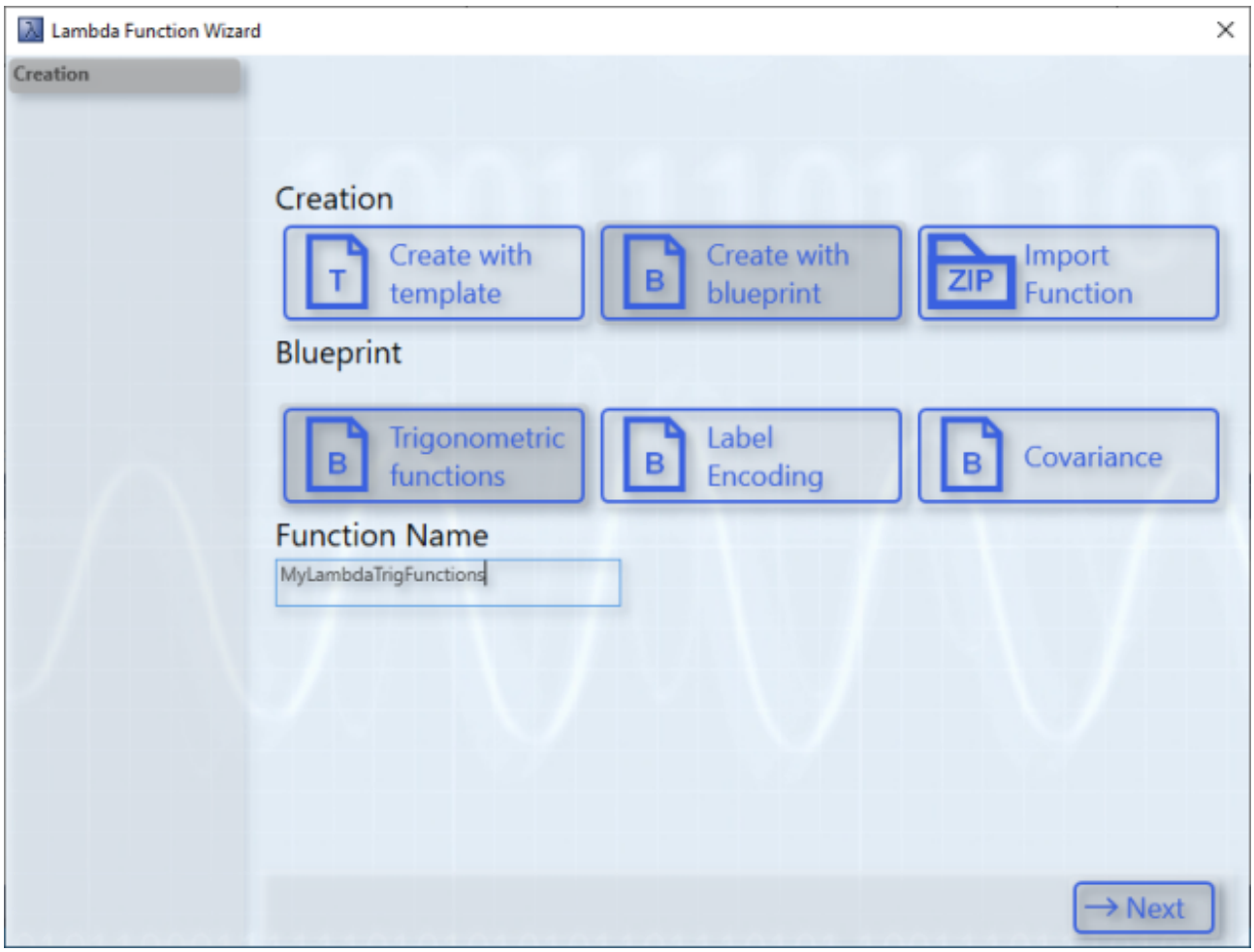
ラムダテンプレートはラムダ関数の開発をガイドします。パラメータエリア内にコントロールがあります。このコントロールには、ラムダ関数の設定、プログラミング、パブリッシュを行うための4つのボタンがあります。4つ目のボタンは、開発プロセスのログファイルを見るためのものです。ステータスバーは、プロセス中に追加情報を提供します。右下のドロップダウンメニューで、ラムダ関数のリリース版とテスト版から選択できます。

ラムダ関数の設定

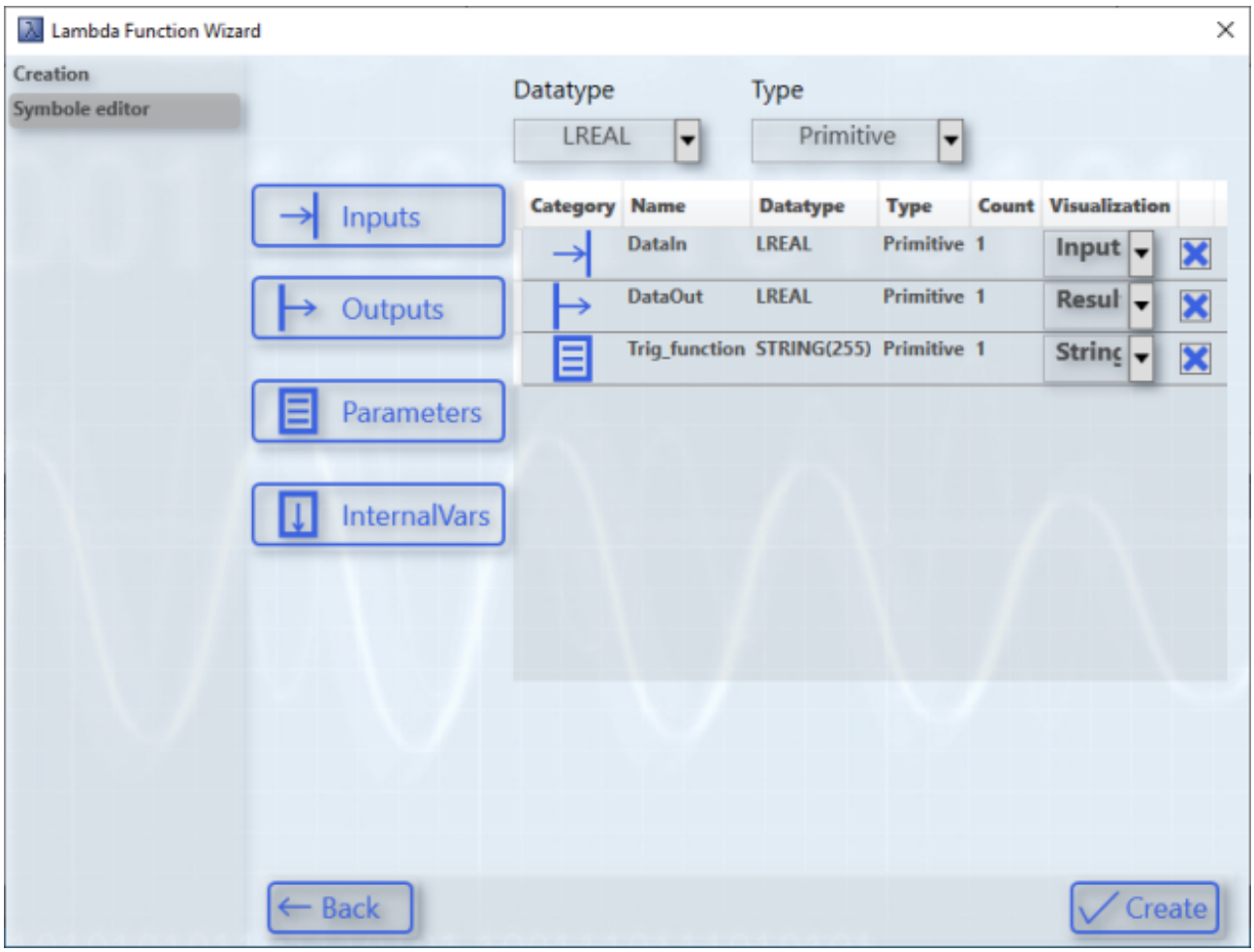
左から1つ目のボタンはウィザードを開きます。このウィザードを使ってラムダ関数を設定できます。



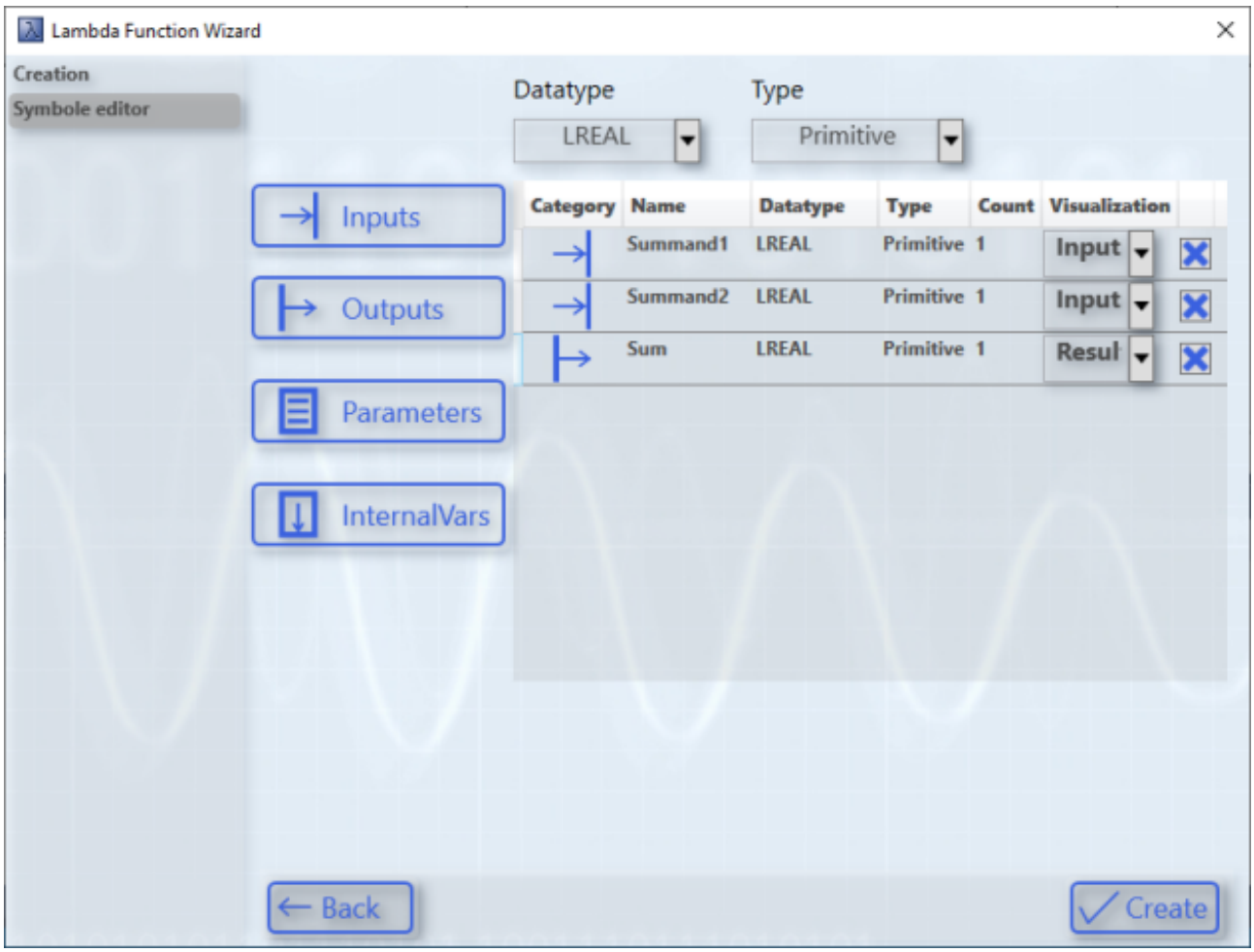
最初のページでは、3つのモードから選択できます。1つ目のモードはTemplate（テンプレート）と呼ばれます。このモードでは、新しいラムダ関数を開発できます。全てのファイルと依存関係はテンプレートとして用意されており、ユーザ固有のロジックで実装可能です。



2つ目のモードはBlueprint（ブループリント）と呼ばれます。いずれのケースにも、既に3つのブループリントが実装されています。三角関数の計算、ラベルエンコーディング、共分散の計算のためのブループリントがあります。これらのブループリントは、ラムダ関数の設定と実装の例として役立ちます。さらに詳しい情報は、以下のリンク（ラムダ ブループリント [▶ 298]）を参照してください。
3つ目のモードでは、エクスポートしたラムダ関数をインポートします。**Import Function**ボタンをクリックすると、Open Fileダイアログが開きます。次に、ZIPアーカイブとしてエクスポートされたラムダ関数を選択します。ラムダ関数がインポートされるとすぐに、名前が認識され、設定したシンボルが次のページに表示されます。
ラムダ関数の名前は個別に選択できます。ただし、ラムダ関数の名前にあるバージョン番号は、調整や変更はできません。変更すると、ラムダ関数が適切に使用できない可能性があります。関数名は少なくとも4文字以上である必要があります。
ウィザードの2ページ目では、ラムダ関数のシンボルと変数を設定します。
ブループリントモードでは、一部のシンボルがあらかじめ設定されています。



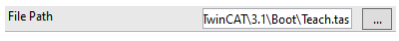
テンプレートモードでは、シンボルのリストは空です。下図に示すように、ユーザ固有のシンボルを設定できます。

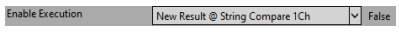
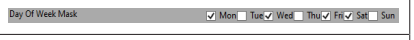


ページの1番上にある2つのドロップダウンボックスを使って、データ型とデータ構造型をそれぞれ選択します。

使用可能なデータ型				
LREAL	REAL	BOOL	WORD	INT
LINT	ULINT	LTIME	STRING(255)	

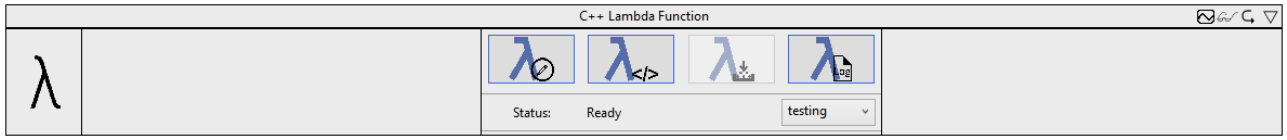
データ構造型として、プリミティブ型または配列型を選択できます。
右側のボタンを使用して、選択したデータ型とデータ構造型で変数を追加してください。
内部変数はメンバ変数です。これらはアルゴリズムインターフェースには表示されず、Analytics Workbenchではリンクできません。これらの目的は、サイクルをまたいで変数の値を保持することです。
変数にデータ構造型「Array」が選択されている場合、配列のサイズはその変数の「Count」列で指定できます。多次元配列を作成するには、次元を文字 「x」 で区切る必要があります（例：2x4）。
Workbenchで、ドロップダウンメニューを使用して、シンボルの可視化を設定できます。データの種類によって、様々な可視化が可能です。定義されていない可視化は実行できません。内部変数はWorkbenchに表示されないため、可視化は選択できません。入力、出力、パラメータのそれぞれのデータ型について、以下の可視化が可能です：

データ型	可視化	サンプル
STRING(255)	String (入力)	
	String（出力）	
	String (パラメータ)	
	FileOpen/FileSave（パラメータ）	
ULINT	Timestamp（出力）	

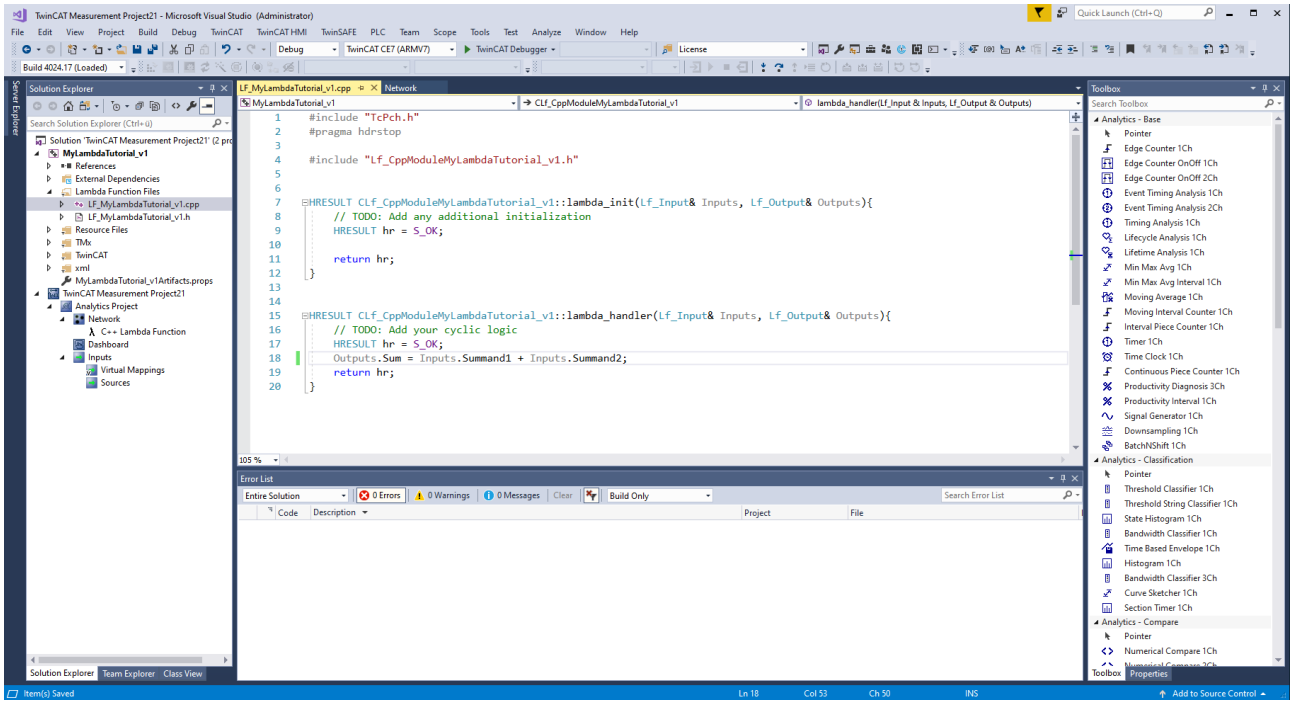
データ型	可視化	サンプル
BOOL	Boolean (入力)	
	Boolean（出力）	
	Boolean (パラメータ)	
REAL	Input （入力）	
	Result （出力）	
INT	Count （出力）	
LINT	Timespan （出力）	
	Clock （パラメータ）	
	Interval （パラメータ）	
WORD	DayOfWeekMask (パラメータ)	
LREAL	Input （入力）	
	Result （出力）	
	Level （パラメータ）	

Createボタンをクリックすると、C++プロジェクトがバックグラウンドで生成され、ウィザードが終了します。

ラムダ関数のプログラミング



2番目のボタンをクリックすると、C++プロジェクトがSolution Explorerに追加されます。このプロジェクトには、Analytics Workbenchでリアルタイムに実行可能なTcCOMモジュールを開発するための全てのファイル、依存関係、設定が含まれています。



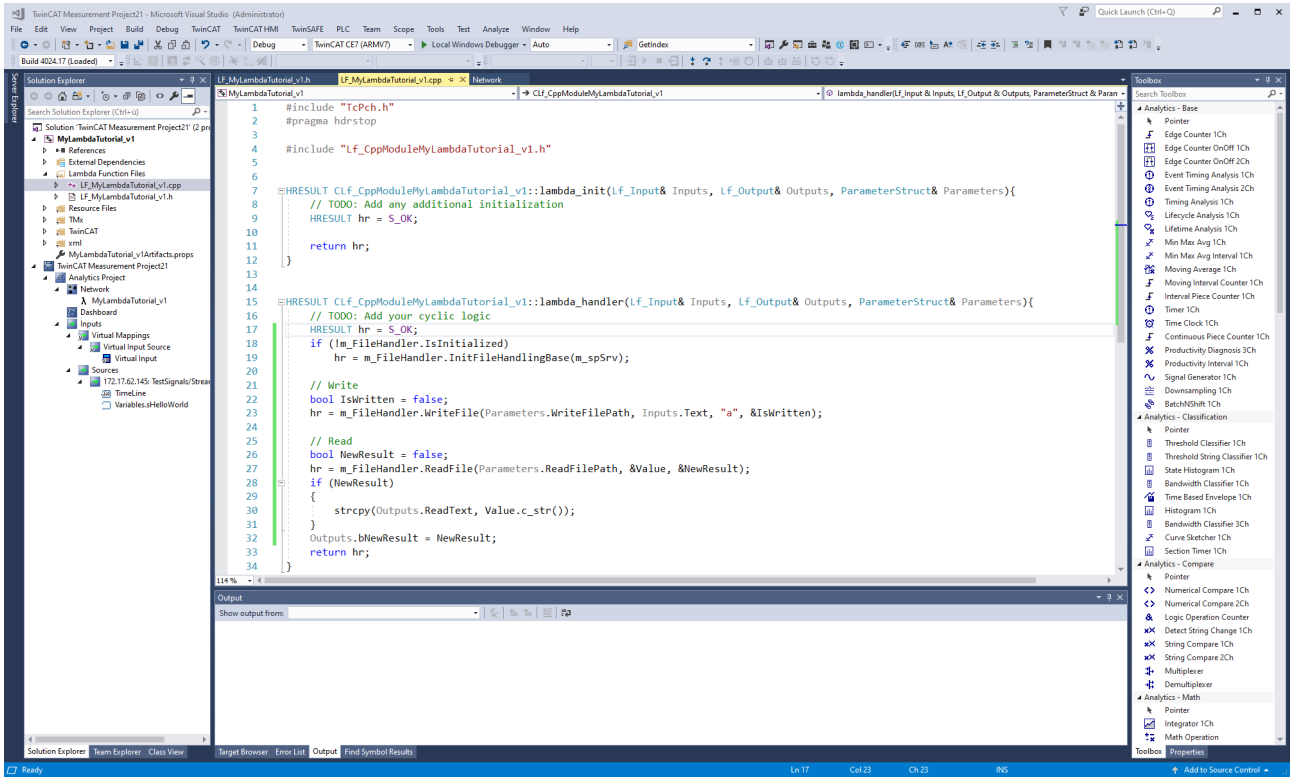
自動的に開くファイルには、複数のカプセル化したファンクションが含まれています。lambda_initメソッドは変数の初期化に使用されます。lambda_handlerメソッドは周期的に実行されます。lambda_uninitメソッドは、変数の初期化を解除するためのものです。いずれのメソッドでも、入力、出力、パラメータ、内部変数は、メソッドのパラメータとして構造体で利用できます。ラムダ関数では、デフォルトで以下の入出力が利用できます。

カテゴリ	シンボル名	データ型	説明	表示
入力	tTimestamp	ULINT	この変数にはTwinCATのタイムスタンプが含まれます。	非表示
	bReset	BOOL	この変数は設定されたシンボルをリセットします。	オプション
	bEnableExecution	BOOL	この変数はlambda_handlerメソッドを有効化します。	オプション
出力	bNewResult	BOOL	この変数はアルゴリズムが新しい結果を返すかどうかを示します。	オプション

メソッド内で作成された変数は、1サイクル後にその内容を損失します。内部変数は、サイクルをまたいで変数の値を保持することができます。オブジェクト「m_FileHandler」は、2つのラムダメソッドで利用できます。まだ初期化していない場合は、このオブジェクトを一度、初期化する必要があります。その後、オブジェクトの2つのファンクション 「ReadFile 」と 「WriteFile 」を使用できます。ファンクションには次の特性があります：

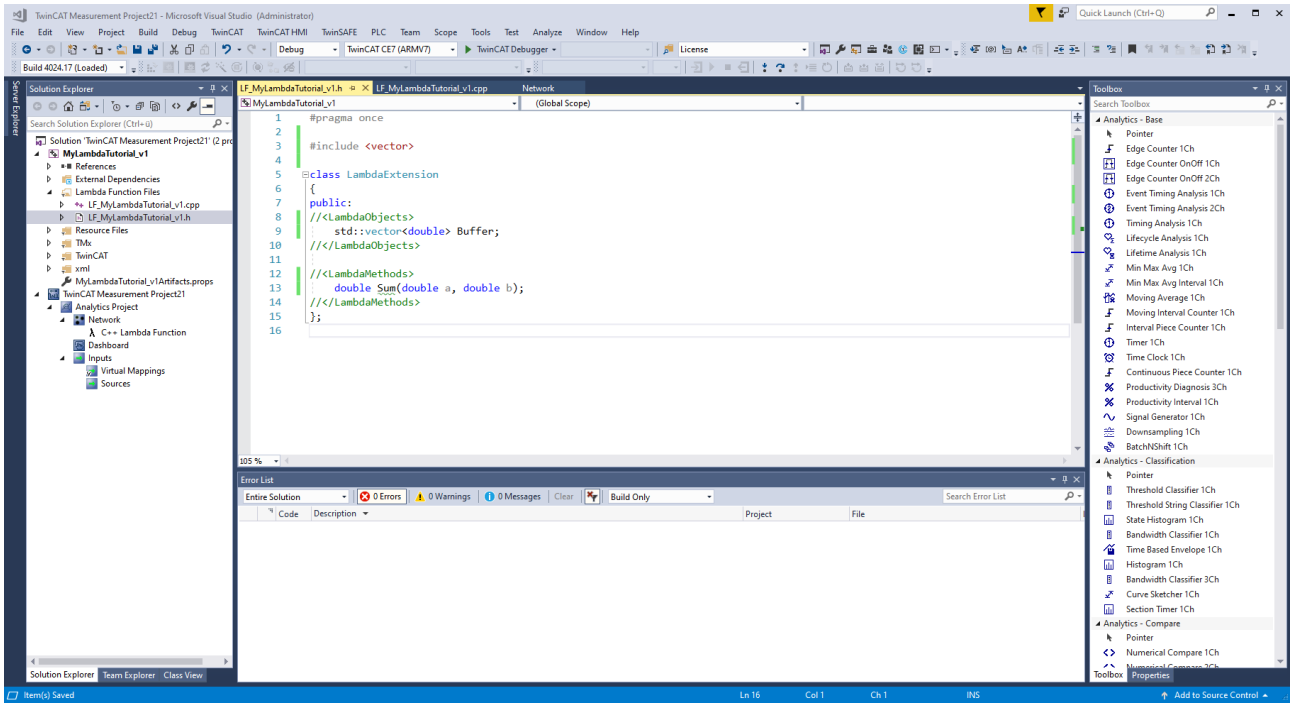
ファンクション名	パラメータ				戻り値
InitFileHandling ベース	ITComObjectServer* ipSrv				Hresult
ReadFile	PCCH filename	Std::string* value	bool* bNewResult		Hresult
WriteFile	PCCH filename	Std::string value	Std::string mode	bool* bIsWritten	Hresult

ファイルアクセスは、TwinCAT Analytics Workbenchでは同期アクセスですが、Analytics Runtimeでは非同期アクセスです。この機能はメソッド中で自動的に区別されます。非同期のReadアクセスの場合、メソッドは同じサイクルで読み取り値を返しません。パラメータbNewResultが値trueを返すと同時に、パラメータ値に読み取り値が含まれます。非同期のWriteアクセスでは、データは内部に保存され、次のタイミングで書き込まれます。データが書き込まれると、bIsWrittenパラメータはtrueを返します。bNewResultとbIsWrittenの2つのパラメータは、同期ファイルアクセスにも同様に使用できます。次の図は、ラムダ関数におけるファイルアクセスの使用例を示しています：

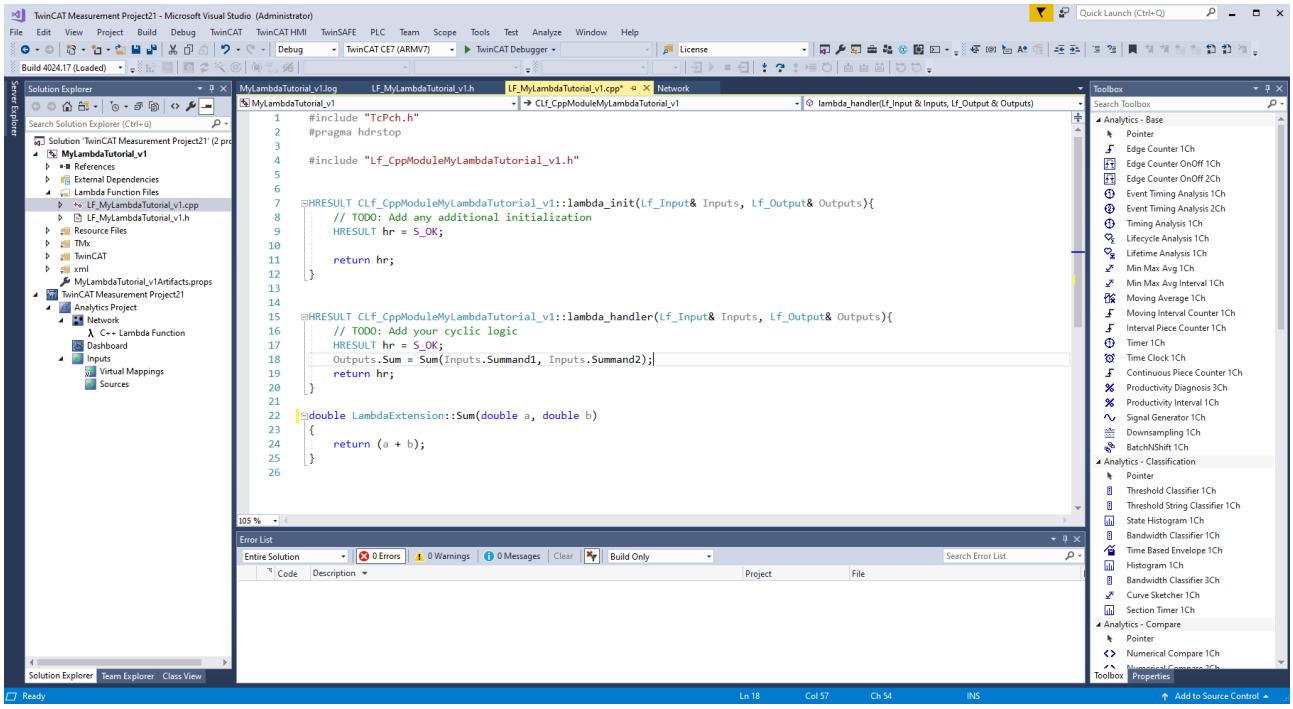


Visual Studio®で通常通りプロジェクトを作成し、構文をチェックします。

「Lambda Function Files 」フォルダ内のヘッダーファイルで、ユーザ固有のオブジェクトやメソッドを宣言できます。



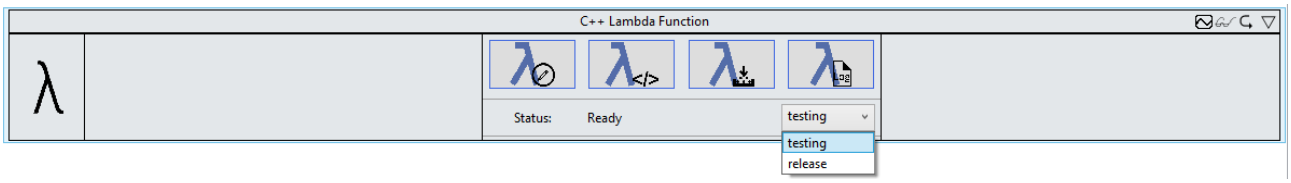
クラス 「LambdaExtension 」の上には、ライブラリを含めることができます。クラス内では、オブジェクトとメソッドを宣言し、ラムダ関数で使用できます。新しく宣言されたメソッドは、2つのラムダ関数の下に実装できます。



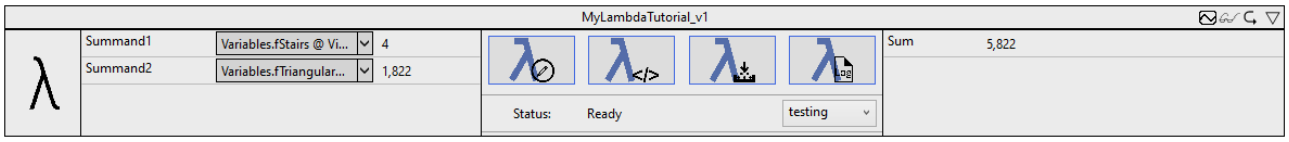
ロジックが統合されたら、プロジェクトを保存する必要があります。その後、Analyticsプロジェクトのラムダテンプレートに戻ります。
TcCOMモジュールを取り扱うのに十分な知識がある場合、C++プロジェクトを個別に拡張できます。TcCOMモジュールが適切に使用されない場合、ラムダ関数の後続のステップでエラーが発生する可能性があります。TcCOMモジュールの開発に関する詳細は、以下のリンク を参照ください。

ラムダ関数のパブリッシュ

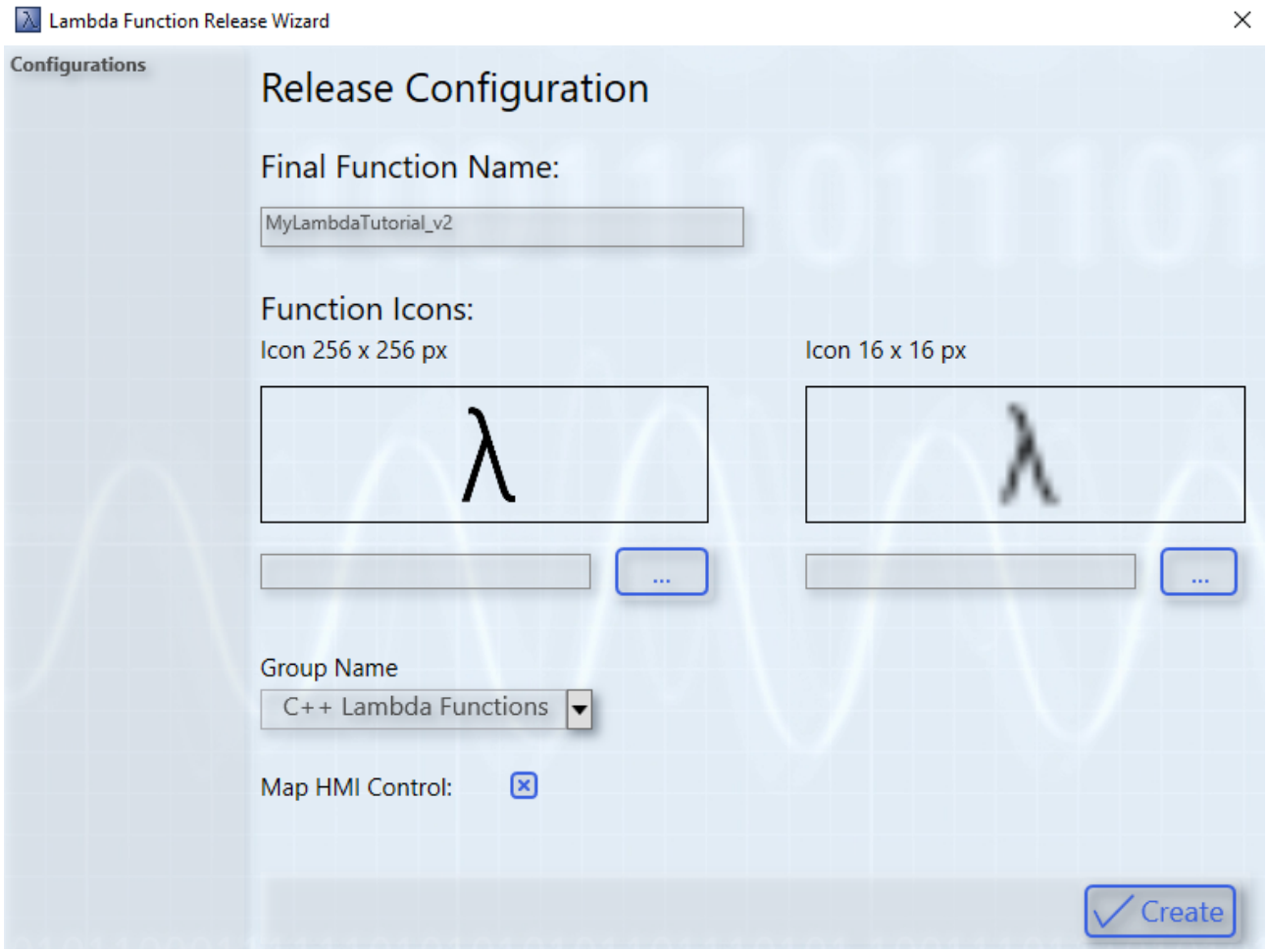
パブリッシュする前に、右下のドロップダウンメニューを使ってラムダ関数のタイプを設定できます。



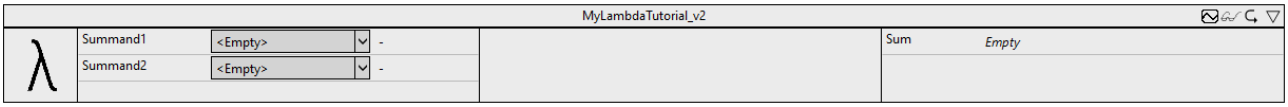
「testing 」タイプのラムダ関数には、関数を開発するための4つのボタンが含まれます。このタイプは、開発やテスト段階で役立ちます。



ラムダ関数が動作したら、「release」タイプを選択します。3番目のボタンをクリックすると、Releaseウィザードが開きます。



ウィザードでは、最終的な名前を割り当て、個別のアイコンを選択できます。256×256ピクセルのアイコンを1つ、16×16ピクセルのアイコンを1つ選択する必要があります。あるいは、ラムダアイコンを使い続けることもできます。ドロップダウンメニューを使って、ラムダ関数をツールボックスの特定のグループに割り当てます。デフォルトでは、「C++ Lambda Functions」グループが選択されています。例えば、最後のラムダ関数は、ツールボックスの「Analytics - Base」セクションに配置できます。「Map HMI Control」チェックボックスを有効にすると、パブリッシュ後にHMIコントロールとラムダ関数をリンクするためのウィザードが開きます。Analytics HMI Control Managerでは、モジュール選択のポイントが、対応するラムダ関数によって直接事前選択されます（カスタマイズされた独自コントロールの使用 [▶ 438]を参照）。ラムダ関数を生成するには、Createボタンを使用します。



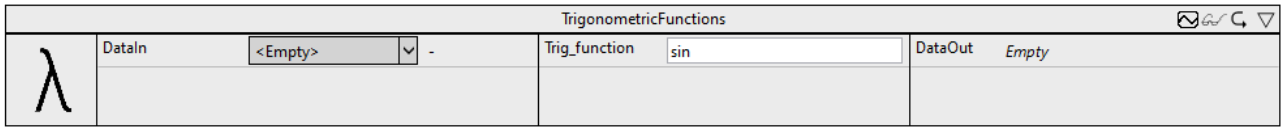
パブリッシュに成功すると、ラムダテンプレートは新しく生成されたラムダ関数に自動的に置き換えられます。

ログファイルの閲覧

ラムダ関数のパブリッシュ中にエラーが発生した場合、ステータス表示にエラーが表示され、4番目のボタンからログファイルを見ることができます。

6.5.14.1.4 ラムダ ブループリント

Triginometric function（三角関数）



ブループリントの**Triginometric function**は、入力チャンネルで三角ファンクションの計算を行い、結果を出力チャンネルに返します。入力チャンネルは、角度をラジアン単位で受け取ります。それぞれの三角関数は、パラメータ領域で指定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

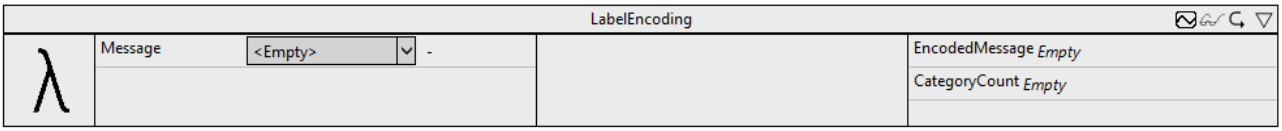
設定オプション

- **Trig_function**: 計算する三角関数の略称（例：sin は正弦関数）。選択肢には、sin、cos、tan、asin、acos、atanがあります。asinとacosの入力値は、[-1,1]の区間にある必要があります。

出力値

- **DataOut**: 三角関数の計算結果を出力します。

Label Encoding



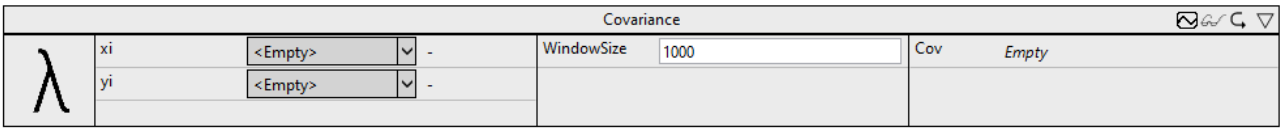
ブループリント**Label Encoding**は、入力チャンネルのテキストを数値に変換します。この目的のため、テキストは入力チャンネルで異なるカテゴリに分類されます。同じ入力テキストは同じ数値を返します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

出力値

- **EncodedMessage**: この出力チャンネルは、対応するカテゴリのインデックスを返します。
- **CategoryCount**：この出力チャンネルはカテゴリ数を出力します。

Covariance



ブループリント**Covariance**は、2つの入力チャンネル間の共分散を計算します。WindowSizeは、共分散の計算に含まれる値の数を指定します。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

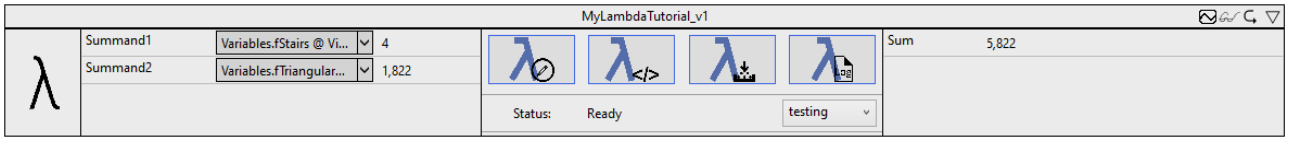
- **WindowSize**：このパラメータは、共分散が計算されるサイクル数を指定します。ブループリントにはスライディングウィンドウが実装されています。現在値は解析に含まれ、出力はサイクルごとに更新されます。このパラメータを設定する際は、ルーターのメモリサイズを考慮する必要があります。

出力値

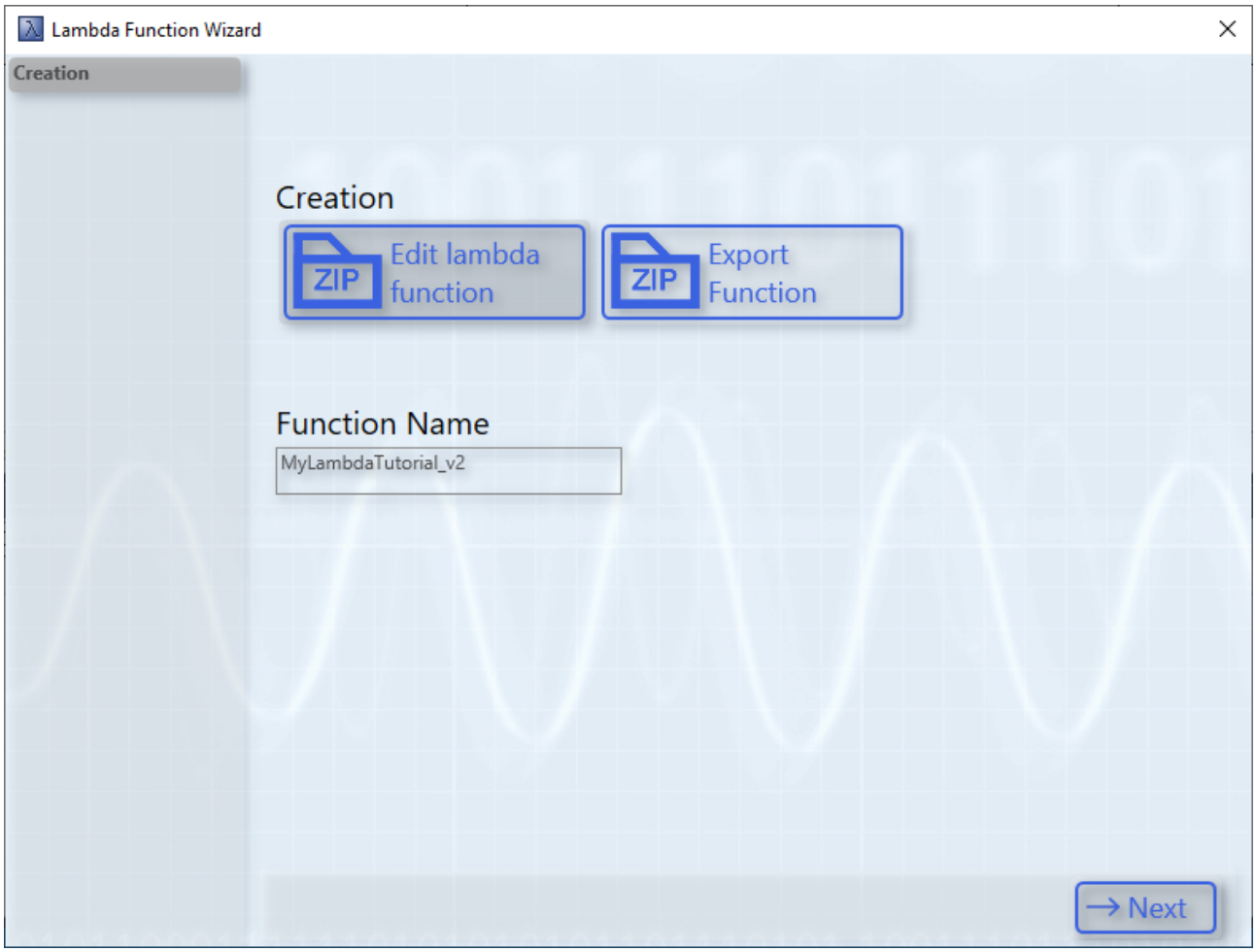
- **Cov**：この出力チャンネルは、設定されたウィンドウサイズにおける2つの入力チャンネルの共分散を示します。

6.5.14.1.5 ラムダ関数

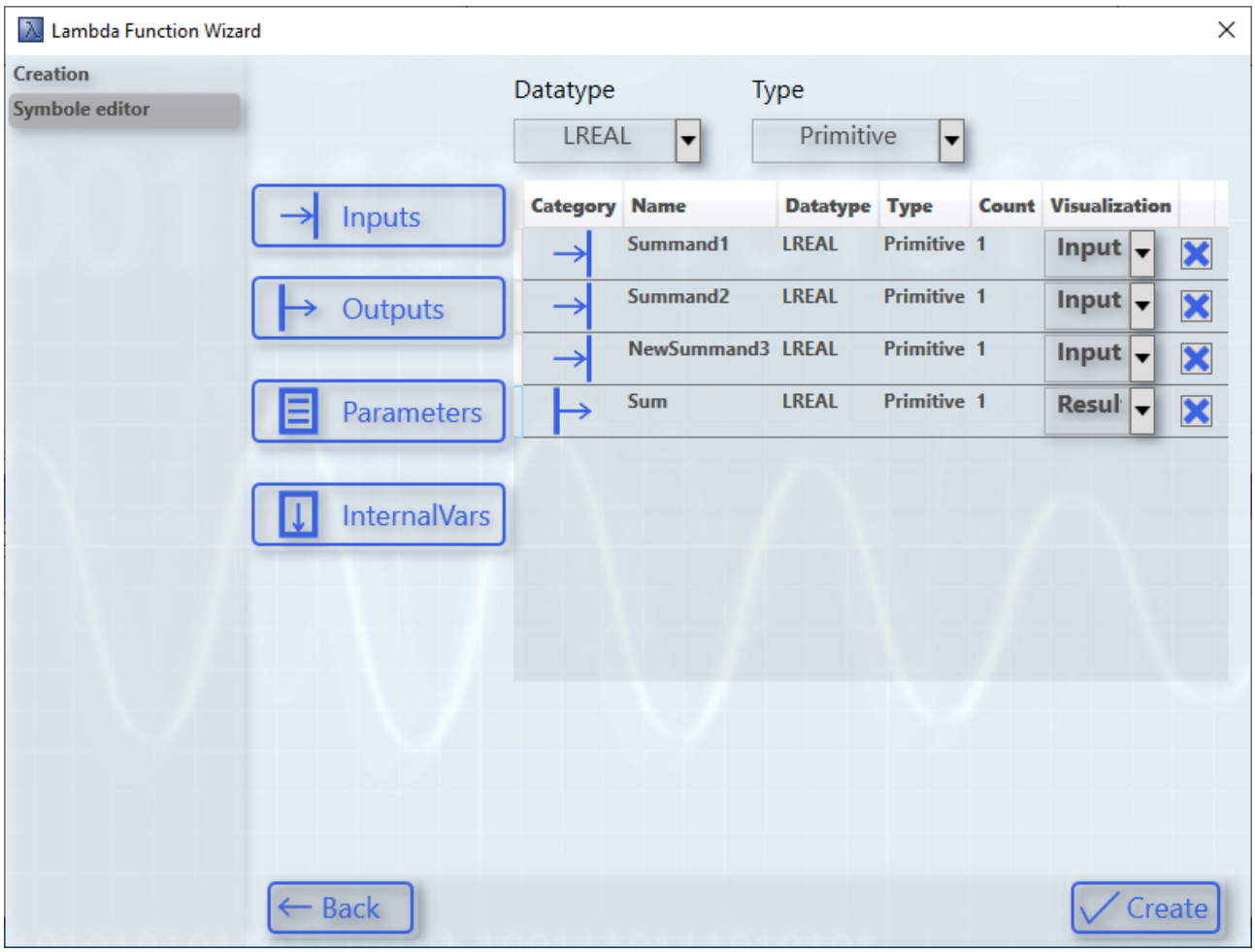
開発・テスト段階では、ラムダ関数はテスト版としてパブリッシュする必要があります。これらには、設定したシンボルと、4つのボタンを備えたLambda Controlが含まれます。



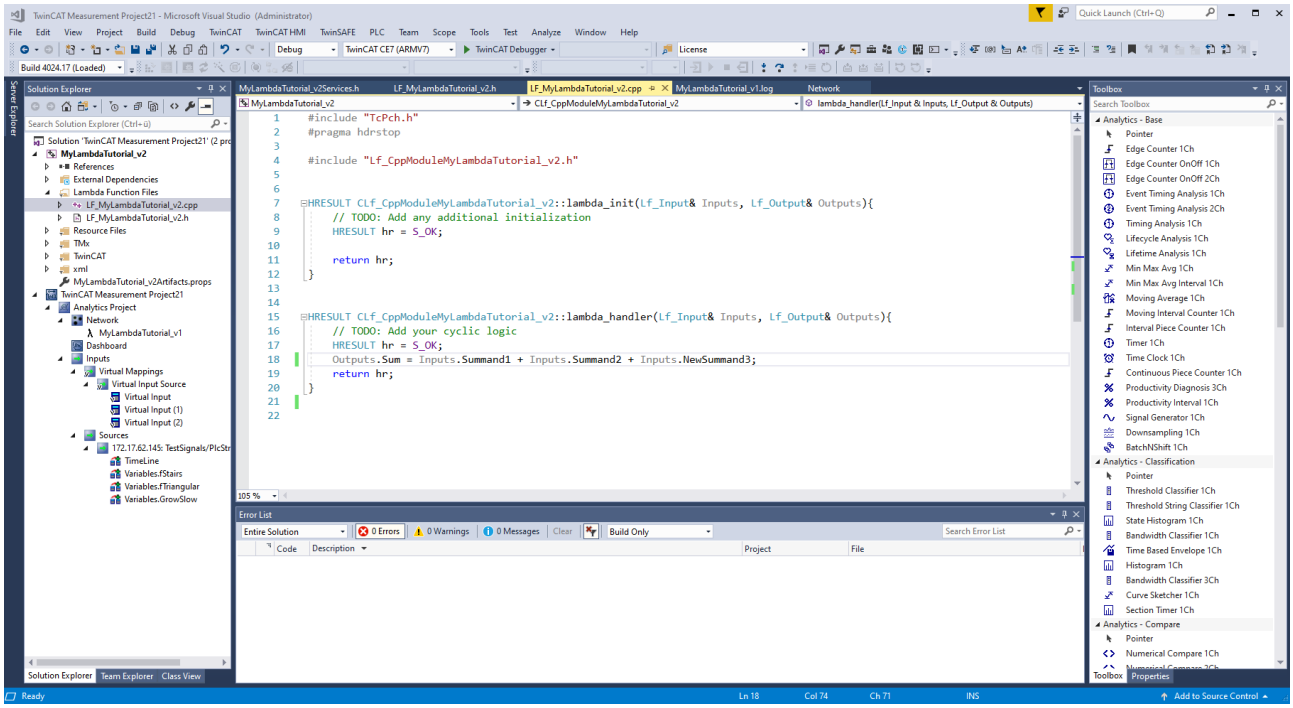
このラムダ関数は、Analytics Workbenchの他のアルゴリズムと同様にテストできます。変更が必要な場合は、Lambda Controlがその後の開発手順をガイドします。最初のボタンをクリックすると、ラムダ関数を設定するためのウィザードが開きます。既存のラムダ関数のウィザードには2つのモードがあります。



Export Functionボタンをクリックすると、ラムダ関数をエクスポートできます。この目的のため、C++プロジェクトとラムダ関数のドライバは zip アーカイブとして保存されます。ラムダテンプレートのインポート機能を使えば、ラムダ関数を他のシステムにインポートして統合できます。**Edit lambda function**モードでは、既に開発されたラムダ関数を編集できます。2つ目のボタンは、現在のラムダ関数をエクスポートするために使用します。このために Save File ダイアログが開きます。このダイアログを使ってファイルの場所を指定できます。選択後、エクスポートが開始されます。エクスポートされたラムダ関数は、他のシステムに展開できます。他のシステムでは、ウィザードを使ってラムダテンプレートからラムダ関数をインポートできます。
Edit lambda functionモードでは、ウィザードの次のページでシンボルをカスタマイズできます。



この例では、「NewSummand3 」という別の入力が追加されています。これらの変更は、次のステップでC++ファイルにも反映されます。



新しい変数は入力から構造体に反映され、プログラムコードで使用できます。前バージョンのプログラムコードが引き継がれます。バージョン間でシンボルが削除された場合は、プログラムコードにも対応する変更を行う必要があります。
ラムダ関数の開発における以降のステップは、ラムダテンプレートの手順と同様です。

ラムダ関数のデバッグ

ラムダ関数をデバッグするには、TwinCAT C++のツールを使用する必要があります。このためには、新規TwinCATプロジェクトを作成し、ラムダ関数のC++プロジェクトをC++ノードの下に挿入する必要があります。全てのTwinCAT C++デバッグオプションが使用できます（ を参照）。Debugタブを使って、TcScopeServerプロセスにアタッチできます。その後、Workbenchで解析を開始し、C++プロジェクトで設定されたブレークポイントに到達します。

ツールボックスの整理

複数のラムダ関数が開発、テストされると、ツールボックスがいっぱいになる場合があります。ツールボックスをクリーンアップするには、ラムダ関数をModulesフォルダから削除する必要があります。このフォルダは*<TwinCAT3Dir>/CustomConfig/Modules*のパスにあります。

6.5.14.1.6 ラムダ関数 その他の手順

ランタイムへのデプロイ

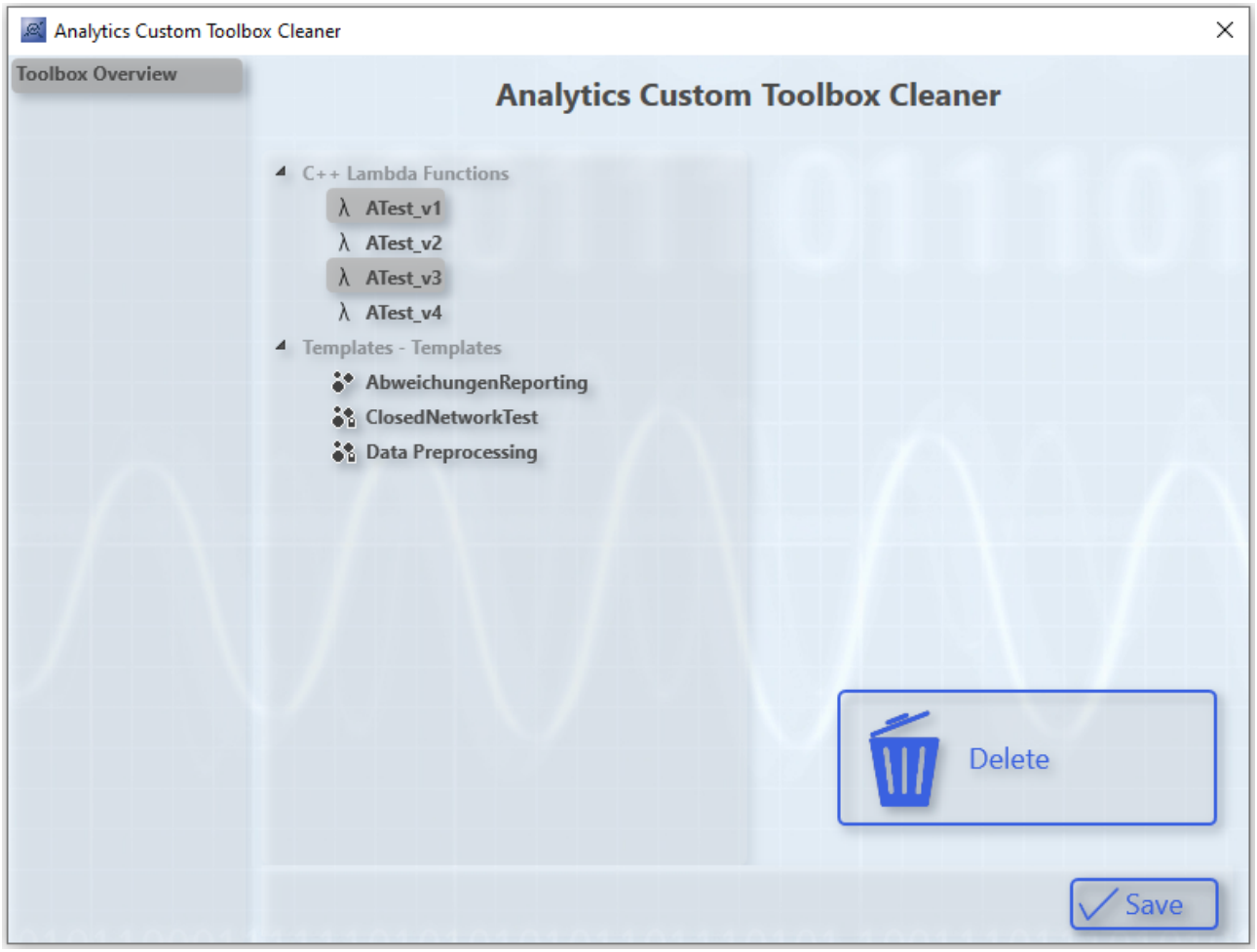
ラムダ関数は実行時のデプロイにも適しています。この目的のため、開発したラムダ関数をTwinCATプロジェクト内にTcCOMモジュールとして作成します。PLC ファンクションブロックが TcCOM モジュールを処理します。この目的のため、PLC ファンクションブロックは、ラムダ関数のインターフェースメソッドを呼び出します。ラムダ関数のインターフェースメソッドと関連する PLC ファンクションブロックは自動的に生成されるため、明示的な設定は不要です。これにより、ユーザ固有のラムダ関数をTwinCAT Analytics Runtimeにダウンロードし、独自の常時データ解析を実現できます。詳細は以下のリンクを参照してください（ランタイムへのデプロイ [▶ 357]）。

HMIワンクリックダッシュボード

ラムダ関数の結果を表示するには、HMIワンクリックダッシュボードへの統合が理想的です。新しいコントロールは、AnalyticsプロジェクトのDashboardノードからリンクできます。Analytics Dashboardウィザードでは、HMI コントロールと対応するラムダ関数を選択してリンクできます。Analytics Dashboardウィザードの詳細については、以下のリンクを参照してください（Analyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理 [▶ 428]）。

6.5.14.1.7 Analytics Custom Toolbox Cleaner

ネットワークテンプレートとラムダ関数を使用して、様々なツールボックスの要素を作成できます。これらの要素を簡単に削除するには、Analytics Custom Toolbox Cleanerを利用できます。次の図はウィザードを示しています：



ツールボックスの各要素はツリー構造から選択できます。CTRLを押しながら複数の要素を選択できます。Deleteボタンで構造から要素を削除します。Saveボタンをクリックして変更を確定します。Saveボタンをクリックしないでウィザードを閉じた場合、変更は適用されません。

6.5.14.2 その他のTwinCATライブラリのアルゴリズム

TwinCAT FilterとTwinCAT Condition Monitoringライブラリを実装することで、さらに多様なアルゴリズムを利用できます。これらのアルゴリズムをTwinCAT Analyticsに統合することで、プログラミング作業が純粋な設定作業に変わります。これにより、解析作成にかかる労力が削減されます。これらのアルゴリズムをTwinCAT Analytics Workbenchコンフィグレータや Service Toolで、Analyticsライブラリのアルゴリズムと共に利用できるようになります。他のアルゴリズムグループと同様に、ツールボックスで見つけることができます。追加のアルゴリズムは、エンジニアリングモジュールとして使用できます。追加のライセンスは不要です。ただし、アルゴリズムをTwinCAT Analyticsランタイムにデプロイし、HMIワンクリックダッシュボードを生成するには、各ライブラリのライセンスが必要です。次のリストは、統合できるライブラリとアルゴリズムの概要、デプロイまたはダッシュボード生成に必要なライセンス、および各PLC ファンクションブロックのドキュメントへのリンクを示しています。

ライブラリ	アルゴリズム	ライセンス	ドキュメント
TwinCAT Filter	*Filter*	TF3680 TC3 Filter / TF3600 TC3 Condition Monitoring	
	PT1		PT1 [▶ 320]
	IIRCoeff		IIRCoeff [▶ 321]
	IIRSpec		IIRSpec [▶ 321]
	PT2		PT2 [▶ 322]
	PT3		PT3 [▶ 322]
	PTn		PTn [▶ 323]

ライブラリ	アルゴリズム	ライセンス	ドキュメント
	IIRSos		IIRSos [▶ 323]
	Notch		Notch [▶ 324]
	LeadLag		LeadLag [▶ 324]
	PT2oscillation		PT2oscillation [▶ 325]
	PTt		PTt [▶ 325]
	Median		Median [▶ 326]
	ActualValue		ActualValue [▶ 326]
	Gaussian		Gaussian [▶ 326]
TwinCAT Condition Monitoring	*Condition Monitoring*	TF3600 TC3 Condition Monitoring	
	ArgSort		ArgSort [▶ 306]
	Crest Factor		Crest Factor [▶ 306]
	Crest Factor Plus		Crest Factor Plus [▶ 307]
	DiscreteClassification		Discrete Classification [▶ 308]
	Envelope		Envelope [▶ 308]
	Envelope Spectrum		Envelope Spectrum [▶ 309]
	Fatigue Analysis		Fatigue Analysis [▶ 310]
	Instantaneous Frequency		Instantaneous Frequency [▶ 311]
	Instantaneous Phase		Instantaneous Phase （瞬時位相） [▶ 305]
	Integrated RMS		Integrated RMS [▶ 312]
	Magnitude Spectrum		Magnitude Spectrum [▶ 313]
	Multi-Band RMS		Multi Band RMS [▶ 314]
	Power Spectrum		Power Spectrum [▶ 315]
	RMS		RMS [▶ 316]
	Spike Energy Spectrum		Spike Energy Spectrum [▶ 317]
	Vibration Assessment		Vibration Assessment [▶ 318]
	Watch Upper Thresholds		Watch Upper Thresholds [▶ 319]

TwinCAT Filterライブラリにより、各種のファンクションブロックを使用して、様々なデジタルフィルタを実装できます。

TwinCAT Condition Monitoringライブラリは、機械やプラントの状態監視のための数学的アルゴリズムを提供します。

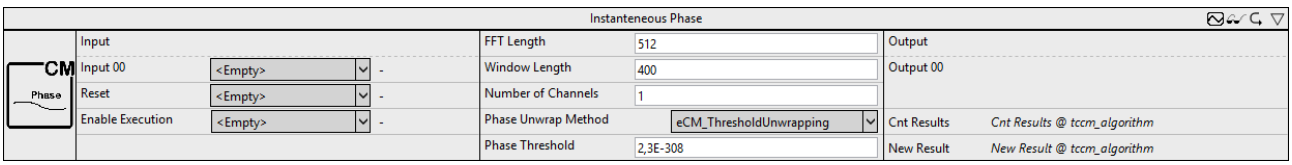
6.5.14.2.1 Condition Monitoring

TwinCAT Condition Monitoringライブラリは、機械やプラントの状態監視のための数学的アルゴリズムを提供します。

アルゴリズムの概要

アルゴリズム	説明	PLCファンクションブロック
ArgSort [▶ 306]	入力された引数をソートします。	FB_CMA_ArgSort
Crest Factor [▶ 306]	各チャンネルの時系列データに対して、シングルチャンネルおよびマルチチャンネルのクレストファクタを計算します。	FB_CMA_CrestFactor
Crest Factor Plus [▶ 307]	各チャンネルの時系列データに対して、シングルチャンネルおよびマルチチャンネルのクレストファクタプラスを計算します。	FB_CMA_CrestFactorPlus
Discrete Classification [▶ 308]	設定可能な閾値に基づいてマルチチャンネルデータを分類します。	FB_CMA_DiscreteClassification
Envelope [▶ 308]	時間信号の包絡線を計算します。	FB_CMA_Envelope
Envelope Spectrum [▶ 309]	時間信号の包絡線スペクトルを計算します。	FB_CMA_EnvelopeSpectrum
Fatigue Analysis [▶ 310]	疲労寿命解析と損傷計算をします。	FB_CMA_RainflowCounting FB_CMA_MeanStressCorrection FB_CMA_MinersRule
Instantaneous Frequency [▶ 311]	時間信号の瞬時周波数を計算します。	FB_CMA_InstantaneousFrequency
Instantaneous Phase （瞬時位相） [▶ 305]	時間信号の瞬時位相を計算します。	FB_CMA_InstantaneousPhase
Integrated RMS [▶ 312]	シングルチャンネルおよびマルチチャンネルの実数値時系列データのRMS値（オプションで積分）を計算します。	FB_CMA_IntegratedRMS
Magnitude Spectrum [▶ 313]	入力信号の実数値のマグニチュードスペクトル（振幅スペクトルともいう）を計算します。	FB_CMA_MagnitudeSpectrum
Multi Band RMS [▶ 314]	シングルチャンネルおよびマルチチャンネルの実数値時系列データに対して、設定可能な周波数帯域ごとのRMS値を計算します。	FB_CMA_MultiBandRMS
Power Spectrum [▶ 315]	実数値入力信号のパワースペクトルを計算し、オプションでデシベルスケーリングを適用します。	FB_CMA_PowerSpectrum
RMS [▶ 316]	シングルチャンネルおよびマルチチャンネルの実数値信号の時間的RMS値を計算します。	FB_CMA_RMS
Spike Energy Spectrum [▶ 317]	高周波信号成分のピークエネルギーを解析します。	FB_CMA_SpikeEnergySpectrum
Vibration Assessment [▶ 318]	実数値入力信号の振動を評価します。	FB_CMA_VibrationAssessment
Watch Upper Thresholds [▶ 319]	マルチチャンネルデータの設定可能な閾値監視を実行します。	FB_CMA_WatchUpperThresholds

6.5.14.2.1.1 Instantaneous Phase（瞬時位相）



時間信号の瞬時位相を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_InstantaneousPhase

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

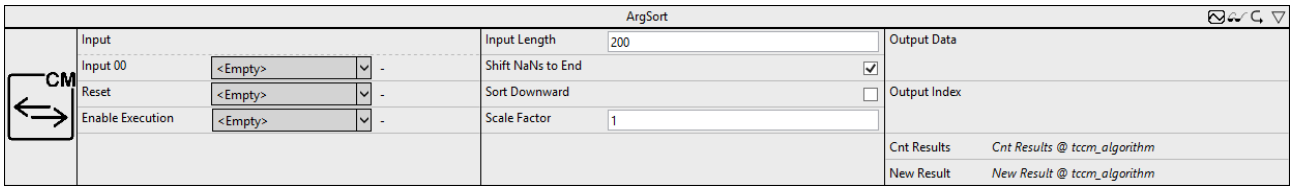
設定オプション

- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。

Phase Unwrap Method ：位相のアンラッピングに使用される方法を、2 PIの倍数の位相に関して定義します（E_CM_UnwrapMethodを参照）。

- **Phase Threshold**：瞬時位相を計算するための制限値。この値は信号の包絡線に関係します。解釈：信号レベルが低すぎる場合、位相の計算は数値的に不確実であり、信頼できる評価ができません。その場合、0が位相として出力されます。

6.5.14.2.1.2 ArgSort



入力された引数をソートします。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_CMA_ArgSort

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

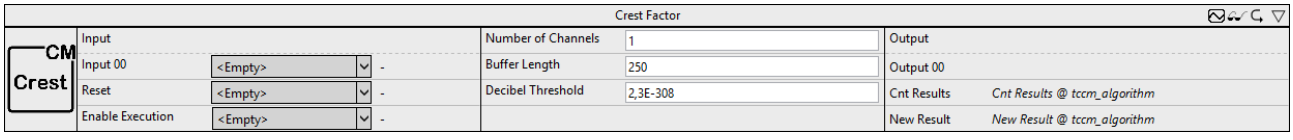
設定オプション

- **Input Length**: 入力配列の長さです。
- **Shift NaNs to End**：NaN（非数）を出力配列の末尾にシフトするかどうかを選択するフラグです。
- **Sort Downward**: データを昇順（FALSE ）または降順（TRUE ）のどちらでソートするかを選択するフラグです。
- **Scale Factor**：このパラメータは、例えば、インデックス位置の代わりに、関連する周波数を持つ振幅を直接表示するために使用できます（Scale Factor = 1）。

出力値

- **Output Data**：入力配列と同じ長さの出力配列で、オプションで昇順または降順にソートされた入力値を含みます。
- **Output Index**：入力配列と同じ長さの出力配列で、入力配列内のソートされた値に対応する（スケーラブルな）インデックスを含みます。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.3 Crest Factor



各チャンネルの時系列データに対して、シングルチャンネルおよびマルチチャンネルのクレストファクタを計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_CrestFactor

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

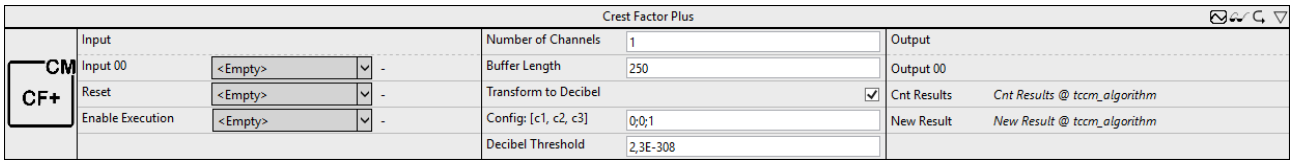
設定オプション

- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Buffer Length**：内部バッファに保持されているチャンネルごとの入力値の数です。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)

出力値

- **Output**：関連する入力データストリームのクレストファクタ。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.4 Crest Factor Plus



各チャンネルの時系列データに対して、シングルチャンネルおよびマルチチャンネルのクレストファクタを計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_CrestFactorPlus

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

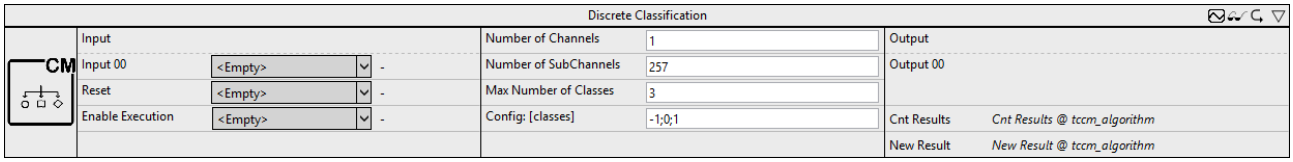
設定オプション

- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Buffer Length**：内部バッファに保持されているチャンネルごとの入力値の数です。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Config: [c1, c2, c3]:** 最大振幅(c1)、実効値(c2)、クレストファクタ(c3)の重みを設定可能。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)

出力値

- **Output**：設定された重みに基づく、関連する入力データストリームの拡張クレストファクタ。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.5 Discrete Classification



設定可能な閾値に基づいてマルチチャンネルデータを分類します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_DiscreteClassification

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

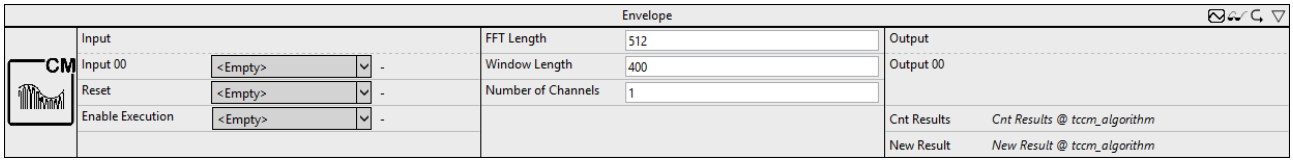
設定オプション

- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Number of SubChannels**：独立したサブチャンネルの数を定義します。この値は、リンクされた入力配列の長さに自動的に調整されます。複数のチャンネルを使用する場合は、全ての長さが一致している必要があります。
- **Max Number of Classes**：設定するクラスの最大数を定義します。値は1以上である必要があります。
- **Config: [classes]**: 設定可能な閾値の定義。

出力値

- **Output**：入力配列と同じ長さの出力配列で、エントリの分類を含みます。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.6 Envelope



時間信号の包絡線を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_Envelope

時間同期での表示および後続処理については、Timeshift [▶ 309]の項を参照してください。

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

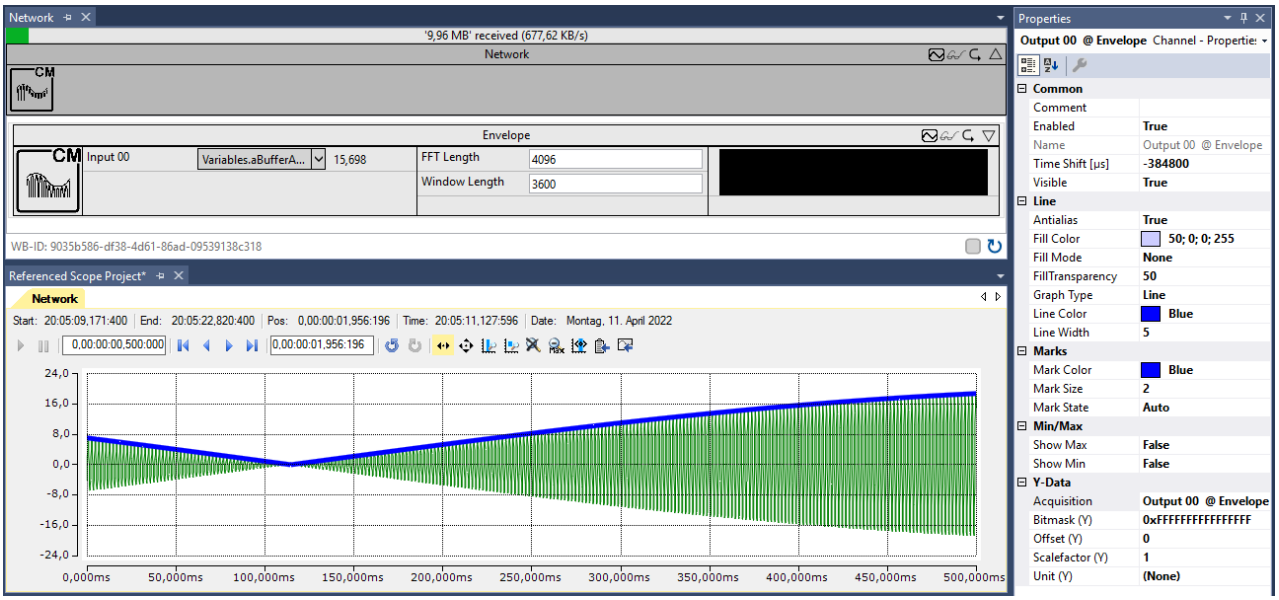
- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。

出力値

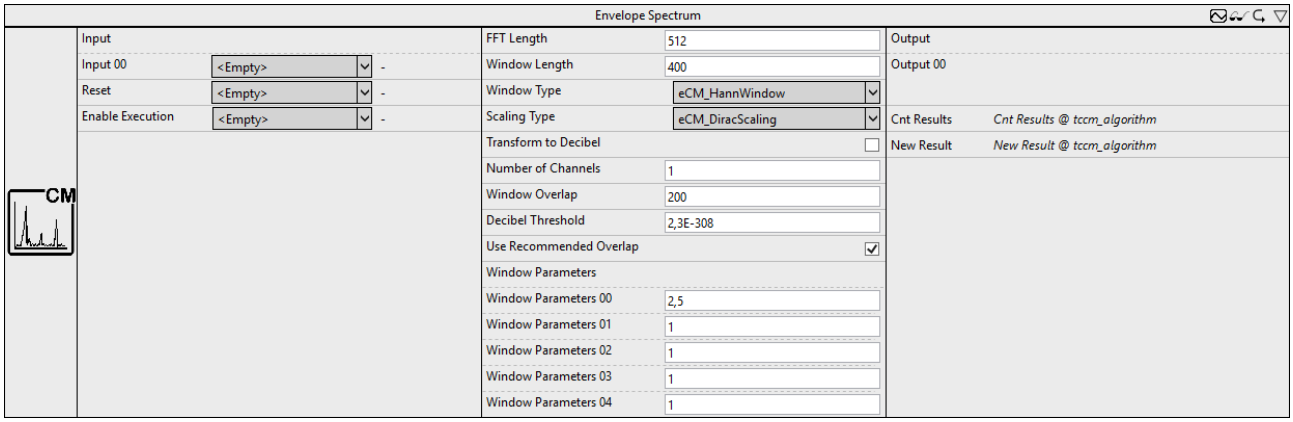
- **Output**：計算された包絡線を含む、ウィンドウ半分の長さの出力配列。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

Time shift

解析にウェルチ法（50%のオーバーラップ）を使用し、合成では解析信号のセグメント化された畳み込みを行うため、有効な出力信号を計算するには、指定された長さnWindowLength/2の完全なデータバッファが4つ必要です。さらに、結果は(nFFTLentgth + nWindowLength)/2 サンプル分、遅延します。これは、出力チャンネルでTime Shift を適切に選択することで修正できます。



6.5.14.2.1.7 Envelope Spectrum



時間信号の包絡線スペクトルを計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_EnvelopeSpectrum

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

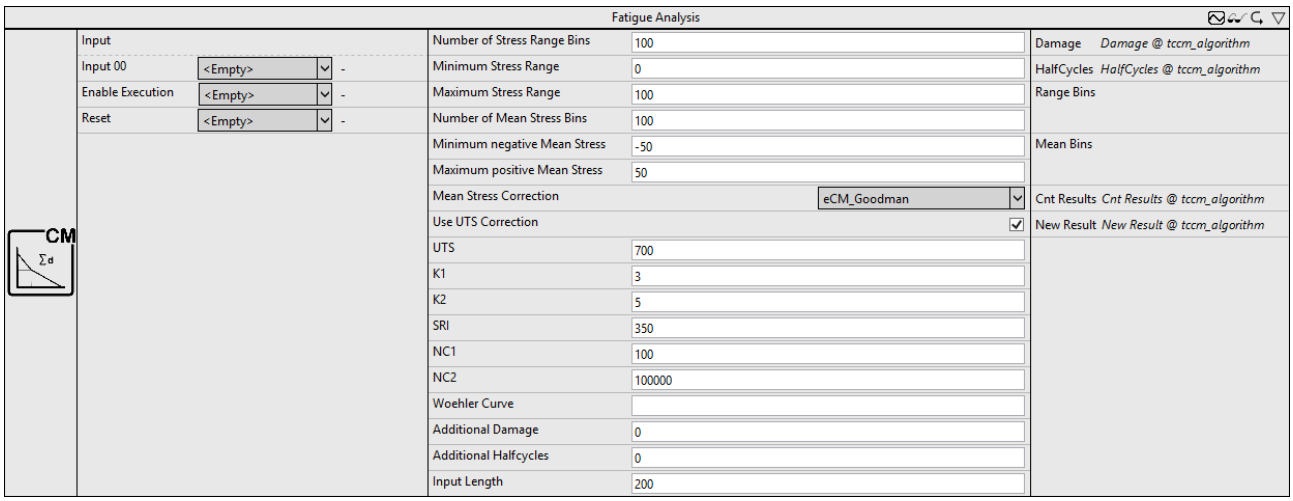
- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Scaling Type**：絶対スケーリングが必要な場合に、使用するスケーリング方式（E_CM_ScalingType型）を選択できるようにします。デフォルト値はeCM_DiracScalingです。スケーリングを選択する際には、信号が決定論的な信号なのか、確率的な部分を持つ広帯域信号なのか、信号の種類を考慮する必要があります。それぞれ異なるスケーリングを必要とします。

- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**：包絡線スペクトルの計算されたスペクトル線を含む出力配列。配列の長さは、N / 2 + 1 になります（FFT長が N の場合）。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.8 Fatigue Analysis



監視対象部品の疲労過程を推定するための疲労寿命解析と損傷計算。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_RainflowCounting、FB_CMA_MeanStressCorrection、FB_CMA_MinersRuleを参照してください。

ウェラー曲線の計算には、F_CM_CalculateWoehlerCurveファンクションが架空の材料パラメータに基づいて実行されます。

より詳細な説明については、Condition Monitoringドキュメントの アプリケーションのコンセプトの章、疲労寿命解析と損傷計算の項を参照してください。

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

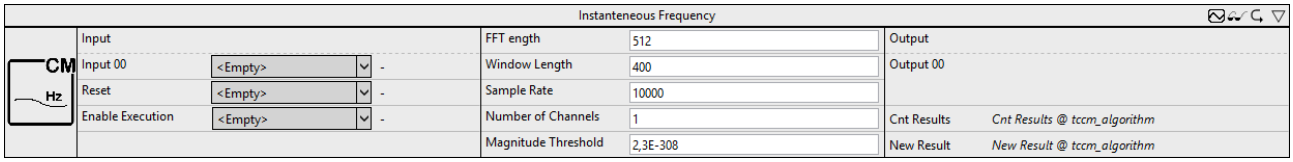
設定オプション

- **Number of Stress Range Bins**：応力レベルのビンの数を定義します。ビンの総数は「nBins + 2」です。この値は1以上である必要があります。各行の最初のビンには、定義された最小値以下の応力レベルが含まれます。同様に、定義された最大値を超える全ての値は、各行の最後のビンに格納されます。
- **Minimum Stress Range**：応力レベルの下限を定義します。この値以下の値は全てビンに蓄積されます。
- **Maximum Stress Range**：応力レベルの上限を定義します。最大値を超える値は、最後のビンでカウントされます。
- **Number of Mean Stress Bins**：平均値のビンの数を定義します。値は1以上である必要があります。Halfcycle Count Matrix（ハーフサイクル・カウント・マトリックス）の最初と最後の列に、2つのビンが追加されます。それぞれ、定義された制限値以下の平均値、または制限値を超える平均値が保持されます。
- **Minimum negative Mean Stress**：平均値の下限を定義します。この値以下の値は全てビンに蓄積されます。
- **Maximum positive Mean Stress**：平均値の上限を定義します。最大値を超える値は、最後のビンでカウントされます。
- **Mean Stress Correction**：使用する補正モードを定義します。グッドマン補正とガーバー補正が可能で、補正をしないという選択肢もあります。
- **Use UTS Correction**：有効の場合、応力マトリックスのUTS補正が実行されます。
- **UTS**：監視の対象となる材料の引張強さを定義します。
- **K1**：N = 1..NC1.の範囲でウェラー曲線の勾配を定義します。
- **K2**：N >= NC2.で始まるウェラー曲線の勾配を定義します。
- **SRI**：「Stress Range Intercept（応力範囲切片）」を定義します。
- **NC1**：UTS補正の移行点を定義します。
- **NC2**：K1とK2の間の移行点を定義します。
- **Woehler Curve**：ウェラー曲線を定義します。
- **Additional Damage**：初期状態から一定の損傷を与えることを定義します。このようにして、設定されたウェラー曲線に対する fDamage と累積損傷の合計から、全体の損傷が算出されます。
- **Additional Halfcycles**：カウントされるハーフサイクルの初期値を定義します。総サイクル数は、nCyclesと現在の入力データ数の合計から計算されます。
- **Input Length**：ブロック処理用の入力データ配列の長さを定義します。

出力値

- **Damage**：定義されたウェラー曲線に対して、計算された疲労損傷を示します。
- **HalfCycles**：カウントされたハーフサイクルの数を示します。
- **Range Bins**：中間応力に沿って集計されたハーフサイクルの合計を含む、「Number of Stress Range Bins（応力範囲のビンの数）」分の長さを持つ出力配列です。
- **Mean Bins**：絶対応力に沿って集計されたハーフサイクルの合計を含む、「Number of Mean Stress Bins（平均応力のビンの数）」分の長さを持つ出力配列です。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.9 Instantaneous Frequency



時間信号の瞬時周波数を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_InstantaneousFrequency

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

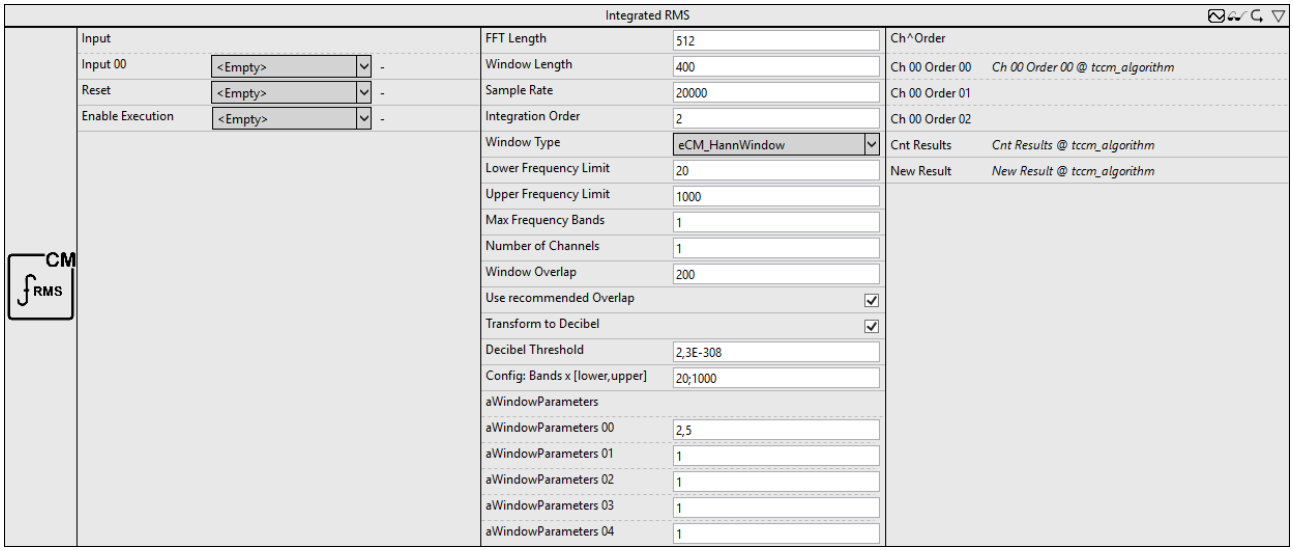
設定オプション

- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Sample Rate**：入力される時間信号のサンプリングレート。この値は、Hz単位での結果のスケーリングに使用されます。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Magnitude Threshold**：瞬時周波数の数値計算における制限値を定義します。

出力値

- **Output**：計算された瞬時周波数を含む、ウィンドウの半分の長さの出力配列です。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Integrated RMS 0



シングルチャンネルおよびマルチチャンネルの実数値時系列データのRMS値（オプションで積分）を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_IntegratedRMS

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

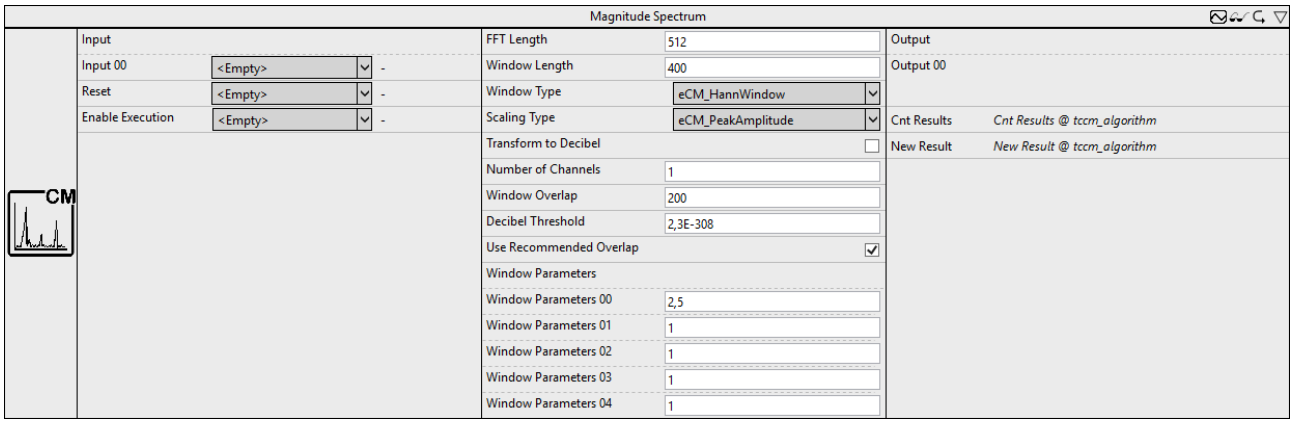
- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Sample Rate**：入力される時間信号のサンプリングレート。この値は、Hz単位での結果のスケーリングに使用されます。

- **Max Integration Order**：積分の最大次数。これは0から2までの整数でなければなりません。チャンネルごとに決定される値の数は（Order+1 ）です。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Lower Frequency Limit**：考慮される周波数間隔の下限。カットオフ周波数の下限値は、サンプリングレートをFFT長で割った値以上に設定する必要があります。
- **Upper Frequency Limit**：考慮される周波数間隔の上限。カットオフ周波数の上限値は、サンプリングレートの半分以下で、下限周波数より大きい値に設定する必要があります。
- **Max Frequency Bands**：この値は、RMS値を計算する周波数帯域の最大数を指定します。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Config: MaxBands x [fmin, fmax]**：設定可能な周波数帯域の下限と上限を定義します。
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**：チャンネルと次数固有の実効値を含む、周波数帯域数に対応した出力配列です。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Magnitude Spectrum 1



入力信号の実数値のマグニチュードスペクトル（振幅スペクトルともいう）を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_MagnitudeSpectrum

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

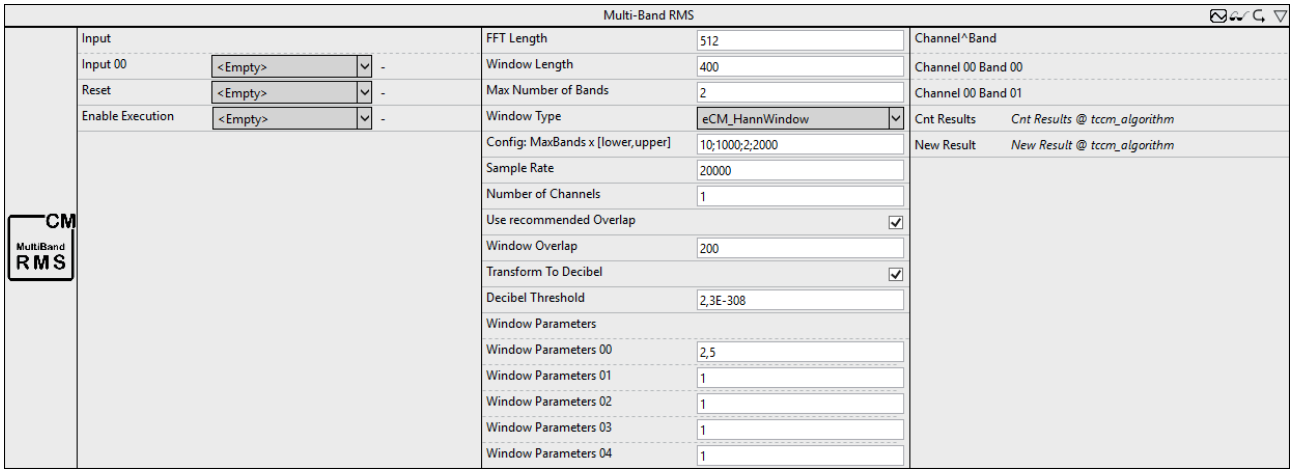
設定オプション

- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Scaling Type**：絶対スケーリングが必要な場合に、使用するスケーリング方式（E_CM_ScalingType型）を選択できるようにします。デフォルト値はeCM_DiracScalingです。スケーリングを選択する際には、信号が決定論的な信号なのか、確率的な部分を持つ広帯域信号なのか、信号の種類を考慮する必要があります。それぞれ異なるスケーリングを必要とします。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**：長さN / 2 + 1 （FFT 長N の場合）の出力配列で、マグニチュードスペクトルの計算済みスペクトル線を含みます。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Multi Band RMS 2



シングルチャンネルおよびマルチチャンネルの実数値時系列データに対して、設定可能な周波数帯域ごとのRMS値を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_MultiBandRMS

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

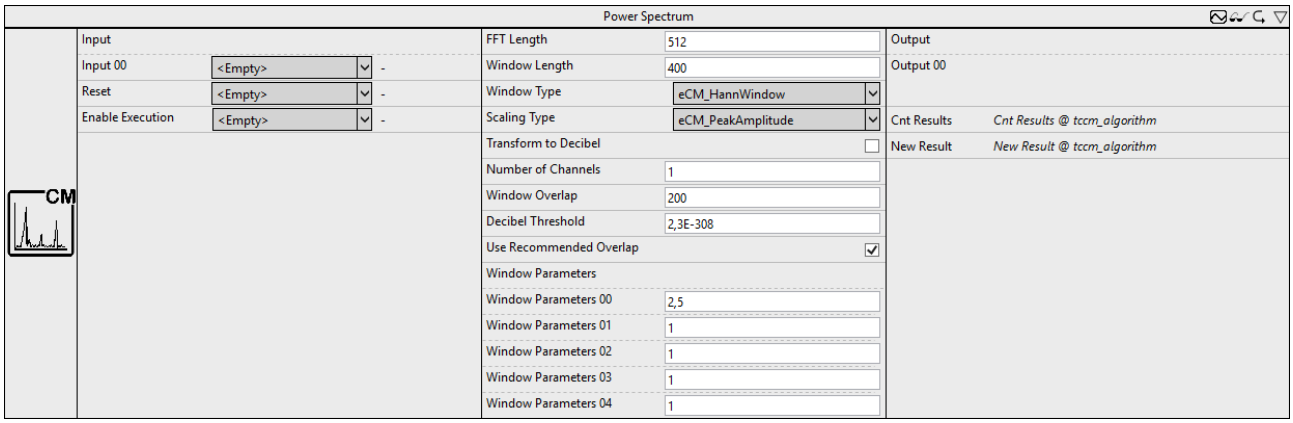
設定オプション

- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Max Number of Bands**：この値は、RMS値を計算する周波数帯域の最大数を指定します。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Config: MaxBands x [fmin, fmax]**：設定可能な周波数帯域の下限と上限を定義します。
- **Sample Rate**：入力される時間信号のサンプリングレート。この値は、Hz単位での結果のスケーリングに使用されます。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**：チャンネルおよび帯域固有の実効値。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Power Spectrum 3



実数値入力信号のパワースペクトルを計算し、オプションでデシベルスケーリングを適用します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_PowerSpectrum

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

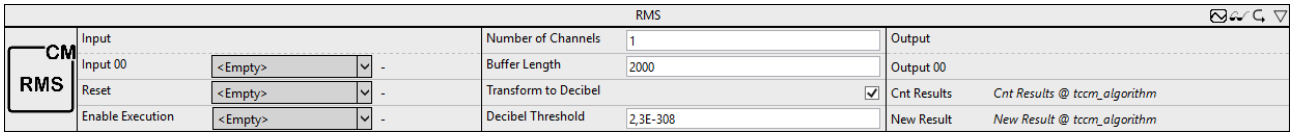
設定オプション

- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Scaling Type**：絶対スケーリングが必要な場合に、使用するスケーリング方式（E_CM_ScalingType型）を選択できるようにします。デフォルト値はeCM_DiracScalingです。スケーリングを選択する際には、信号が決定論的な信号なのか、確率的な部分を持つ広帯域信号なのか、信号の種類を考慮する必要があります。それぞれ異なるスケーリングを必要とします。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**：長さN / 2 + 1 （FFT 長N の場合）の出力配列で、マグニチュードスペクトルの計算済みスペクトル線を含みます。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 RMS 4



シングルチャンネルおよびマルチチャンネルの実数値信号の時間的RMS値を計算します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_CMA_RMS

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

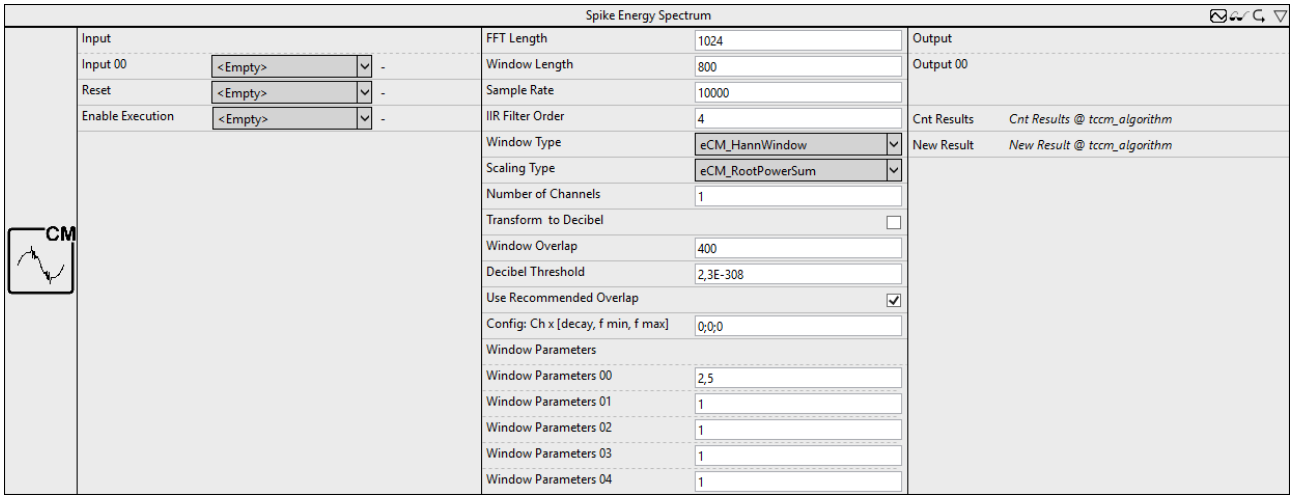
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Buffer Length**：内部バッファに保持されているチャンネルごとの入力値の数です。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。

- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)

出力値

- **Output**：関連する入力データストリームの実効値。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Spike Energy Spectrum 5



高周波信号成分のピークエネルギーを解析します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_SpikeEnergySpectrum

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

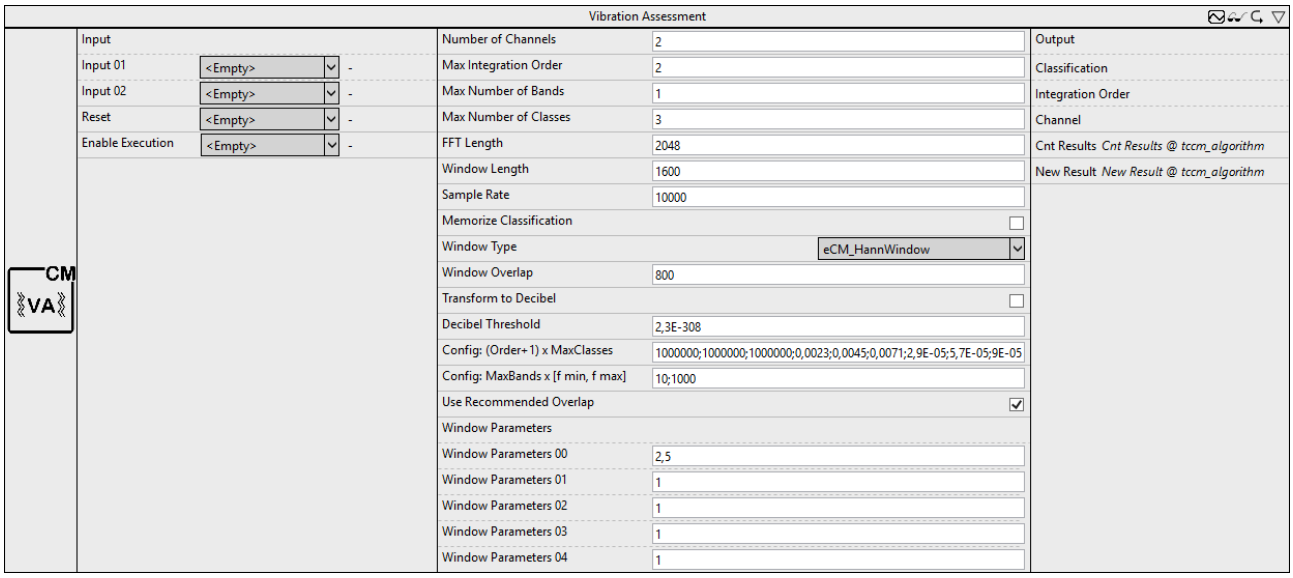
- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Sample Rate**：入力される時間信号のサンプリングレート。この値は、Hz単位での結果のスケーリングに使用されます。
- **IIR Filter Order**：使用するIIRフィルタの次数を定義します。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Scaling Type**：絶対スケーリングが必要な場合に、使用するスケーリング方式（E_CM_ScalingType型）を選択できるようにします。デフォルト値はeCM_DiracScalingです。スケーリングを選択する際には、信号が決定論的な信号なのか、確率的な部分を持つ広帯域信号なのか、信号の種類を考慮する必要があります。それぞれ異なるスケーリングを必要とします。
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。

- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Config: Ch x [decay, fmin, fmax]**: 設定可能なパラメータを定義します：減衰時間（decay）と、対象とする周波数帯域の下限（fmin）および上限（fmax）です。減衰時間は、ピークエネルギーが完全に減衰できるように、最適に選択する必要があります。つまり、減衰時間 > 1 / 誤差周波数となるように設定します。
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**：長さN / 2 + 1 （FFT 長N の場合）の出力配列で、スパイクエネルギースペクトルの計算済みのスペクトル線が含まれます。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Vibration Assessment 6



ISO 10816-3に基づいて、実数値入力信号の振動を評価します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_VibrationAssessment

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

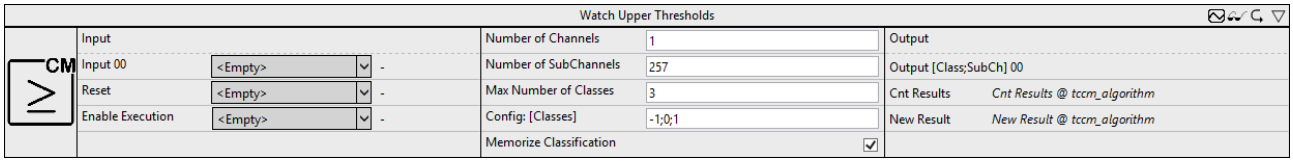
- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Max Integration Order**：積分の最大次数。これは0から2までの整数でなければなりません。チャンネルごとに決定される値の数は（Order+1 ）です。
- **Max Number of Bands**：この値は、RMS値を計算する周波数帯域の最大数を指定します。
- **Max Number of Classes**：設定するクラスの最大数を定義します。値は1以上である必要があります。

- **FFT Length**：FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length**：解析ウィンドウの長さを（サンプル数）です。長さは1より大きく、偶数である必要があります。
- **Sample Rate**：入力される時間信号のサンプリングレート。この値は、Hz単位での結果のスケーリングに使用されます。
- **Memorize Classification**：選択した場合、ファンクションブロックは各ステップについて、最上位カテゴリの番号と対応するチャンネルを再計算します。それ以外の場合は、制限値を超えると、リセットが実行されるか、チャンネルが上位カテゴリに達するまで、結果値が保存されます。
- **WindowType**：使用するウィンドウ関数（E_CM_WindowType型）を定義します。良いデフォルト値はウィンドウタイプeCM_HannWindow です。
- **Window Overlap**：オーバーラップするサンプル数を定義します。これはゼロ以上でなければなりません。
- **Transform to Decibel**: FFTの結果をデシベル単位に変換するかどうかを示すブール値で、変換x → 20 * log10(x)に従って処理されます。
- **Decibel Threshold**：非常に小さな浮動小数点値で、ゼロより大きな値です。この数値より小さい値は、デシベルスケールへの変換前にこの値に置き換えられます。(その目的は、値域エラーの回避です。ゼロの対数は定義されておらず、小さな引数の極限では値が負の無限大に発散します。数字のゼロの引数、arg(0)も同様です。最小値は2.3e-308です。)
- **Config: (Order+1) x MaxClasses**：積分次数に対する設定可能な閾値を定義します。
- **Config: MaxBands x [fmin, fmax]**：設定可能な周波数帯域の下限と上限を定義します。
- **Use Recommended Overlap**：選択した場合、内部的に推奨オーバーラップが計算されます（F_CM_CalculateRecommendedOverlapを参照）。
- **Window Parameters**： 選択したウィンドウ関数のフリーパラメータが含まれます。eCM_KaiserWindowを使用する場合、最初のエントリはパラメータβを定義します。eCM_FlatTopWindowを使用する場合、全てのパラメータが使用されます。ウィンドウ関数を参照してください。

出力値

- **Output**: 結果は、各周波数帯域について、計算された最高分類（範囲 -1..'Max Number of Classes'）、関連する積分次数（範囲 0..'Max Integration Order'）、チャンネル（範囲 1..'Number of Channels'）の3つの値を含む1次元配列が得られます。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

6.5.14.2.1.1 Watch Upper Thresholds 7



マルチチャンネルデータの設定可能な閾値監視を実行します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_CMA_WatchUpperThresholds

TwinCAT 3 Condition Monitoring PLCライブラリのドキュメントは、こちらをご覧ください：概要

設定オプション

- **Number of Channels**：独立したチャンネルの数を定義します。これはゼロより大きくなければなりません。
- **Number of SubChannels**：独立したサブチャンネルの数を定義します。この値は、リンクされた入力配列の長さに自動的に調整されます。複数のチャンネルを使用する場合は、全ての長さが一致している必要があります。

- **Max Number of Classes**：設定するクラスの最大数を定義します。値は1以上である必要があります。
- **Config: [classes]**: 設定可能な閾値の定義。
- **Memorize Classification**：選択した場合、ファンクションブロックは各ステップについて、最上位カテゴリの番号と対応するチャンネルを再計算します。それ以外の場合は、制限値を超えると、リセットが実行されるか、チャンネルが上位カテゴリに達するまで、結果値が保存されます。

出力値

- **Output**：各入力データストリームに対して、長さ2の出力配列。タプルの値は、分類結果およびそのクラスが識別されたサブチャンネルを示します。
- **Cnt Results**：計算される出力配列の数を指定します。
- **New Result**：現在のサイクルで新しい結果が得られたかどうかを示すブール値。

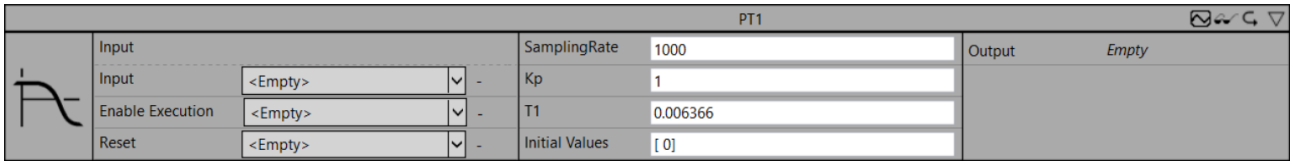
6.5.14.2.2 Filter

TwinCAT Filterライブラリは、デジタルフィルタを実装するためのアルゴリズムを提供します。

アルゴリズムの概要

アルゴリズム	説明	PLCファンクションブロック
PT1 [▶ 320]	一次遅れ要素（PT1要素）を実装します。	FB_FTR_PT1
IIRCoeff [▶ 321]	フィルタ係数を用いてIIR（無限インパルス応答）フィルタを実装します。	FB_FTR_IIRCoeff
IIRSpec [▶ 321]	フィルタ仕様を用いてIIR（無限インパルス応答）フィルタを実装します。	FB_FTR_IIRSpec
PT2 [▶ 322]	2次遅れ要素（PT2要素）を実装します。	FB_FTR_PT2
PT3 [▶ 322]	3次遅れ要素（PT3要素）を実装します。	FB_FTR_PT3
PTn [▶ 323]	等しい時定数を持つn次の遅延要素を実装します。	FB_FTR_PTn
IIRSos [▶ 323]	フィルタ係数を用いてIIR（無限インパルス応答）フィルタを実装します。	FB_FTR_IIRSos
Notch [▶ 324]	狭帯域バンドストップフィルタを実装します。	FB_FTR_Notch
LeadLag [▶ 324]	一次位相補正要素を実装します。	FB_FTR_LeadLag
PT2oscillation [▶ 325]	二次振動遅延要素を実装します。	FB_FTR_PT2oscillation
PTt [▶ 325]	デッドタイム要素（PTt要素）を実装します。	FB_FTR_PTt
Median [▶ 326]	メディアンフィルタを実装します。	FB_FTR_Median
ActualValue [▶ 326]	計測した入力値の妥当性チェックとフィルタリングを実装します。	FB_FTR_ActualValue
Gaussian [▶ 326]	ガウシアンフィルタを実装します。	FB_FTR_Gaussian

6.5.14.2.2.1 PT1



一次遅れ要素（PT1要素）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_PT1

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

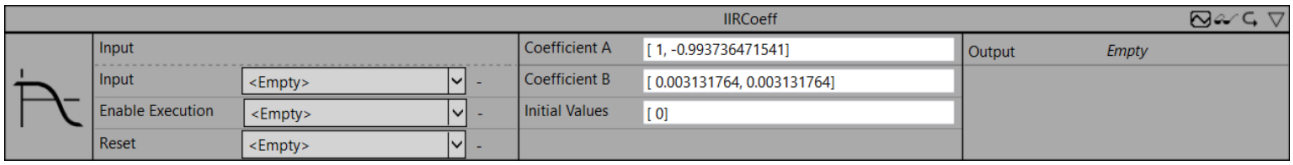
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Kp**：ゲイン係数。ゼロより大きい値を設定します。
- **T1**: 時定数$_{T1}$(単位：秒)。ゼロより大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.2 IIRCoeff



無限インパルス応答フィルタ（IIR）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_IIRCoeff

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

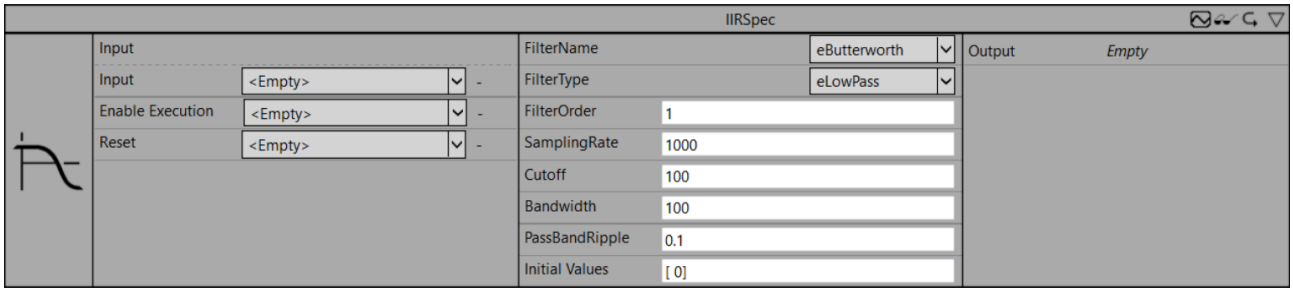
パラメータ設定

- **Coefficient A**：係数 a_k（分母）[a_0, a_1, a_2,..., a_N]は自由に選択できます。
- **Coefficient B**：係数 b_k（分子）[b_0, b_1, b_2,..., b_M]は自由に選択できます。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.3 IIRSpec



無限インパルス応答フィルタ（IIR）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_IIRSpec

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

パラメータ設定

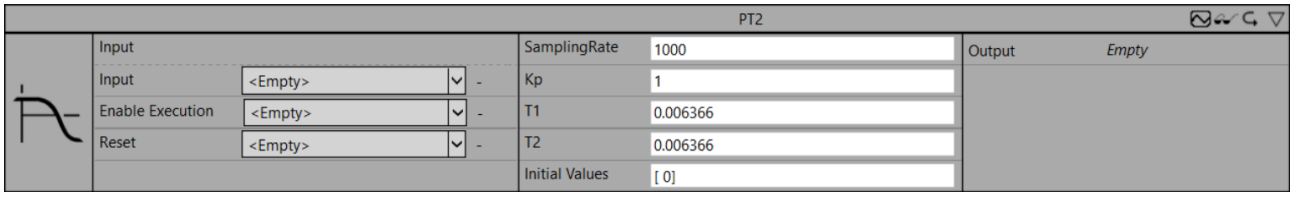
- **FilterName**： フィルタの実装方式を示します（バターワース、チェビシェフ など）。
- **FilterType**：フィルタの種類を示します（ハイパス、ローパス、バンドパス、バンドストップ）。
- **FilterOrder**：フィルタの次数（ハイパスとローパスは最大20、バンドパスとバンドストップは最大10）。
- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。

- **Cutoff**：カットオフ周波数（単位：Hz）。0より大きくSamplingRate/2より小さい値を設定します。
- **Bandwidth**：バンドパス および バンドストップに関する帯域幅（単位：Hz）。
- **PassBandRipple**:フィルタの通過帯域における振幅応答のリップル（単位：dB）。0より大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.4 PT2



2次遅れ要素（PT2要素）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_PT2

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

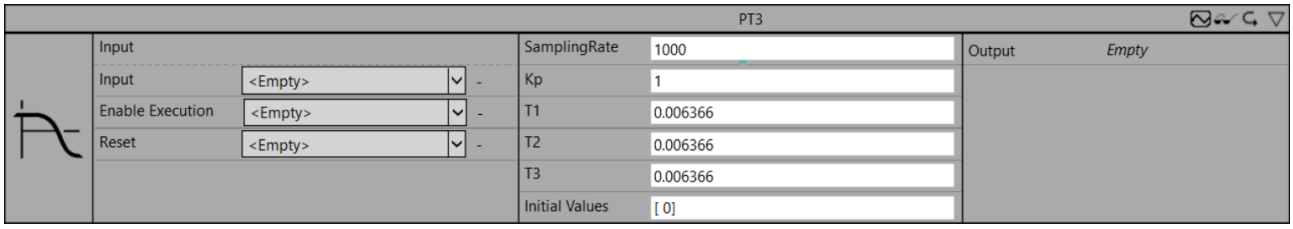
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Kp**：ゲイン係数。ゼロより大きい値を設定します。
- **T1**: 時定数$_{T1}$(単位：秒)。ゼロより大きい値を設定します。
- **T2**: 時定数$_{T2}$(単位：秒)。ゼロより大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.5 PT3



3次遅れ要素（PT3要素）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_PT3

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

パラメータ設定

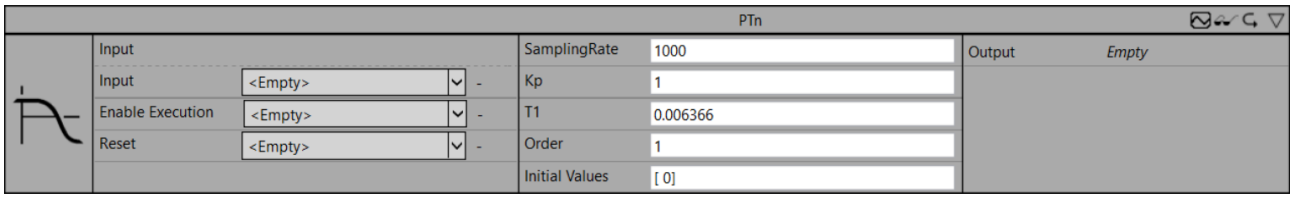
- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Kp**：ゲイン係数。ゼロより大きい値を設定します。
- **T1**: 時定数$_{T1}$(単位：秒)。ゼロより大きい値を設定します。
- **T2**: 時定数$_{T2}$(単位：秒)。ゼロより大きい値を設定します。

- **T3**: 時定数$_{T3}$(単位：秒)。ゼロより大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.6 PTn



等しい時定数を持つn次の遅延要素を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_PTn

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

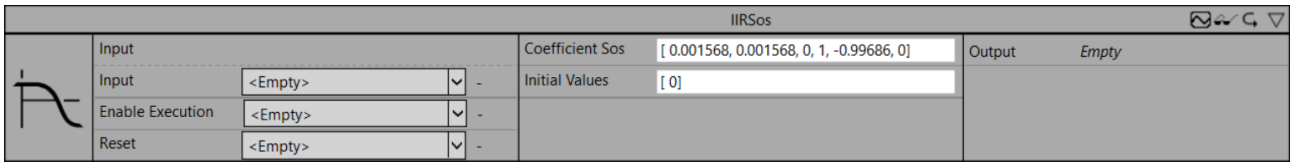
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Kp**：ゲイン係数。ゼロより大きい値を設定します。
- **T1**: 時定数$_{T1}$(単位：秒)。ゼロより大きい値を設定します。
- **Order**：フィルタの次数（1～10）。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.7 IIRSos



無限インパルス応答フィルタ（IIR）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_IIRSos.

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

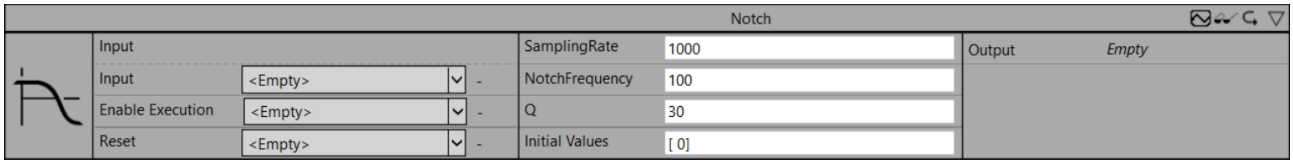
パラメータ設定

- **Coefficient Sos**：係数[b_{01},b_{11},$b_{,21}$ a_{01},a_{11},a_{21}, b_{02},b_{12},b_{22}, a_{02},a_{12},a_{22},…,b_{0M},b_{1M},b_{2M}, a_{0M},a_{1M},a_{2M}]を自由に選択可能です。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.8 Notch



狭帯域バンドストップフィルタを実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_Notch

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

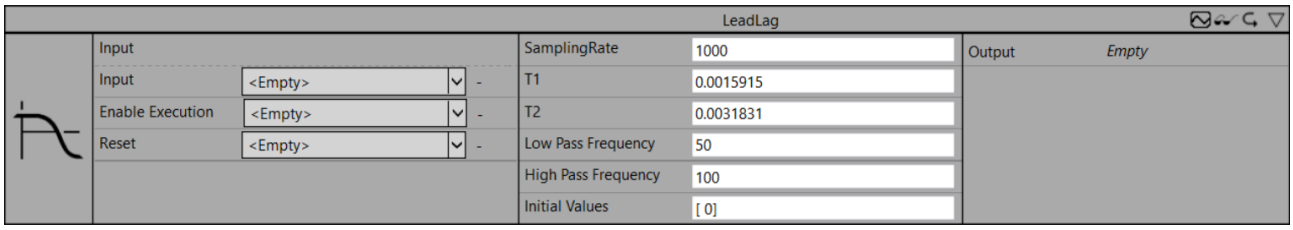
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **NotchFrequency**：ノッチ周波数（単位：Hz）。0より大きくSamplingRate/2より小さい値を設定します。
- **Q**: Qファクタ = ノッチ周波数/帯域幅。ゼロより大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.9 LeadLag



一次位相補正要素を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_FTR_LeadLag.

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

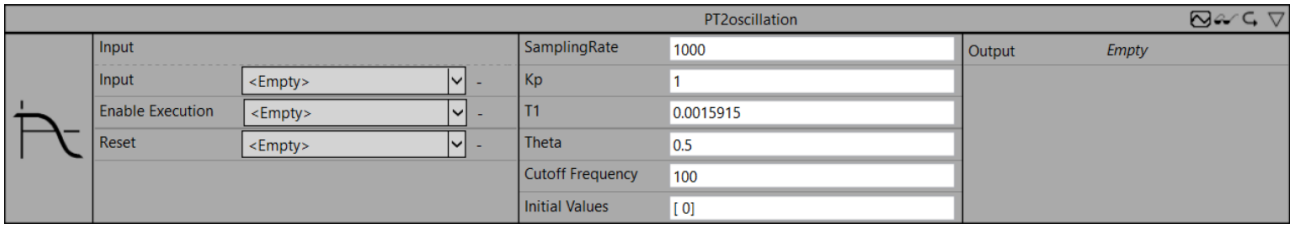
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **T1**: 時定数$_{T1}$(単位：秒)。ゼロより大きい値を設定します。
- **T2**: 時定数$_{T2}$(単位：秒)。ゼロより大きい値を設定します。
- **Low Pass Frequency（オプション）**：ローパス周波数（単位：Hz）。0より大きくSamplingRate/2より小さい値を設定します。
- **High Pass Frequency（オプション）**：ハイパス周波数（単位：Hz）。0より大きくSamplingRate/2より小さい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.1 PT2oscillation 0



二次振動遅延要素を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_FTR_PT2oscillation

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

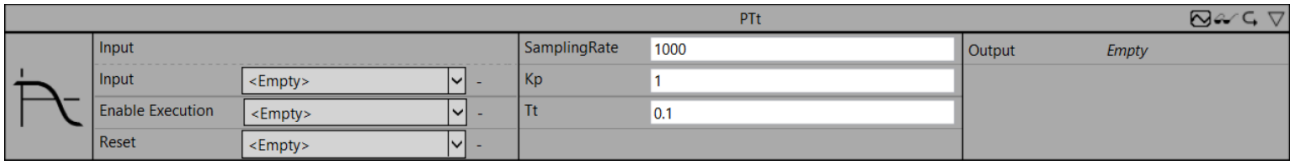
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Kp**：ゲイン係数。ゼロより大きい値を設定します。
- **T1**: 時定数$_{T1}$(単位：秒)。ゼロより大きい値を設定します。
- **Theta**：減衰係数。ゼロより大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.1 PTt 1



デッドタイム要素（PTt要素）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_PTt

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

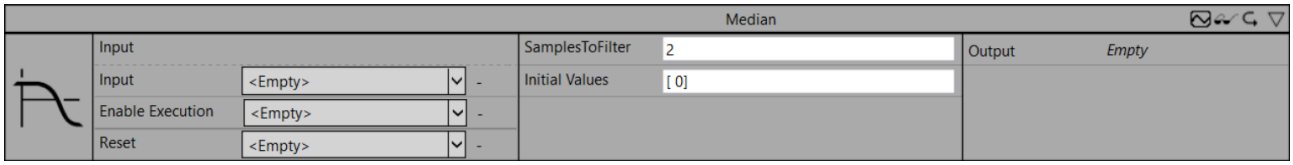
パラメータ設定

- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Kp**：ゲイン係数。ゼロより大きい値を設定します。
- **Tt**：デッドタイム$_{Tt}$（単位：秒）。ゼロより大きい値を設定します。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.1 Median 2



メディアンフィルタを実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_Median

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

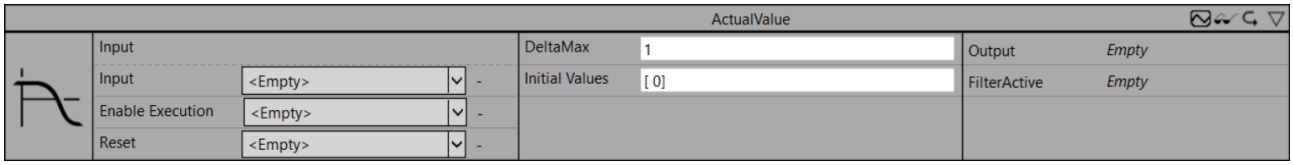
パラメータ設定

- **SamplesToFilter**：スライディング平均値を計算するためのサンプル数（一般に、ウィンドウサイズと呼ばれます）。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.2.1 ActualValue 3



計測した入力値の妥当性チェックとフィルタリングを実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：
FB_FTR_ActualValue

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

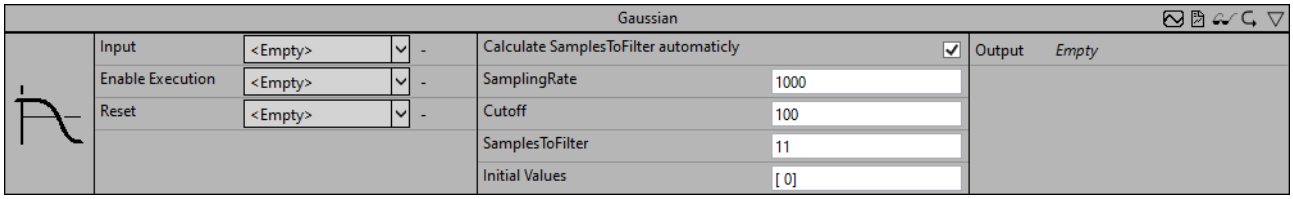
パラメータ設定

- **DeltaMax**：連続する2つの入力値の差の最大値（ゼロ以上）。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。
- **FilterActive**：入力信号のどの要素が変更されたかを示します。

6.5.14.2.2.1 Gaussian 4



無限インパルス応答フィルタ（IIR）を実装します。

対応する PLC ファンクションブロックのドキュメントは、こちらを参照してください：FB_FTR_IIRSpec

TwinCAT 3 Filter PLCライブラリのドキュメントは、こちらを参照してください：概要

パラメータ設定

- **Calculate SamplesToFilter automatically**：SamplesToFilterを自動的に計算するかどうかを示すブール値。
- **SamplingRate**：サンプリングレートf_s（単位：Hz）ゼロより大きい値を設定します。
- **Cutoff**：カットオフ周波数（単位：Hz）。0より大きくSamplingRate/2より小さい値を設定します。
- **SamplesToFilter**：スライディング平均値を計算するためのサンプル数（一般に、ウィンドウサイズと呼ばれます）。
- **Initial Values（オプション）**：初期値はフィルタの内部状態を定義します。

出力値

- **Output**：操作された出力信号。

6.5.14.2.3 電力モニタリング

ベッコフのI/Oポートフォリオは、送電網の解析と監視を実行するための様々なターミナルがを提供します。単相または三相グリッドでは、EL3403 や EL3413 などの電力計測ターミナルを使用して、電流と電圧の実効値、および有効電力、無効電力、皮相電力を直接、計測できます。EL3773・EL3783電源モニタリングターミナルは、電流と電圧のローデータを記録するために使用することも可能で、グリッドイベントをより正確に検出し、コントローラで様々な値を計算できます。

TwinCAT 3 Power Monitoringファンクションは、電源モニタリングターミナルから提供される電流・電圧のローデータを評価するためのPLCライブラリです。

このライブラリには、電流、電圧、電力の実効値を計算するファンクションブロックが用意されています。これらは瞬時値または平均値として出力できます。また、ファンクションブロックは、最小値と最大値も出力できます。周波数や周波数スペクトルを解析することで、たとえばグリッド内の高調波や、それに伴う負荷を全高調波歪み（THD）として把握できます。さらに、このライブラリには、周波数や回転界を検出するためのファンクションブロックも用意されています。

一般的なアルゴリズム

アルゴリズム	説明	PLCファンクションブロック
スケーリング [▶ 328]	電流と電圧の入力データをスケーリングします。	
Scaling EL3773 [▶ 329]	EL3773 EtherCATターミナルからの電流と電圧の入力データをスケーリングします。	
Scaling EL3783 [▶ 329]	EL3783 EtherCATターミナルからの電流と電圧の入力データをスケーリングします。	

単相アルゴリズム

アルゴリズム	説明	PLCファンクションブロック
Frequency Period 1Ph [▶ 330]	入力信号の基本周波数を計算します。	
Basic Values Period 1Ph [▶ 331]	電流および電圧の時系列信号に対する解析値を計算します。	
Power Values Period 1Ph [▶ 332]	信号周期に基づいて、接続機器の特性電力値を計算します。	
Harmonics Period 1Ph [▶ 333]	信号周期に基づいて、電流と電圧の高調波を計算します。	

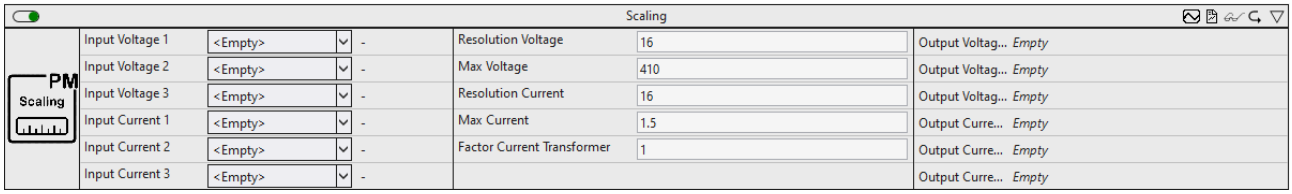
アルゴリズム	説明	PLCファンクションブロック
Spectrum 1Ph [▶ 334]	電流値と電圧値のマグニチュードスペクトルを計算します。	
Spectrum Quantiles 1Ph [▶ 335]	電流値と電圧値のマグニチュードスペクトルを計算します。加えて、スペクトル分布のp分位数も計算できます。	
Power Values 1Ph [▶ 336]	周波数範囲での演算結果に基づいて、接続機器の電力特性を計算します。	
Harmonics 1Ph [▶ 337]	電流と電圧の高調波を計算します。	

三相アルゴリズム

アルゴリズム	説明	PLCファンクションブロック
Frequency Period 3Ph [▶ 339]	入力信号の基本周波数を計算します。	
Basic Values Period 3Ph [▶ 340]	電流および電圧の時系列信号に対する解析値を計算します。	
Power Values Period 3Ph [▶ 341]	信号周期に基づいて、接続機器の特性電力値を計算します。	
Harmonics Period 3Ph [▶ 342]	信号周期に基づいて、電流と電圧の高調波を計算します。	
Spectrum 3Ph [▶ 343]	電流値と電圧値のマグニチュードスペクトルを計算します。	
Spectrum Quantiles 3Ph [▶ 344]	電流値と電圧値のマグニチュードスペクトルを計算します。加えて、スペクトル分布のp分位数も計算できます。	
Power Values 3Ph [▶ 345]	周波数範囲での演算結果に基づいて、接続機器の電力特性を計算します。	
Harmonics 3Ph [▶ 347]	電流と電圧の高調波を計算します。	

6.5.14.2.3.1 全般

6.5.14.2.3.1. スケーリング 1



EtherCATターミナルなどからの電流および電圧の入力データをスケーリングします。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Scalingのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Resolution Voltage :** 電圧の分解能（単位：ビット）。値の範囲：1-32。
- **Max Voltage :** 入力電圧の最大振幅を指定します。
- **Resolution Current:**電流の分解能（単位：ビット）。値の範囲：1-32。

- **Max Current :** 入力電流の最大振幅を指定します。
- **Factor Current Transformer:**電流トランス（変流器）の係数。これは、入力電流を出力電流で割ることで算出されます。

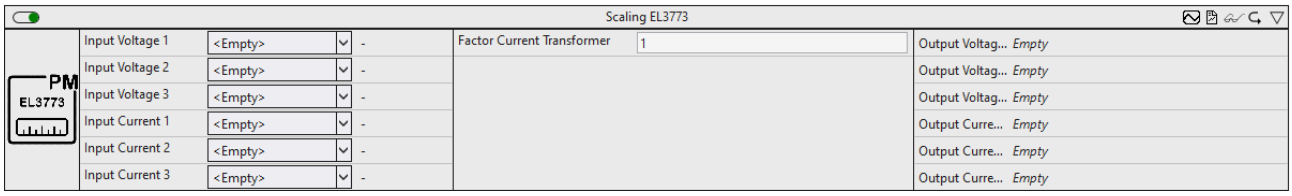
入力値

- **Input Voltage 1 ... Input Voltage 3:**各相電圧の生データの値。
- **Input Current 1 ... Input Current 3:**各相電流の生データの値。

出力値

- **Output Voltage 1 ... Input Voltage 3:**各相電圧値をスケーリングした値。
- **Output Current 1 ... Input Current 3:**各相電流値をスケーリングした値。

6.5.14.2.3.1. Scaling EL3773 2



EL3773 EtherCATターミナルからの電流と電圧の入力データをスケーリングします。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Scaling_EL3773のドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Factor Current Transformer:**電流トランス（変流器）の係数。これは、入力電流を出力電流で割ることで算出されます。

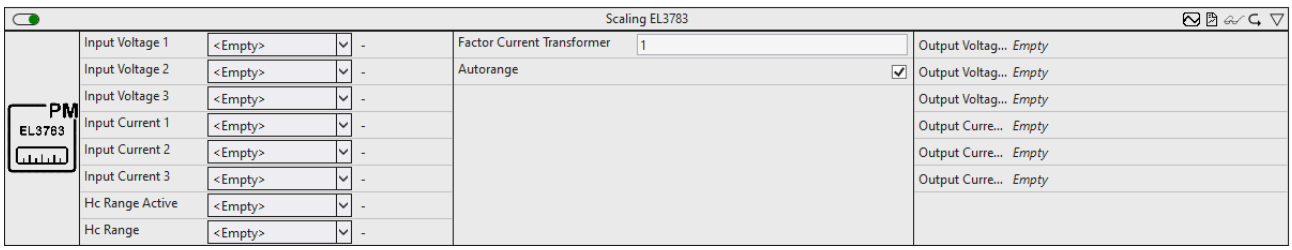
入力値

- **Input Voltage 1 ... Input Voltage 3:**各相電圧の生データの値。
- **Input Current 1 ... Input Current 3:**各相電流の生データの値。

出力値

- **Output Voltage 1 ... Input Voltage 3:**各相電圧値をスケーリングした値。
- **Output Current 1 ... Input Current 3:**各相電流値をスケーリングした値。

6.5.14.2.3.1. Scaling EL3783 3



EL3783 EtherCATターミナルからの電流と電圧の入力データをスケーリングします。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Scaling_EL3783のドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Factor Current Transformer:**電流トランス（変流器）の係数。これは、入力電流を出力電流で割ることで算出されます。
- **Autorange:** EtherCATターミナルで自動範囲選択を有効にします。

入力値

- **Input Voltage 1 ... Input Voltage 3:**各相電圧の生データの値。
- **Input Current 1 ... Input Current 3:**各相電流の生データの値。
- **Hc Range Active:** ターミナルでAuto-Range（自動範囲）モードが有効な場合、*TRUE*。
- **Hc Range:** Auto-Rangeモード時のEL3783の電流計測範囲。
- **Use 5A range（オプション）：** この値が*TRUE* の場合、EL3783 の 5A 計測範囲が適用されます。*FALSE*の場合、1Aの計測範囲が適用されます。Auto-Rangeモードとの併用はできません。

出力値

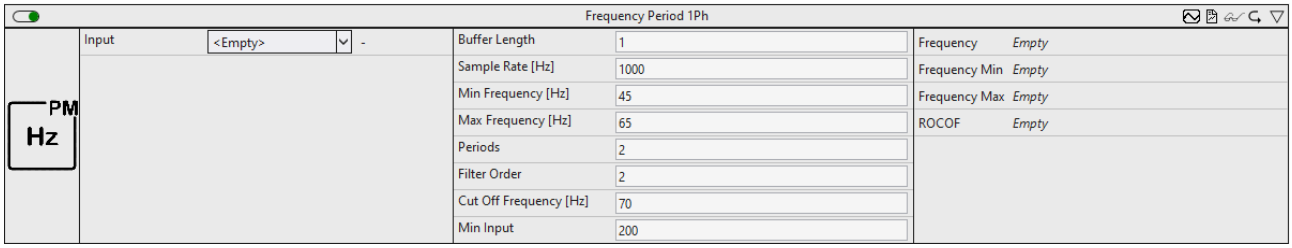
- **Output Voltage 1 ... Input Voltage 3:**各相電圧値をスケーリングした値。
- **Output Current 1 ... Input Current 3:**各相電流値をスケーリングした値。

6.5.14.2.3.2 単相

6.5.14.2.3.2. 信号周期に基づく

1

6.5.14.2.3.2.1 Frequency Period 1Ph .1



*Frequency Period*モジュールは、入力信号の基本周波数を計算します。そのため、信号はまずバターワース型ローパスフィルタでフィルタリングされます。その後、フィルタ後の信号からゼロクロス点を検出し、これらの間隔に基づいて信号の周波数を計算します。結果は、設定に応じて、1つまたは複数の区間を参照します。統計結果は、ランタイム全体、または最後に統計がリセットされた時点からのデータを参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック
FB_PMA_Frequency_Period_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。

- **Min Frequency [Hz]**：計測対象として想定される最小周波数。
- **Max Frequency [Hz]**：計測対象として想定される最大周波数。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。
- **Filter Order:**ローパスフィルタの次数。フィルタの安定性を考慮して設定する必要があります。10次までの値が許可されます。
- **Cut Off Frequency [Hz]:**ローパスフィルタのカットオフ周波数。
- **Min Input:**1周期の最小入力値（RMS）。このパラメータは、下限を下回る入力値の計算を防止します。
- **ROCOF Avg Window Order:**ROCOF（周波数変化率）の計算に用いるスライディング平均のウィンドウサイズ。

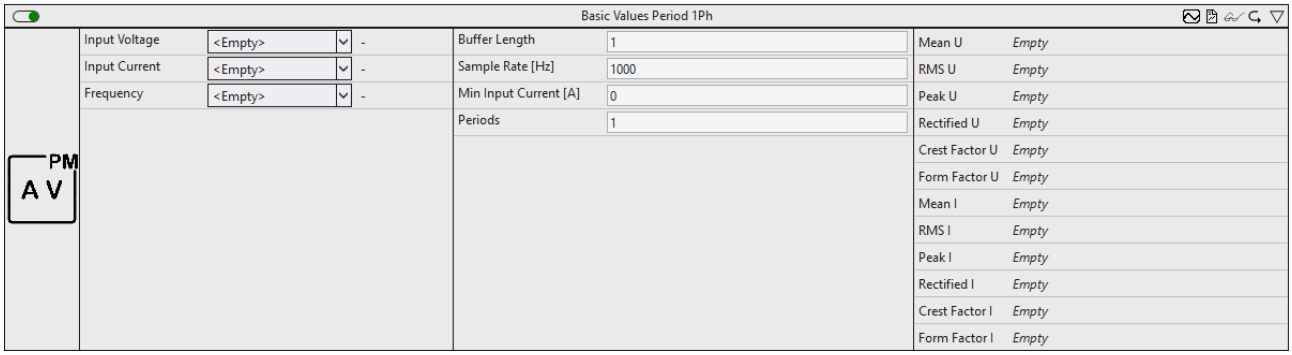
入力値

- **Input :** 電流または電圧の入力値。
- **Reset Statistics（オプション）:** *TRUE* を設定すると、出力の最小値と最大値がリセットされます。

出力値

- **Frequency:** 2回以上のゼロクロスから算出された周波数。
- **Frequency Min:** 計算された周波数の最小値。*Reset Statistics*入力でリセット可能です。
- **Frequency Max :** 計算された周波数の最大値。*Reset Statistics*入力でリセット可能です。
- **ROCOF :** 周波数変化率（Rate of change of frequency）。

6.5.14.2.3.2.1 Basic Values Period 1Ph .2



*Basic Values Period*モジュールは、電流と電圧の時系列信号に対して、解析指標を算出します。これらには、電流と電圧の平均値、実効値、ピーク値、整流値、クレストファクタ、フォームファクタが含まれます。結果は、設定した信号周期数を基準としています。周期の長さは、周期開始時点で入力された周波数に対応します。オプションの統計結果は、ランタイム全体または統計結果が最後にリセットされた時間を参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_BasicValues_Period_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。

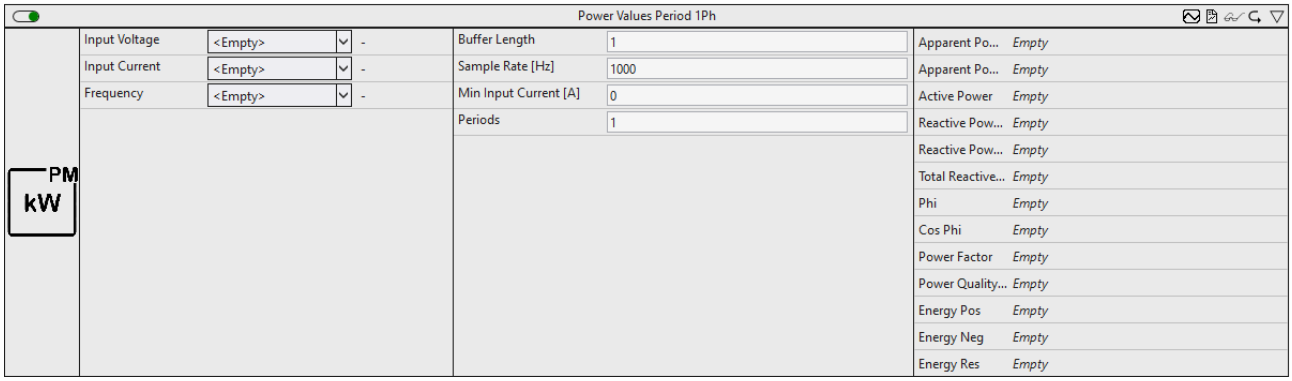
入力値

- **Frequency:** 入力信号の現在の周波数。周期開始時に、周期長を決定するために使用されます。*Frequency Period1 Ch*モジュールの出力が使用できます。
- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション）：** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Mean U:** n周期の平均電圧値。
- **RMS U：** n周期の電圧実効値。
- **Peak U：** n周期の電圧ピーク値。
- **Rectified U:** n周期の整流電圧値。
- **Crest Factor U：** 電圧のクレストファクタ（ピーク値/実効値）。
- **Form Factor U：** 電圧のフォームファクタ（実効値／補正値）。
- **Mean I:** n周期の電流の平均値。
- **RMS I：** n周期の電流の実効値。
- **Peak I：** n周期の電流のピーク値。
- **Rectified I：** n周期の電流の整流値。
- **Crest Factor I：** 電流のクレストファクタ（ピーク値/実効値）。
- **Form Factor I：** 電流のフォームファクタ（実効値／補正値）。

6.5.14.2.3.2.1 Power Values Period 1Ph .3



*Power Values Period*モジュールは、接続機器の電力値を計算します。これには基本波成分と位相シフト角が含まれます。このため、時間範囲内の信号曲線に加えて、入力信号の最初の高調波のみが計算に使用されます。これらのアルゴリズムの特長は、計算処理の動的な応答性が高い点にあります。結果は、設定した信号周期数を基準としています。周期の長さは、周期開始時点で入力された周波数に対応します。統計結果は、ランタイム全体、または最後に統計がリセットされた時点からのデータを参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_PowerValues_Period_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz]：**入力信号のサンプル・レート（サンプル/秒）。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。

- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。

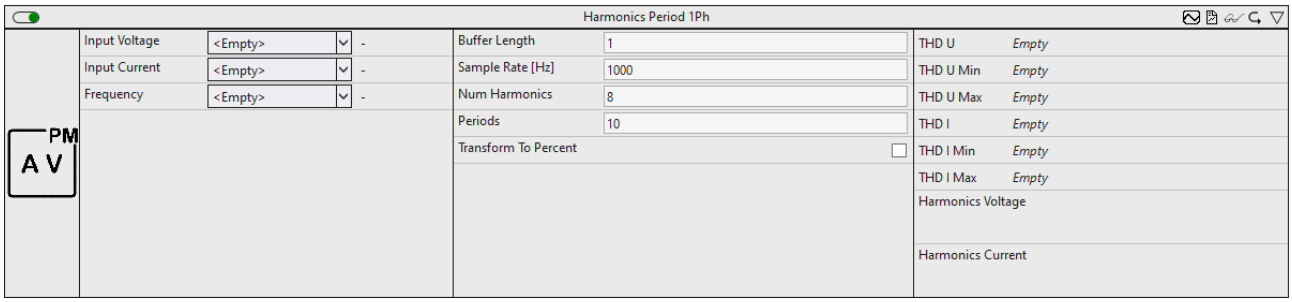
入力値

- **Frequency:** 入力信号の現在の周波数。周期開始時に、周期長を決定するために使用されます。*Frequency Period1 Ch*モジュールの出力が使用できます。
- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション）：** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Apparent Power:** 総皮相電力。
- **Apparent Power 1：** 基本皮相電力。
- **Active Power:**有効電力。
- **Reactive Power d:** 歪み無効電力。
- **Reactive Power 1:** 基本波無効電力。
- **Total Reactive Power:** 総無効電力。
- **Phi:** 位相シフト角。
- **Cos Phi：** CosPhi（有効電力/基本波皮相電力）。
- **Power Factor:** 力率（有効電力/総皮相電力）。
- **Power Quality Factor:** 電源品質係数。周波数、電圧の実効値、および電圧のTHD（全高調波歪み）を考慮し、0～1の範囲で電源の品質を簡易的に表す指標です。
- **Energy Pos:** 正方向の電力量。
- **Energy Neg：** 負方向の電力量。
- **Energy Res:** 結果の電力量。

6.5.14.2.3.2.1 Harmonics Period 1Ph .4



*Harmonics Period*モジュールは、電流と電圧の高調波を計算します。さらに、入力値のTHDは高調波から計算されます。*Harmonics*モジュールとは対照的に、結果は設定可能な信号周期数に基づいています。周期の長さは、周期開始時点で入力された周波数に対応します。統計結果は、ランタイム全体、または最後に統計がリセットされた時点からのデータを参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_Harmonics_Period_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz]：**入力信号のサンプル・レート（サンプル/秒）。

- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。
- **Num Harmonics :** 計算する高調波の数。
- **Transform To Percent:** 結果をパーセント値で出力するかどうかを指定するブール値。第一高調波は100%に相当します。

入力値

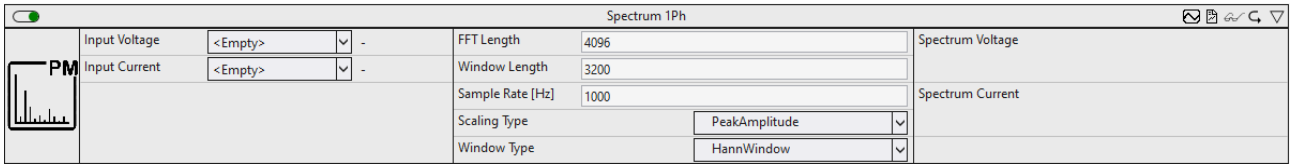
- **Frequency:** 入力信号の現在の周波数。周期開始時に、周期長を決定するために使用されます。*Frequency Period1 Ch*モジュールの出力が使用できます。
- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション）:** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **THD U :** 電圧のTHD。出力はパーセント値で表示されます。
- **THD U Min :** *THD U*値の最小値。*Reset Statistics*入力でリセット可能です。
- **THD U Max :** *THD U*値の最大値。*Reset Statistics*入力でリセット可能です。
- **THD I :** 電流のTHD。出力はパーセント値で表示されます。
- **THD I Min :** *THD I*の最小値。*Reset Statistics*入力でリセット可能です。
- **THD I Max :** *THD I*の最大値。*Reset Statistics*入力でリセット可能です。
- **Harmonics Voltage :** 電圧の高調波。
- **Harmonics Current :** 電流の高調波。

6.5.14.2.3.2. 周波数範囲に基づく 2

6.5.14.2.3.2.2 Spectrum 1Ph .1



*Spectrum*モジュールは、電流値と電圧値のマグニチュードスペクトルを計算します。これらは周波数範囲の入力信号を解析するのに適しています。入力バッファのサイズはウィンドウ長の半分です。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Spectrum_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **FFT Length :** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length :** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Scaling Type :** 絶対スケーリングが必要な場合に使用するスケーリング方式を選択します。を参照してください。

- **Window Type：** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Transform To Decibel（オプション）：** FFTの結果をデシベル尺度（x → 20 × log10(x)）に変換するかどうかを示すブール値。
- **Decibel Threshold（オプション）：** 非常に小さな浮動小数点値で、ゼロより大きい値。ゼロの対数は数学的に定義されていないため、デシベルスケールに変換する前に、この閾値より小さい値はすべて閾値の値に置き換えられます。最小値は3.75e-324です。

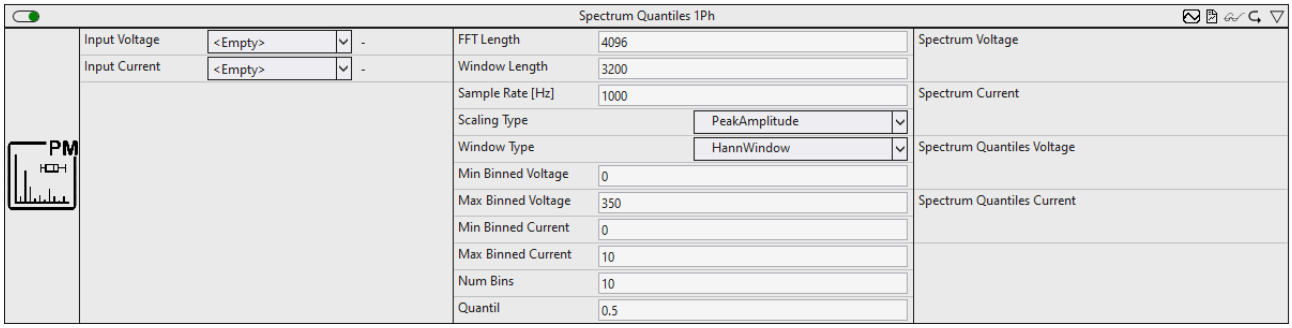
入力値

- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション）：** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Spectrum Voltage:** 電圧値のマグニチュードスペクトル。
- **Spectrum Current:** 電流値のマグニチュードスペクトル。

6.5.14.2.3.2.2 Spectrum Quantiles 1Ph .2



*Spectrum Quantiles*モジュールは、*Spectrum*モジュールと同様に電流値と電圧値のマグニチュードスペクトルを計算します。加えて、スペクトル分布のp分位数も計算できます。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_Spectrum_Quantiles_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **FFT Length：** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length：** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz]：** 入力信号のサンプル・レート（サンプル/秒）。
- **Scaling Type：** 絶対スケーリングが必要な場合に使用するスケーリング方式を選択します。を参照してください。
- **Window Type：** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Transform To Decibel（オプション）：** FFTの結果をデシベル尺度（x → 20 × log10(x)）に変換するかどうかを示すブール値。

- **Decibel Threshold（オプション）：** 非常に小さな浮動小数点値で、ゼロより大きい値。ゼロの対数は数学的に定義されていないため、デシベルスケールに変換する前に、この閾値より小さい値はすべて閾値の値に置き換えられます。最小値は3.75e-324です。
- **Min Binned Voltage：** 電圧スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の下限値。
- **Max Binned Voltage:** 電圧スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の上限値。*Max Binned Voltage*は *Min Binned Voltage*より大きい値を設定する必要があります。
- **Min Binned Current:** 電流スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の下限値。
- **Max Binned Current:** 電流スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の上限値。*Max Binned Current*は *Min Binned Current*より大きい値を設定する必要があります。
- **Num Bins:** ヒストグラムのビンの数。最小値は1です。多くの場合、10と20の間の値が適切な選択肢です。*Min Binned*以下と*Max Binned*以上の値に対する2つの特別なビンは、この値には含まれません。
- **Quantile：** 分位限界値を示します。0.0から1.0の間の値にする必要があります。例えば、0.2は20%の分位数に相当します。

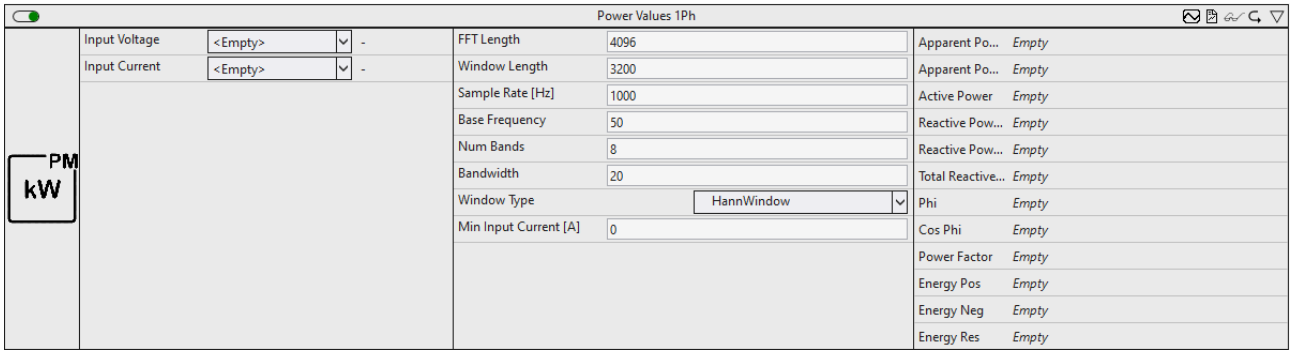
入力値

- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション）：** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Spectrum Voltage:** 電圧値のマグニチュードスペクトル。
- **Spectrum Current:** 電流値のマグニチュードスペクトル。
- **Spectrum Quantiles Voltage:** *Spectrum Voltage*分布のP分位数。
- **Spectrum Quantiles Current:** *Spectrum Current*分布のP分位数。

6.5.14.2.3.2.2 Power Values 1Ph .3



*Power Values*モジュールは、接続機器の電力値を計算します。これには基本波成分と位相シフト角が含まれます。内部的には、計算処理の一環として、各高調波およびその位相角が算出されます。

あるいは、計算に*Power Values Period*モジュールを使用することもできます。よりシンプルな計算方法を採用し、計算処理の動的な応答性を高めます。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_PowerValues_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **FFT Length :** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length :** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Base Frequency** : 第一高調波の周波数（単位：Hz）。
- **Num Bands** : RMS値を計算する周波数帯域の数を指定します。
- **Bandwidth** : 単一のRMS帯域における総帯域幅。
- **Window Type :** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。

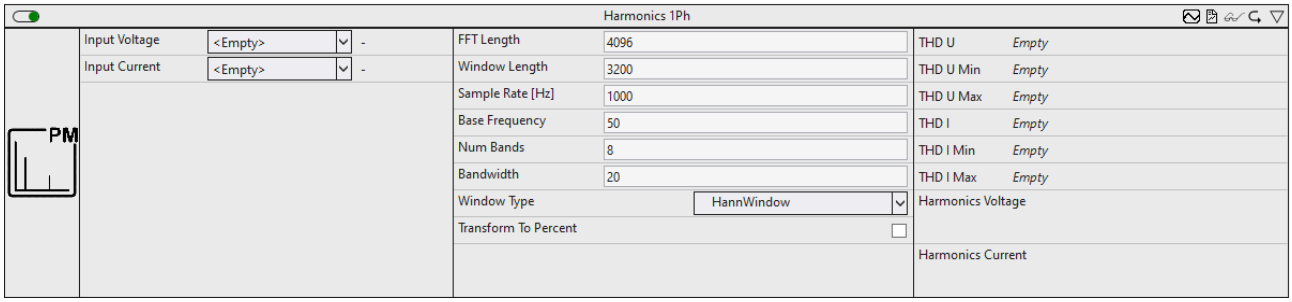
入力値

- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Apparent Power:** 総皮相電力。
- **Apparent Power 1 :** 基本皮相電力。
- **Active Power:**有効電力。
- **Reactive Power d:** 歪み無効電力。
- **Reactive Power 1:** 基本波無効電力。
- **Total Reactive Power:** 総無効電力。
- **Phi:** 位相シフト角。
- **Cos Phi :** CosPhi（有効電力/基本波皮相電力）。
- **Power Factor:** 力率（有効電力/総皮相電力）。
- **Power Quality Factor:** 電源品質係数。周波数、電圧の実効値、および電圧のTHD（全高調波歪み）を考慮し、0～1の範囲で電源の品質を簡易的に表す指標です。
- **Energy Pos:** 正方向の電力量。
- **Energy Neg :** 負方向の電力量。
- **Energy Res:** 結果の電力量。

6.5.14.2.3.2.2 Harmonics 1Ph .4



*Harmonics*モジュールは、電流と電圧の高調波を計算します。さらに、入力値のTHDは高調波から計算されます。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Harmonics_1Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Num Harmonics :** 計算する高調波の数。
- **Transform To Percent:** 結果をパーセント値で出力するかどうかを指定するブール値。第一高調波は100%に相当します。
- **FFT Length :** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length :** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Base Frequency**：第一高調波の周波数（単位：Hz）。
- **Num Bands**：RMS値を計算する周波数帯域の数を指定します。
- **Bandwidth**：単一のRMS帯域における総帯域幅。
- **Window Type :** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Transform To Percent:** 結果をパーセント値で出力するかどうかを指定するブール値。第一高調波は100%に相当します。

入力値

- **Input Voltage:** 電圧の入力値。
- **Input Current:** 電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

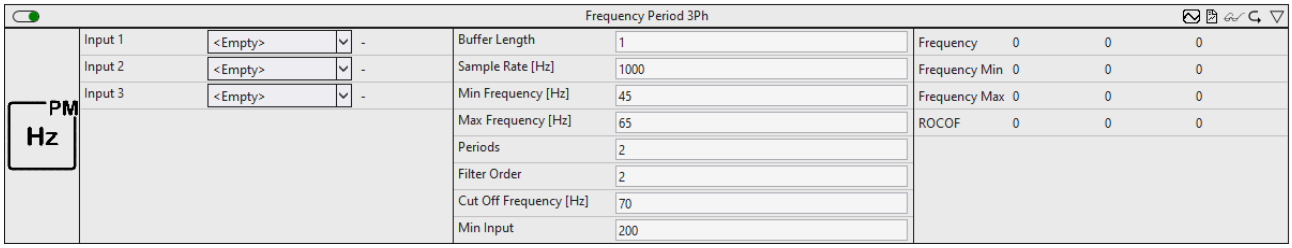
出力値

- **THD U :** 電圧のTHD。出力はパーセント値で表示されます。
- **THD U Min :** *THD U*値の最小値。*Reset Statistics*入力でリセット可能です。
- **THD U Max :** *THD U*値の最大値。*Reset Statistics*入力でリセット可能です。
- **THD I :** 電流のTHD。出力はパーセント値で表示されます。
- **THD I Min :** *THD I*の最小値。*Reset Statistics*入力でリセット可能です。
- **THD I Max :** *THD I*の最大値。*Reset Statistics*入力でリセット可能です。
- **Harmonics Voltage :** 電圧の高調波。
- **Harmonics Current :** 電流の高調波。

6.5.14.2.3.3 三相

6.5.14.2.3.3. 信号周期に基づく 1

6.5.14.2.3.3.1 Frequency Period 3Ph .1



*Frequency Period*モジュールは、入力信号の基本周波数を計算します。そのため、信号はまずバターワース型ローパスフィルタでフィルタリングされます。その後、フィルタ後の信号からゼロクロス点を検出し、これらの間隔に基づいて信号の周波数を計算します。結果は、設定に応じて、1つまたは複数の区間を参照します。統計結果は、ランタイム全体、または最後に統計がリセットされた時点からのデータを参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_Frequency_Period_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Min Frequency [Hz]** : 計測対象として想定される最小周波数。
- **Max Frequency [Hz]** : 計測対象として想定される最大周波数。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。
- **Filter Order:**ローパスフィルタの次数。フィルタの安定性を考慮して設定する必要があります。10次までの値が許可されます。
- **Cut Off Frequency [Hz]:**ローパスフィルタのカットオフ周波数。
- **Min Input:**1周期の最小入力値（RMS）。このパラメータは、下限を下回る入力値の計算を防止します。
- **ROCOF Avg Window Order:**ROCOF（周波数変化率）の計算に用いるスライディング平均のウィンドウサイズ。

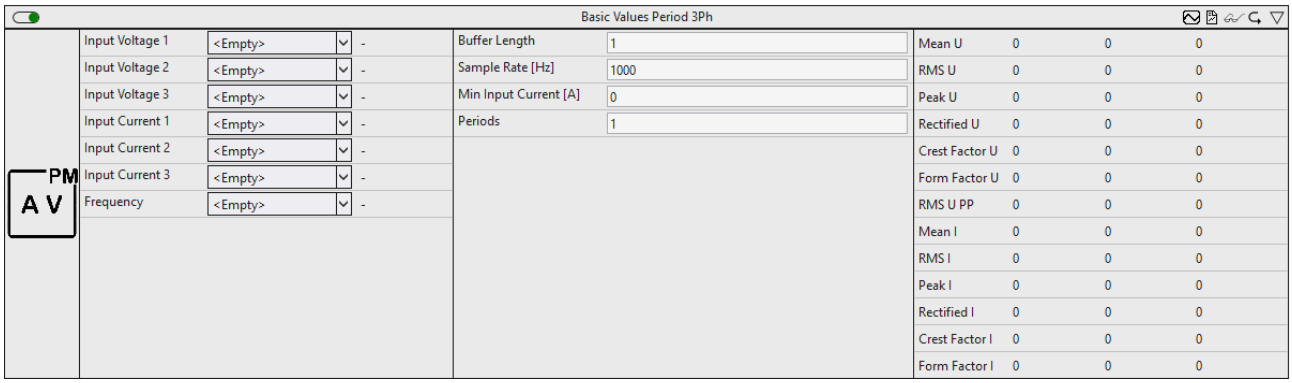
入力値

- **Input 1 .. Input 3:**電流または電圧の入力値。
- **Reset Statistics（オプション） :** *TRUE* を設定すると、出力の最小値と最大値がリセットされます。

出力値

- **Frequency:** 2回以上のゼロクロスから算出された周波数。
- **Frequency Min:** 計算された周波数の最小値。*Reset Statistics*入力でリセット可能です。
- **Frequency Max :** 計算された周波数の最大値。*Reset Statistics*入力でリセット可能です。
- **ROCOF :** 周波数変化率（Rate of change of frequency）。

6.5.14.2.3.3.1 Basic Values Period 3Ph .2



*Basic Values Period*モジュールは、電流と電圧の時系列信号に対して、解析指標を算出します。これらには、電流と電圧の平均値、実効値、ピーク値、整流値、クレストファクタ、フォームファクタが含まれます。結果は、設定した信号周期数を基準としています。周期の長さは、周期開始時点で入力された周波数に対応します。オプションの統計結果は、ランタイム全体または統計結果が最後にリセットされた時間を参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_BasicValues_Period_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。

入力値

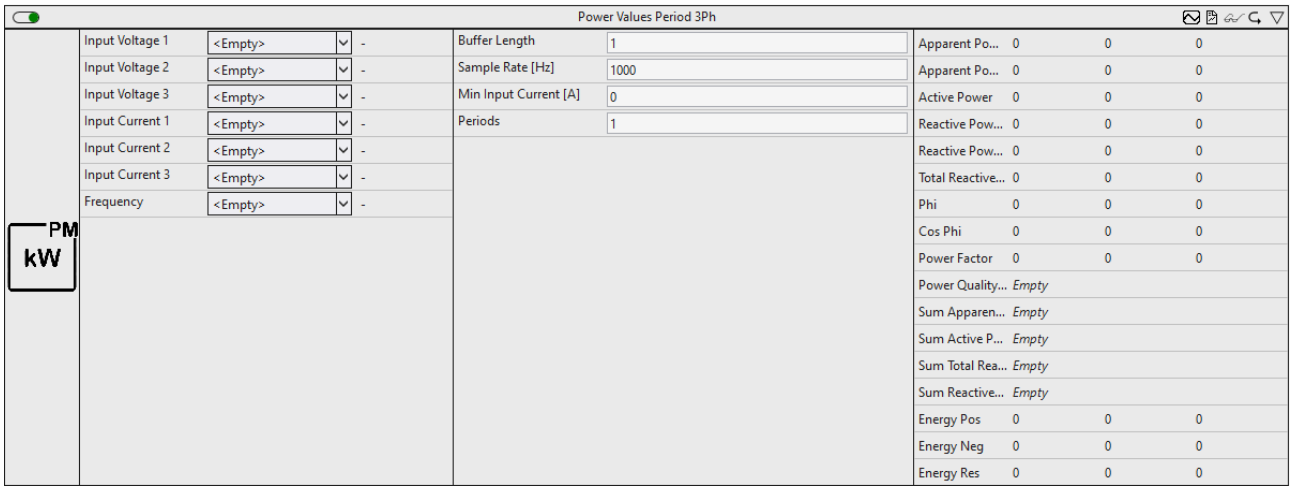
- **Frequency:** 入力信号の現在の周波数。周期開始時に、周期長を決定するために使用されます。*Frequency Period1 Ch*モジュールの出力が使用できます。
- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Mean U:** n周期の平均電圧値。
- **RMS U :** n周期の電圧実効値。
- **Peak U :** n周期の電圧ピーク値。
- **Rectified U:** n周期の整流電圧値。
- **Crest Factor U :** 電圧のクレストファクタ（ピーク値/実効値）。
- **Form Factor U :** 電圧のフォームファクタ（実効値／補正値）。
- **Mean I:** n周期の電流の平均値。
- **RMS I :** n周期の電流の実効値。
- **Peak I :** n周期の電流のピーク値。
- **Rectified I :** n周期の電流の整流値。
- **Crest Factor I :** 電流のクレストファクタ（ピーク値/実効値）。

- **Form Factor I :** 電流のフォームファクタ（実効値／補正値）。

6.5.14.2.3.3.1 Power Values Period 3Ph .3



*Power Values Period*モジュールは、接続機器の電力値を計算します。これには基本波成分と位相シフト角が含まれます。このため、時間範囲内の信号曲線に加えて、入力信号の最初の高調波のみが計算に使用されます。これらのアルゴリズムの特長は、計算処理の動的な応答性が高い点にあります。結果は、設定した信号周期数を基準としています。周期の長さは、周期開始時点で入力された周波数に対応します。統計結果は、ランタイム全体、または最後に統計がリセットされた時点からのデータを参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_PowerValues_Period_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。

入力値

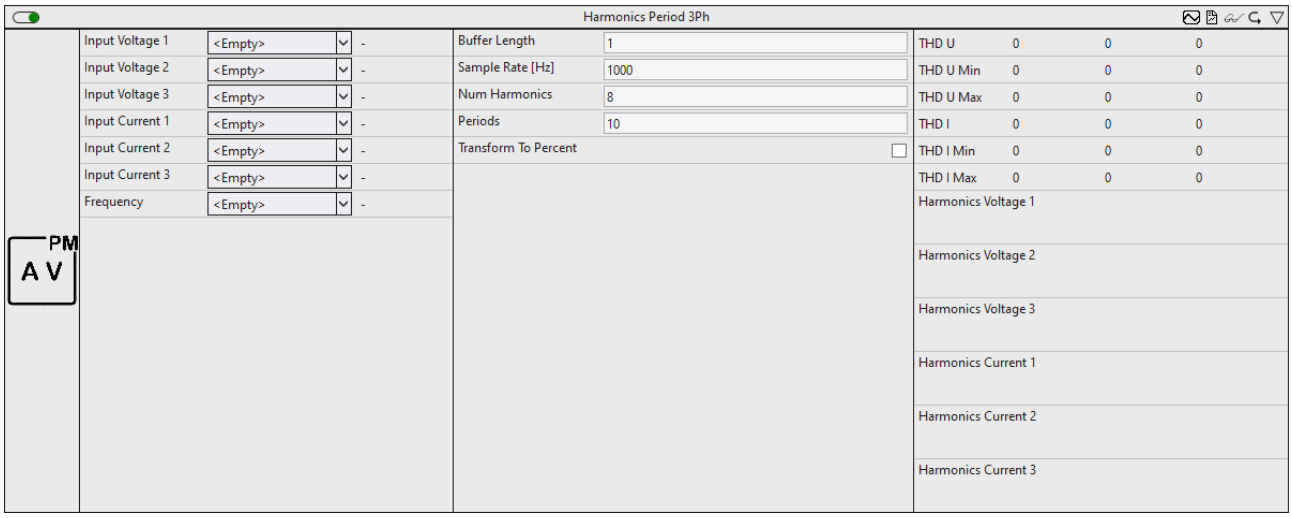
- **Frequency:** 入力信号の現在の周波数。周期開始時に、周期長を決定するために使用されます。*Frequency Period1 Ch*モジュールの出力が使用できます。
- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Apparent Power:** 総皮相電力。
- **Apparent Power 1 :** 基本皮相電力。
- **Active Power:**有効電力。
- **Reactive Power d:** 歪み無効電力。
- **Reactive Power 1:** 基本波無効電力。
- **Total Reactive Power:** 総無効電力。
- **Phi:** 位相シフト角。

- **Cos Phi :** CosPhi（有効電力/基本波皮相電力）。
- **Power Factor:** 力率（有効電力/総皮相電力）。
- **Power Quality Factor:** 電源品質係数。周波数、電圧の実効値、および電圧のTHD（全高調波歪み）を考慮し、0～1の範囲で電源の品質を簡易的に表す指標です。
- **Energy Pos:** 正方向の電力量。
- **Energy Neg :** 負方向の電力量。
- **Energy Res:** 結果の電力量。
- **Sum Apparent Power:** 全相の皮相電力の合計。
- **Sum Active Power:** 全相の有効電力の合計。
- **Sum Total Reactive Power:**全相の無効電力の合計。
- **Sum Reactive Power 1 Res:**全相の基本波シフト無効電力値の合計。

6.5.14.2.3.3.1 Harmonics Period 3Ph .4



*Harmonics Period*モジュールは、電流と電圧の高調波を計算します。さらに、入力値のTHDは高調波から計算されます。*Harmonics*モジュールとは対照的に、結果は設定可能な信号周期数に基づいています。周期の長さは、周期開始時点で入力された周波数に対応します。統計結果は、ランタイム全体、または最後に統計がリセットされた時点からのデータを参照します。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_Harmonics_Period_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Buffer Length:**入力バッファの長さ。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。
- **Periods:**計算に使用される周期の数。(周期長＝サンプルレート/周波数）。
- **Num Harmonics :** 計算する高調波の数。
- **Transform To Percent:** 結果をパーセント値で出力するかどうかを指定するブール値。第一高調波は100%に相当します。

入力値

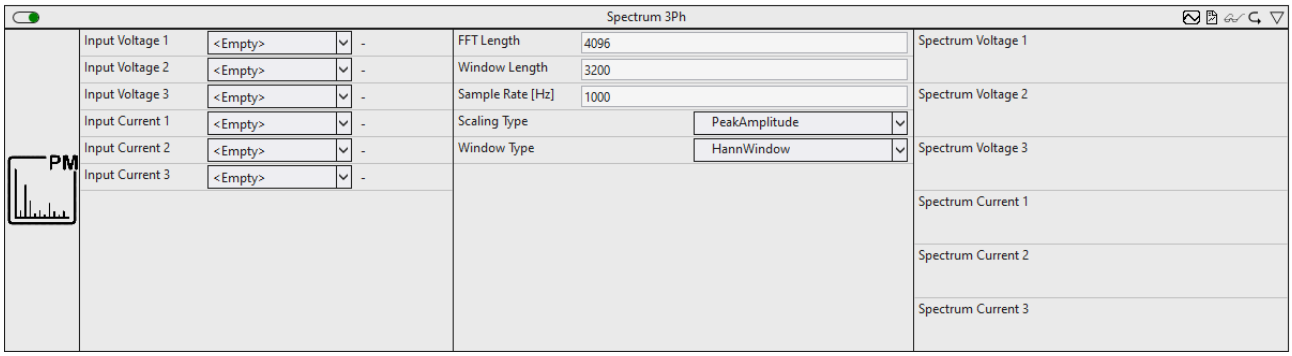
- **Frequency:** 入力信号の現在の周波数。周期開始時に、周期長を決定するために使用されます。*Frequency Period1 Ch*モジュールの出力が使用できます。
- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション）：** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **THD U：** 電圧のTHD。出力はパーセント値で表示されます。
- **THD U Min：** *THD U*値の最小値。*Reset Statistics*入力でリセット可能です。
- **THD U Max：** *THD U*値の最大値。*Reset Statistics*入力でリセット可能です。
- **THD I：** 電流のTHD。出力はパーセント値で表示されます。
- **THD I Min：** *THD I*の最小値。*Reset Statistics*入力でリセット可能です。
- **THD I Max：** *THD I*の最大値。*Reset Statistics*入力でリセット可能です。
- **Harmonics Voltage：**電圧の高調波。
- **Harmonics Current：**電流の高調波。

6.5.14.2.3.3.　周波数範囲に基づく 2

6.5.14.2.3.3.2　Spectrum 3Ph .1



*Spectrum*モジュールは、電流値と電圧値のマグニチュードスペクトルを計算します。これらは周波数範囲の入力信号を解析するのに適しています。入力バッファのサイズはウィンドウ長の半分です。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Spectrum_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **FFT Length：** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length：** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz]：**入力信号のサンプル・レート（サンプル/秒）。
- **Scaling Type：** 絶対スケーリングが必要な場合に使用するスケーリング方式を選択します。を参照してください。
- **Window Type：** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。

- **Transform To Decibel（オプション）：** FFTの結果をデシベル尺度（x → 20 × log10(x)）に変換するかどうかを示すブール値。
- **Decibel Threshold（オプション）：** 非常に小さな浮動小数点値で、ゼロより大きい値。ゼロの対数は数学的に定義されていないため、デシベルスケールに変換する前に、この閾値より小さい値はすべて閾値の値に置き換えられます。最小値は3.75e-324です。

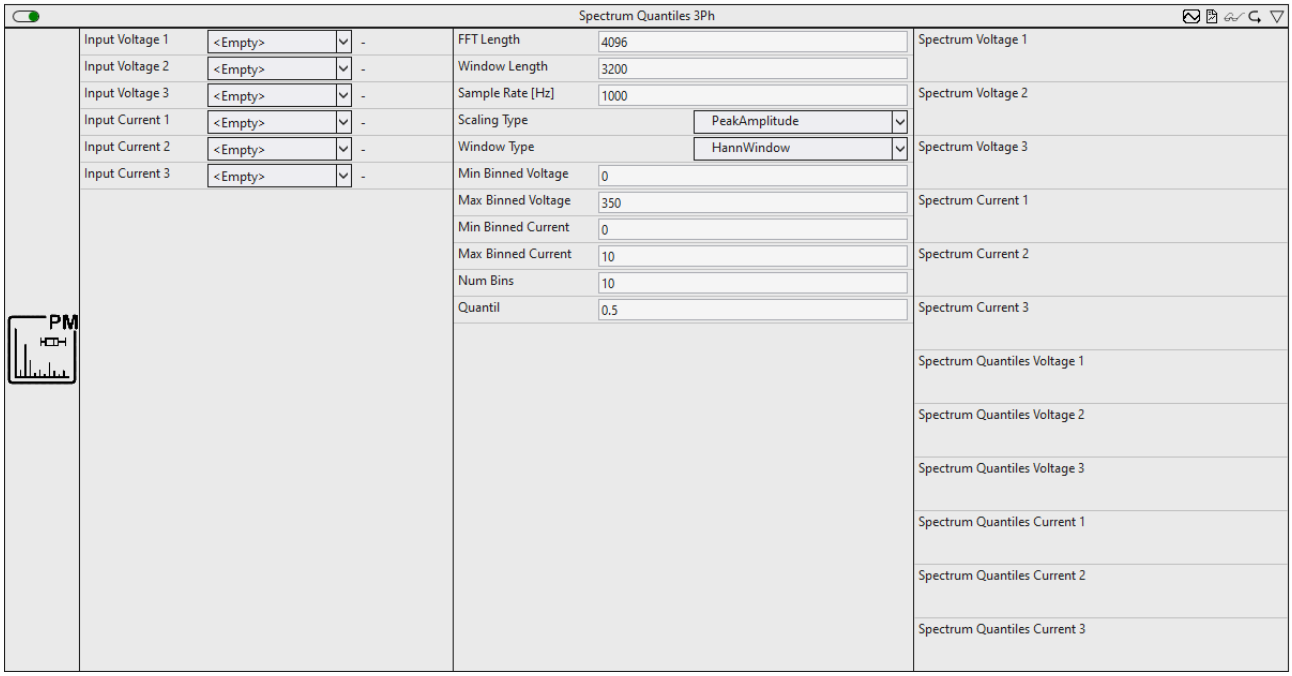
入力値

- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション）：** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Spectrum Voltage 1 .. Spectrum Voltage 3:**電圧値のマグニチュードスペクトル。
- **Spectrum Current 1 ... Spectrum Current 3:**電流値のマグニチュードスペクトル。

6.5.14.2.3.3.2 Spectrum Quantiles 3Ph .2



*Spectrum Quantiles*モジュールは、*Spectrum*モジュールと同様に電流値と電圧値のマグニチュードスペクトルを計算します。加えて、スペクトル分布のp分位数も計算できます。

Beckhoff Information Systemでは、対応するPLCファンクションブロック FB_PMA_Spectrum_Quantiles_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **FFT Length：** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length：** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz]：**入力信号のサンプル・レート（サンプル/秒）。
- **Scaling Type：** 絶対スケーリングが必要な場合に使用するスケーリング方式を選択します。を参照してください。

- **Window Type :** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Transform To Decibel（オプション） :** FFTの結果をデシベル尺度（x → 20 × log10(x)）に変換するかどうかを示すブール値。
- **Decibel Threshold（オプション） :** 非常に小さな浮動小数点値で、ゼロより大きい値。ゼロの対数は数学的に定義されていないため、デシベルスケールに変換する前に、この閾値より小さい値はすべて閾値の値に置き換えられます。最小値は3.75e-324です。
- **Min Binned Voltage :** 電圧スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の下限値。
- **Max Binned Voltage:** 電圧スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の上限値。*Max Binned Voltage*は *Min Binned Voltage*より大きい値を設定する必要があります。
- **Min Binned Current:** 電流スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の下限値。
- **Max Binned Current:** 電流スペクトル計算において、出力データが通常のヒストグラムビンでカウントされる際の上限値。*Max Binned Current*は *Min Binned Current*より大きい値を設定する必要があります。
- **Num Bins:** ヒストグラムのビンの数。最小値は1です。多くの場合、10と20の間の値が適切な選択肢です。*Min Binned*以下と*Max Binned*以上の値に対する2つの特別なビンは、この値には含まれません。
- **Quantile :** 分位限界値を示します。0.0から1.0の間の値にする必要があります。例えば、0.2は20%の分位数に相当します。

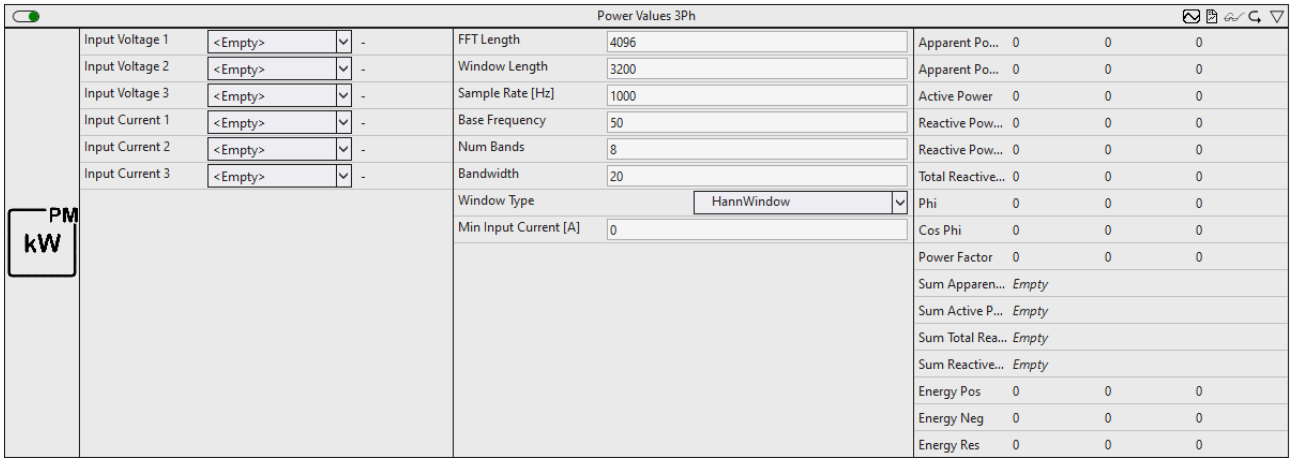
入力値

- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Spectrum Voltage 1 .. Spectrum Voltage 3:**電圧値のマグニチュードスペクトル。
- **Spectrum Current 1 ... Spectrum Current 3:**電流値のマグニチュード・スペクトル
- **Spectrum Quantiles Voltage 1 ... Spectrum Quantiles Voltage 3:***Spectrum Voltage*分布のP分位数。
- **Spectrum Quantiles Current 1 ... Spectrum Quantiles Current 3:***Spectrum Current*分布のP分位数。

6.5.14.2.3.3.2 Power Values 3Ph .3



*Power Values*モジュールは、接続機器の電力値を計算します。これには基本波成分と位相シフト角が含まれます。内部的には、計算処理の一環として、各高調波およびその位相角が算出されます。

あるいは、計算に*Power Values Period*モジュールを使用することもできます。よりシンプルな計算方法を採用し、計算処理の動的な応答性を高めます。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_PowerValues_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **FFT Length :** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length :** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Base Frequency**：第一高調波の周波数（単位：Hz）。
- **Num Bands**：RMS値を計算する周波数帯域の数を指定します。
- **Bandwidth**：単一のRMS帯域における総帯域幅。
- **Window Type :** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Min Input Current [A]:** 電流の最小入力値（実効値）。このパラメータは、下限を下回る入力値の計算を防止します。

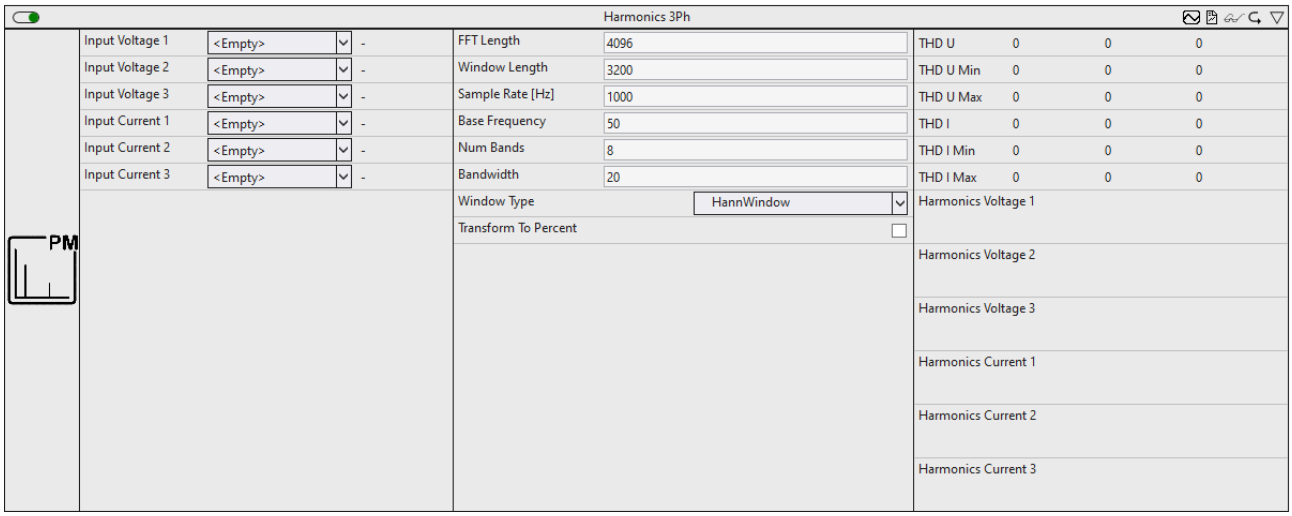
入力値

- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **Apparent Power:** 総皮相電力。
- **Apparent Power 1 :** 基本皮相電力。
- **Active Power:**有効電力。
- **Reactive Power d:** 歪み無効電力。
- **Reactive Power 1:** 基本波無効電力。
- **Total Reactive Power:** 総無効電力。
- **Phi:** 位相シフト角。
- **Cos Phi :** CosPhi（有効電力/基本波皮相電力）。
- **Power Factor:** 力率（有効電力/総皮相電力）。
- **Power Quality Factor:** 電源品質係数。周波数、電圧の実効値、および電圧のTHD（全高調波歪み）を考慮し、0～1の範囲で電源の品質を簡易的に表す指標です。
- **Energy Pos:** 正方向の電力量。
- **Energy Neg :** 負方向の電力量。
- **Energy Res:** 結果の電力量。
- **Sum Apparent Power:** 全相の皮相電力の合計。
- **Sum Active Power:** 全相の有効電力の合計。
- **Sum Total Reactive Power:**全相の無効電力の合計。
- **Sum Reactive Power 1 Res:**全相の基本波シフト無効電力値の合計。

6.5.14.2.3.3.2 Harmonics 3Ph .4



*Harmonics*モジュールは、電流と電圧の高調波を計算します。さらに、入力値のTHDは高調波から計算されます。

Beckhoff Information Systemでは、対応するPLCファンクションブロックFB_PMA_Harmonics_3Phのドキュメントを参照できます。

Beckhoff Information Systemでは、TF3650 TwinCAT 3 Power Monitoring機能のドキュメントと、対応するPLCライブラリの説明を参照できます。

パラメータ設定

- **Num Harmonics :** 計算する高調波の数。
- **Transform To Percent:** 結果をパーセント値で出力するかどうかを指定するブール値。第一高調波は100%に相当します。
- **FFT Length :** FFTの長さ。1より大きく、2の累乗である必要があります。
- **Window Length :** 解析ウィンドウの長さ（サンプル単位）。長さは1より大きく、偶数である必要があります。*FFT Length*より大きい値を設定します。
- **Sample Rate [Hz] :** 入力信号のサンプル・レート（サンプル/秒）。
- **Base Frequency** : 第一高調波の周波数（単位：Hz）。
- **Num Bands** : RMS値を計算する周波数帯域の数を指定します。
- **Bandwidth** : 単一のRMS帯域における総帯域幅。
- **Window Type :** 使用するウィンドウ関数を定義します。ウィンドウのタイプは*HannWindow*がデフォルト値として適しています。*RectangularWindow*を使用することで、ウィンドウをオフにできます。を参照してください。
- **Transform To Percent:** 結果をパーセント値で出力するかどうかを指定するブール値。第一高調波は100%に相当します。

入力値

- **Input Voltage 1 .. Input Voltage 3:** 電圧の入力値。
- **Input Current 1 .. Input Current 3:**電流の入力値。
- **Reset Statistics（オプション） :** TRUE は、出力の最小値と最大値をリセットします。

出力値

- **THD U :** 電圧のTHD。出力はパーセント値で表示されます。
- **THD U Min :** *THD U*値の最小値。*Reset Statistics*入力でリセット可能です。
- **THD U Max :** *THD U*値の最大値。*Reset Statistics*入力でリセット可能です。

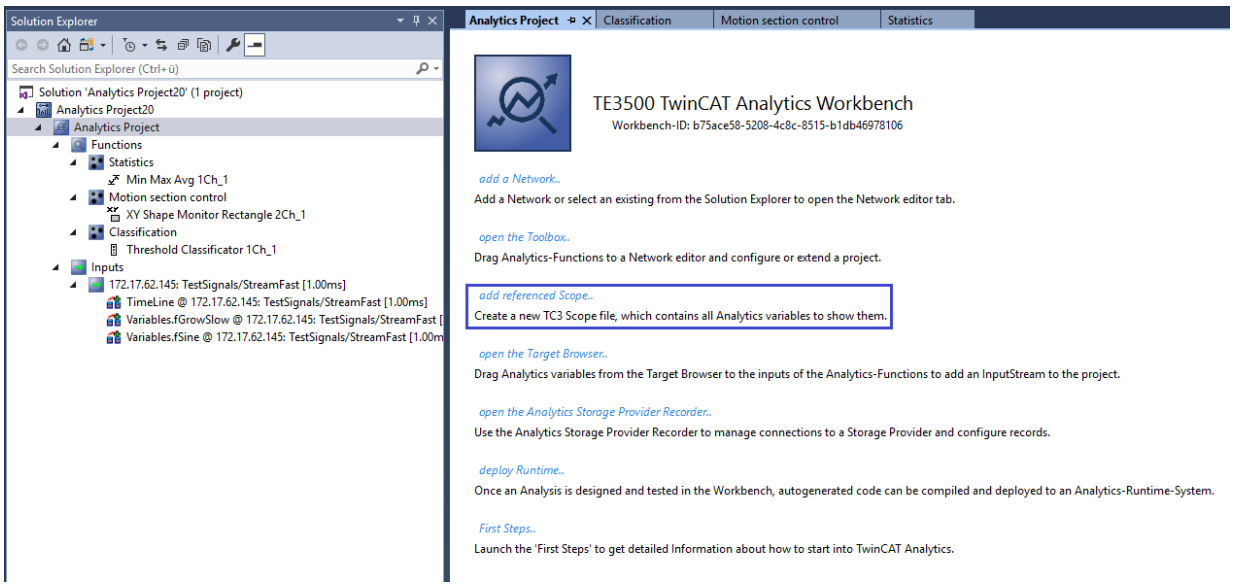
- **THD I :** 電流のTHD。出力はパーセント値で表示されます。
- **THD I Min :** *THD I*の最小値。*Reset Statistics*入力でリセット可能です。
- **THD I Max :** *THD I*の最大値。*Reset Statistics*入力でリセット可能です。
- **Harmonics Voltage :** 電圧の高調波。
- **Harmonics Current :** 電流の高調波。

6.6 スコープとの連携

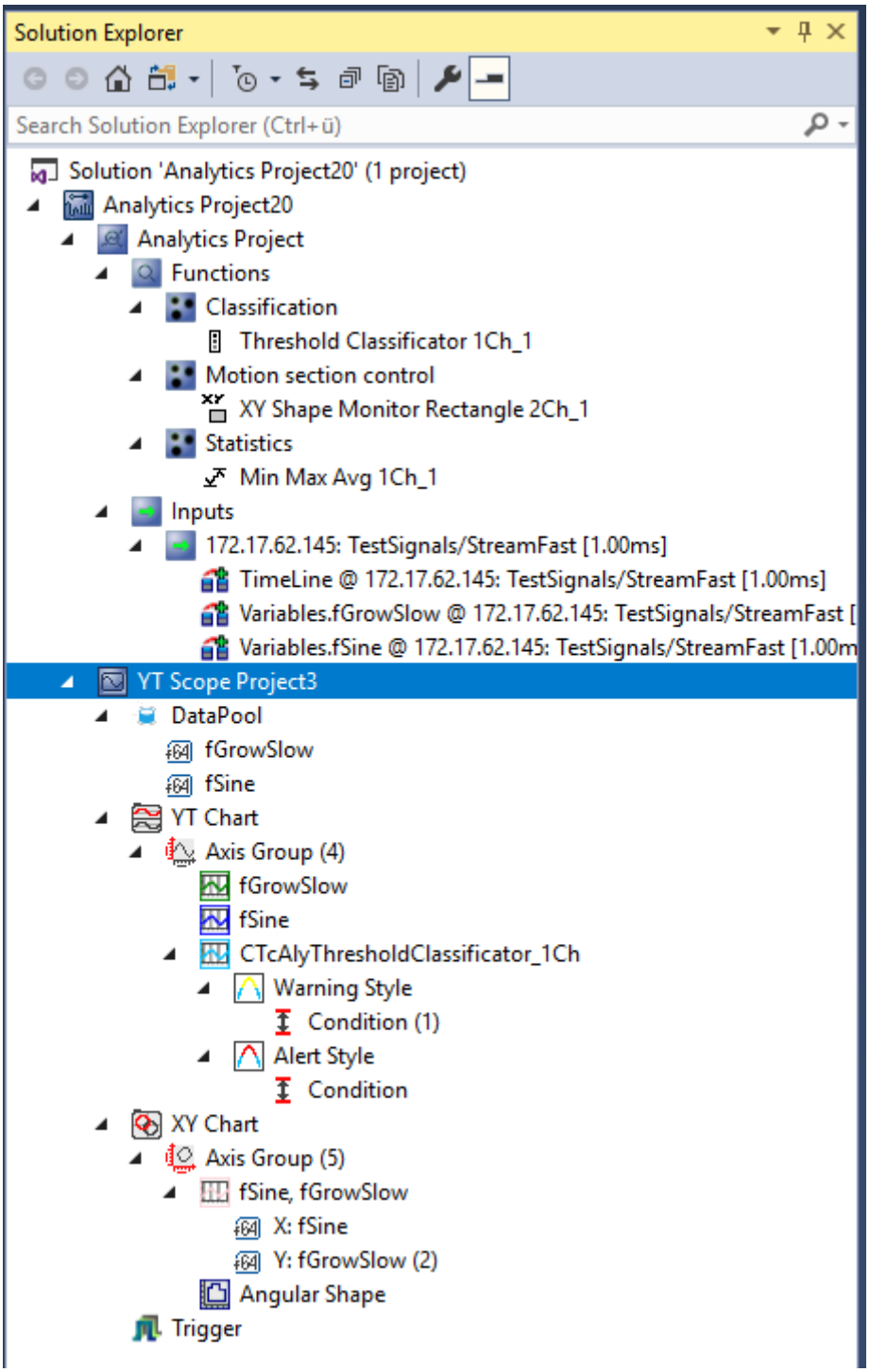
TwinCAT Analyticsエンジニアリングツールは、Analyticsコンフィグレータとスコープビュー間のスムーズな連携を実現します。これにより、データストリームの重要な値をハイライト表示し、正確なサイクルで他のプロセスデータと比較・検証できます。さらに、平均値や最大値など、アルゴリズムの結果データをスコープビューに表示することも可能です。

✓ 解析方法を設定したら、**Analytics Project**のスタートページに切り替えます。

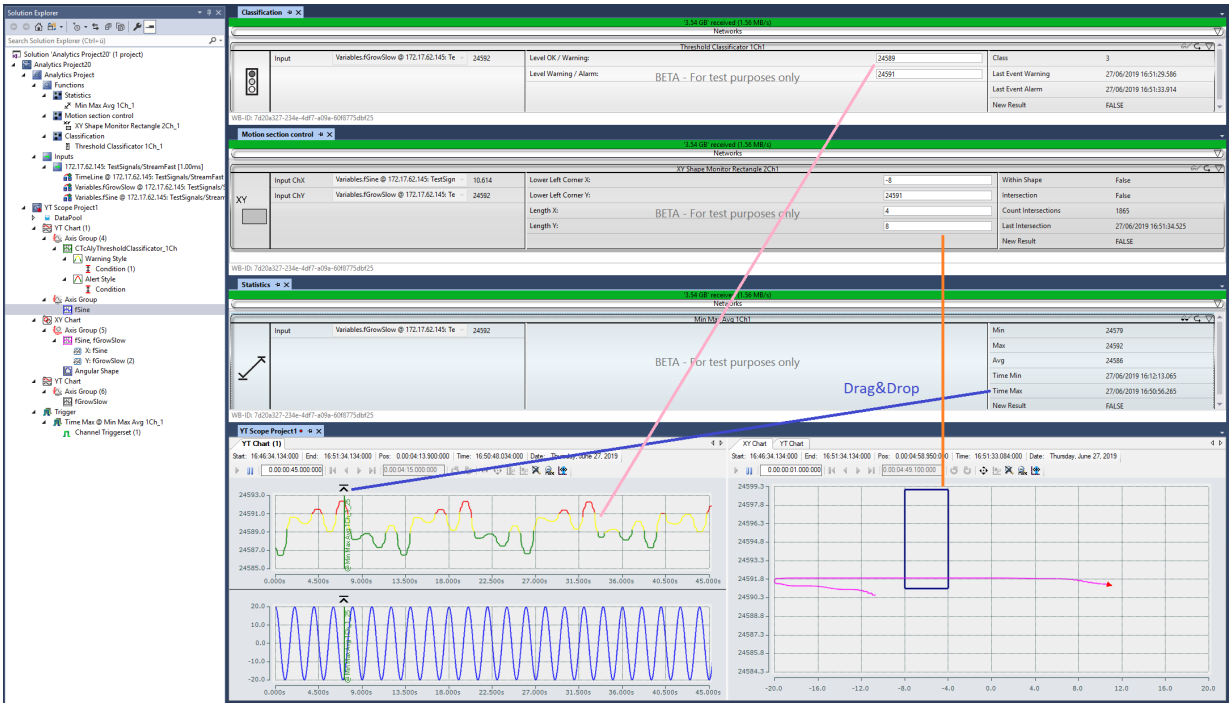
1. **add referenced Scope…**をクリックします。



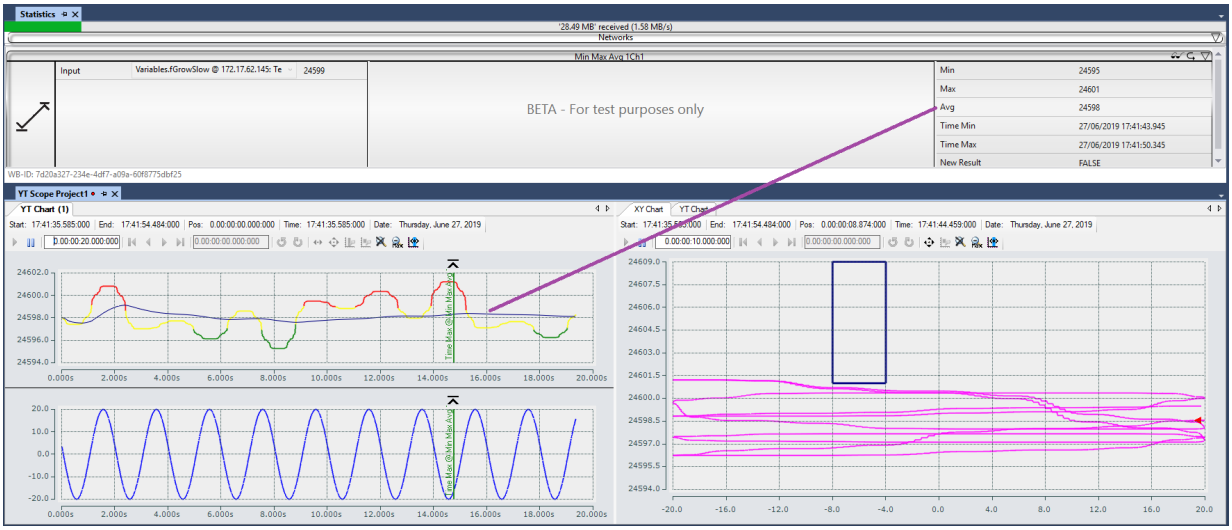
⇨ 適切なスコープ設定が自動的にプロジェクトに追加されます。



2. 様々なアルゴリズムからデータを可視化する方法が数多く用意されています。アルゴリズム出力のタイムスタンプから、必要なものをグラフにドラッグ＆ドロップできます。Scopeでは、イベントの座標をカラーマーカで示します（下図の青線を参照）。

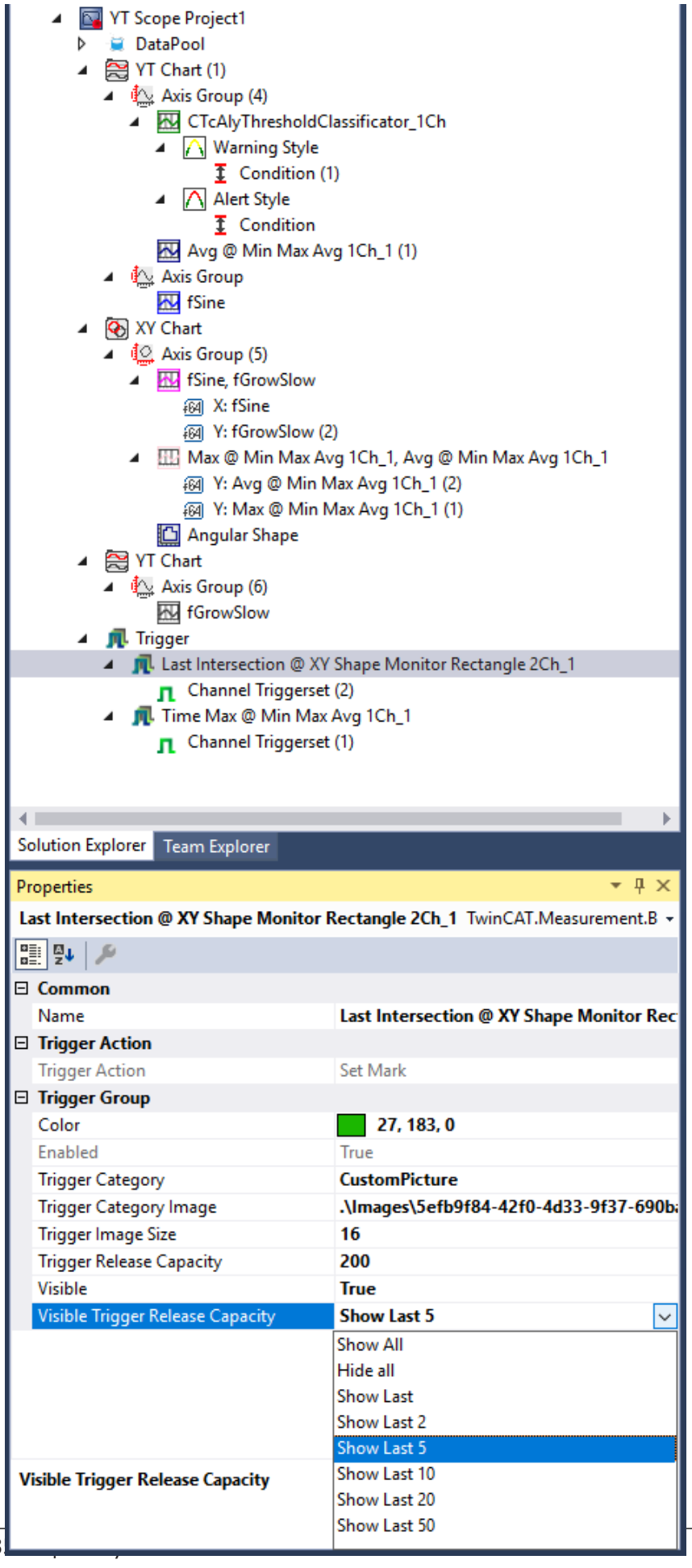


3. 各種のShapeアルゴリズムごとに、ソフトウェアは指定された定義に基づく形状を含むXYダイアグラムをScopeの設定画面に作成します（上図のオレンジ線参照）。Threshold（閾値）分類アルゴリズムでは、スコープビュー上でチャンネルカラーから黄色（警告）、さらに赤色（アラーム）へと動的に色が変化します（ピンク色の線で表示）。
4. さらに、記録前または記録中に、アルゴリズムの結果値をグラフにドラッグ＆ドロップして、平均値、最小値、最大値などを示す新しいチャンネルを追加できます。



5. 表示するイベント/マーカの数をコントロールするために、スコープビューツリーで適切なトリガーグループをクリックします。

6. Propertiesウィンドウでは、Visible Trigger Release Capacity（表示トリガーリリース容量）の値を設定できます。



⇨ 以下より選択可能：

- All（全て表示）
- Hide All（全て非表示）
- Show Last（最後を表示）
- Show Last 2（最後の2件を表示）
- Show Last 5（最後の5件を表示）
- Show Last 10（最後の10件を表示）
- Show Last 20（最後の20件を表示）
- Show Last 50（最後の50件を表示）

ネットワークテンプレートに保存されたスコープの設定

作成したスコープの設定は、関連するネットワークと共にnetwork templete（ネットワークテンプレート） [▶ 352] に保存できます。ネットワークが再度使用される際に自動的に同じスコープの設定を取得できます。

Analyticsの設定とスコープビューとの時間的な関連性

スコープビューの記録時間はAnalyticsの記録時間と異なる場合があります。特に、スコープビューでリングバッファが選択されている場合、重要な解析値が過去のスコープ記録に保存されている可能性があります。

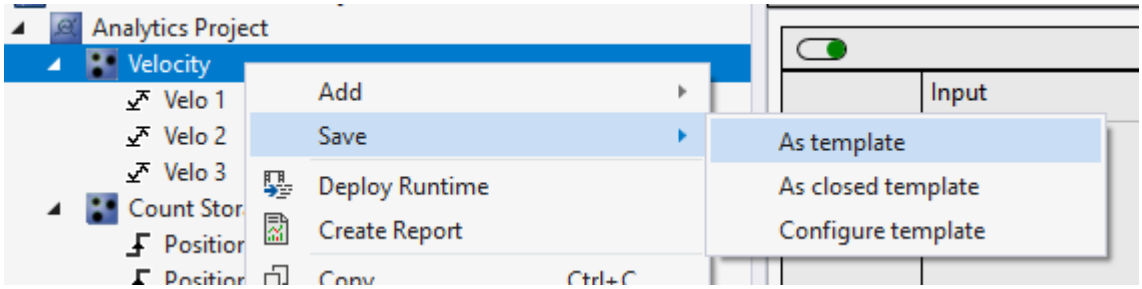
6.6.1 ネットワークテンプレートでのスコープ設定

ネットワークの設定に加えて、接続されたスコーププロジェクトの設定もネットワークテンプレートに保存できます。ただし、保存できるのは、選択したネットワークに接続しているスコープのみです。

保存するスコープ設定に手動で調整を加える必要があるかどうかによって、テンプレートの保存方法は2つあります。

手動の設定なしで保存

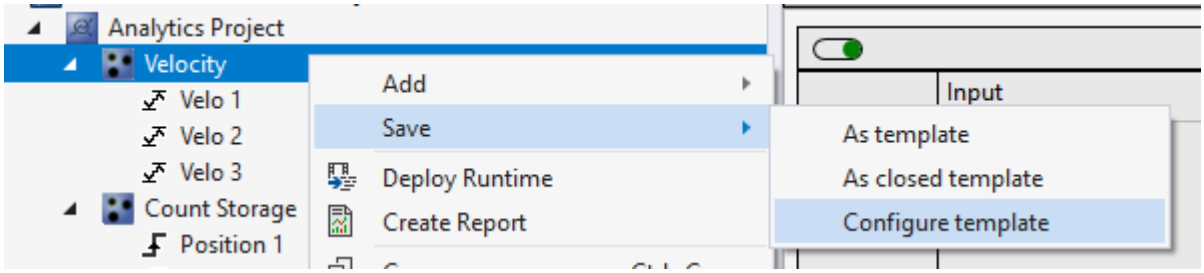
スコープ設定を手動で調整することなくテンプレートに保存したい場合は、コンテキストメニューの**Save-As template**と**Save-As closed template**を選択します。



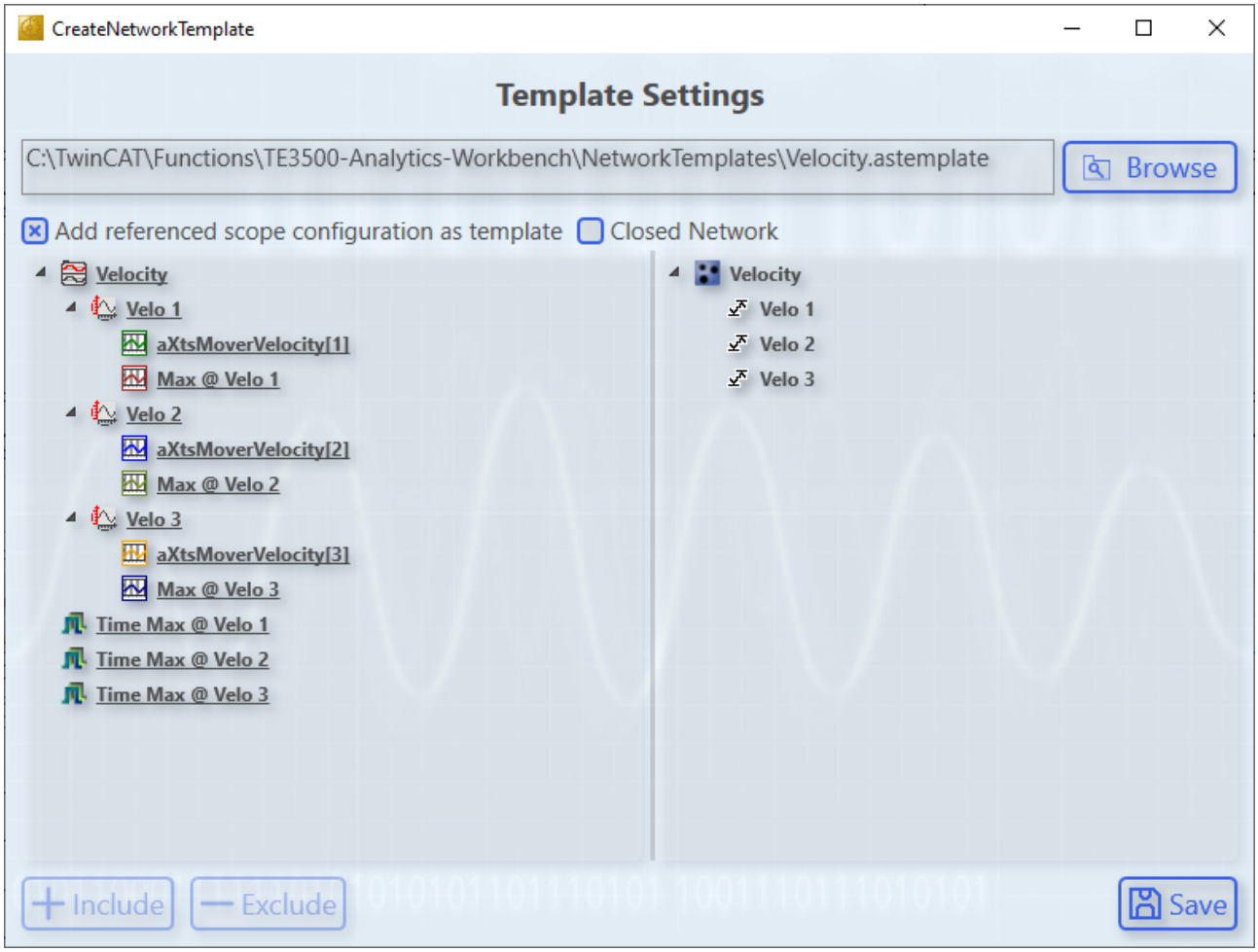
次のダイアログで、テンプレートの保存場所と名前を指定します。この時点で、ファイルの種類**Save as type**を使用して、スコープ設定を含めて保存する**Scope Network Template**か、ネットワークのみを保存する**Network Template**かを選択できます。

手動の設定で保存

スコープの設定をチェックし、必要に応じて保存前に調整する場合は、コンテキストメニューの一番下の項目**Configure template**を選択します。



スコープ設定を調整するには、新しいウィンドウが開き、左側にスコープの全コンポーネント、右側にアナリティクスの全コンポーネントが表示されます：

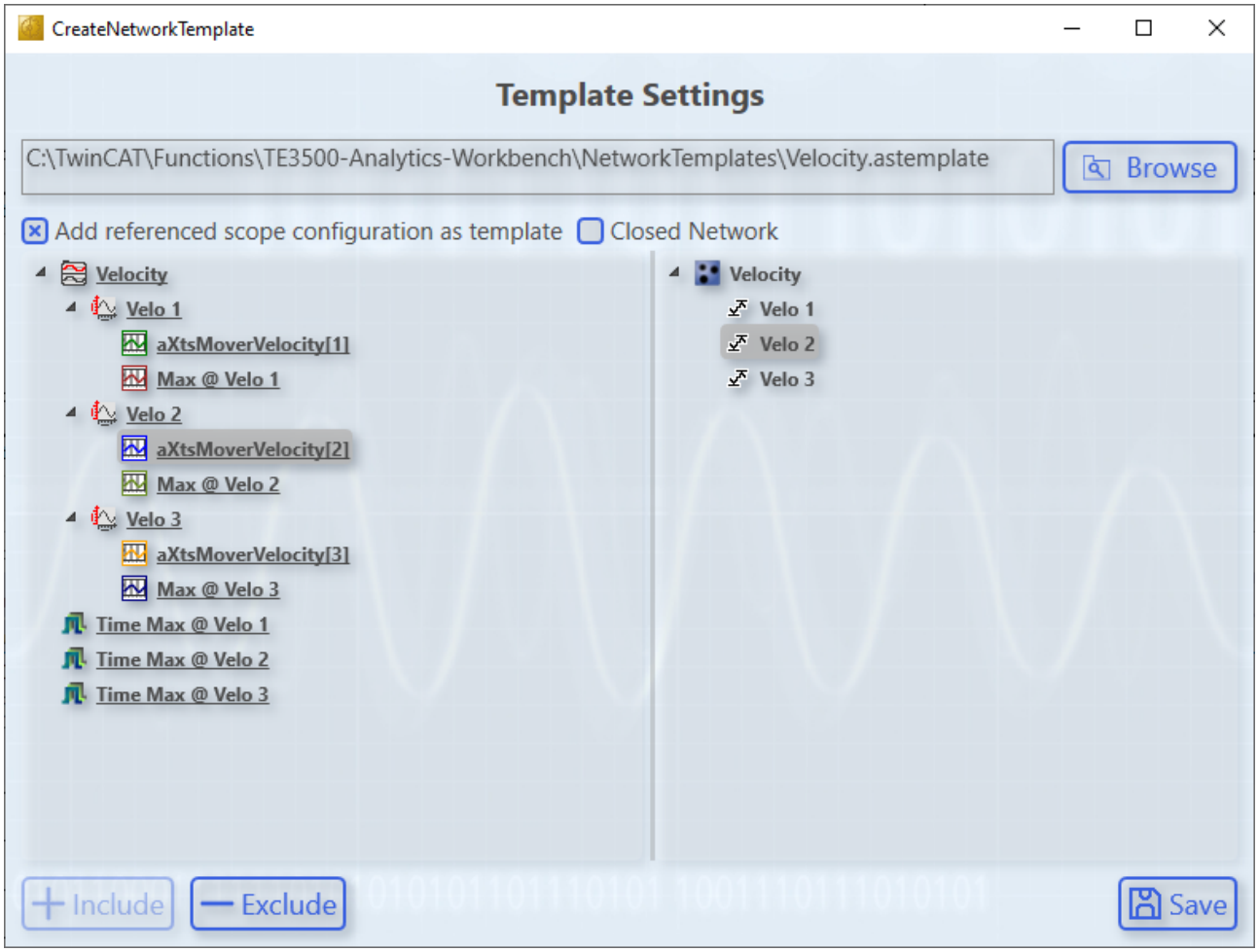


上の行では、テンプレートの保存場所と名前を設定できます。その直下で、テンプレートの一般的な設定ができます。

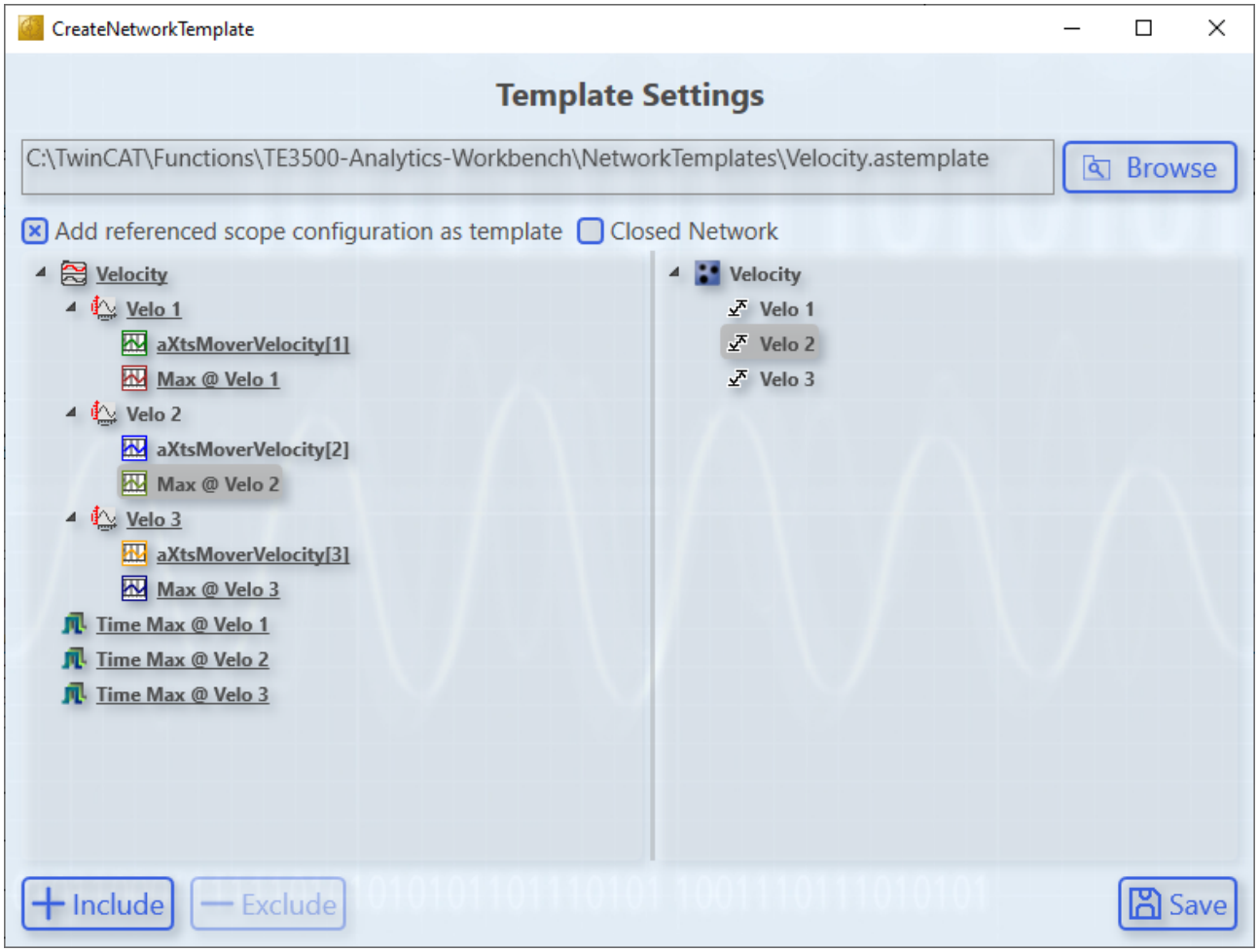
- **Add referenced scope configuration as template**
  この設定を有効にすると、スコープ設定がテンプレートに保存されます。
- **Closed Network**
  この設定は、どの形式でtemplate [▶ 74]を保存するかを決定します。この設定は、ネットワークに少なくとも1つのネットワーク入力と1つのネットワーク出力が設定されている場合にのみ表示されます。

以下の2つのツリービューでは、必要に応じてScope設定を編集するためのコンフィギュレーションが表示されます。ここでは、スコープのコンポーネントをテンプレートから除外したり、再び追加したりすることができます。
テンプレートに追加された全コンポーネントは、スコープビュー上で下線付きで表示されます。

スコープツリーで要素を選択すると、コンポーネントに直接依存する全てのAnalyticsコンポーネントも選択されます。



上記の例では、**aXtsMoverVelocity[2]**を選択すると、チャンネルがモジュールの入力変数を示すため、自動的にモジュールの**Velo 2**が選択されます。
下線が引かれたままテンプレートに保存されているコンポーネントが選択されている場合、**Exclude**ボタンを使ってテンプレートから削除できます。
テンプレートから要素を削除すると、その下にある全ての要素も自動的に削除されます。さらに、上位の要素は、その下の要素も全て削除された場合に削除されます。



要素の削除と同様に、スコープコンポーネントを選択して、**Include**ボタンでテンプレートに追加できます。
以前に選択解除された場合でも、その上層にある全ての要素も自動的にテンプレートに追加されることに注意してください。

テンプレートからスコープ設定を追加

スコープ設定でテンプレートを使用するには、Analyticsプロジェクトにテンプレートを追加する必要があります。既にプロジェクトが参照するスコーププロジェクトがある場合、テンプレート設定はそこに直接追加されます。まだ参照スコーププロジェクトが存在しない場合、テンプレートの設定は後にスコーププロジェクトが作成されるときに追加されます。
設定がプロジェクトに追加される前に、自動追加をキャンセルできるクエリが開きます。

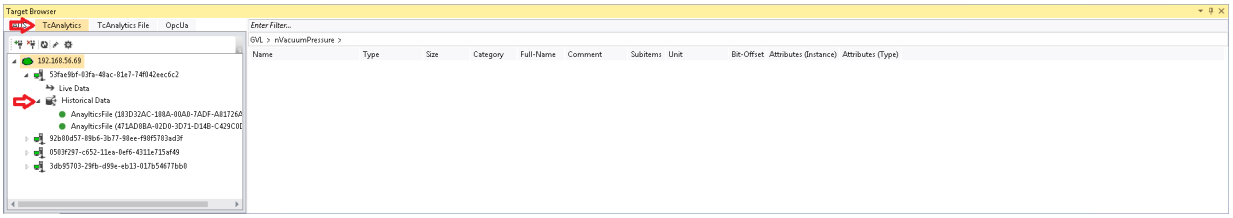
6.7 履歴データの取り扱い

履歴データは、Analytics WorkbenchまたはAnalytics Service Tool.で解析できます。記録されたデータを確認するには、TwinCAT Target Browserが必要です。

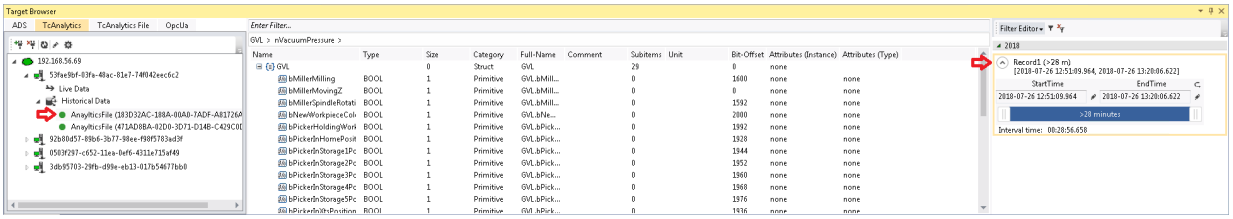
TwinCAT Target Browserからのデータ選択

履歴データは、Target Browserから解析アルゴリズムの入力に直接引き出すことができます。

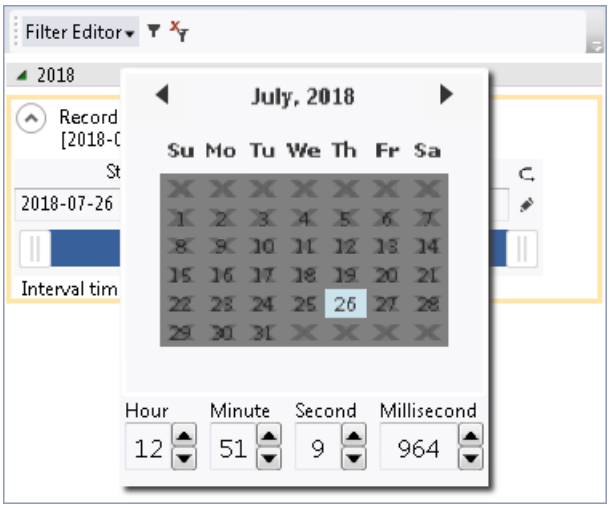
1. まず、Target Browserの左隅にある**TcAnalytics**をクリックします。ブローカーが設定され、様々なデバイスからのライブデータと履歴データが表示されます。下図のように表示されます。



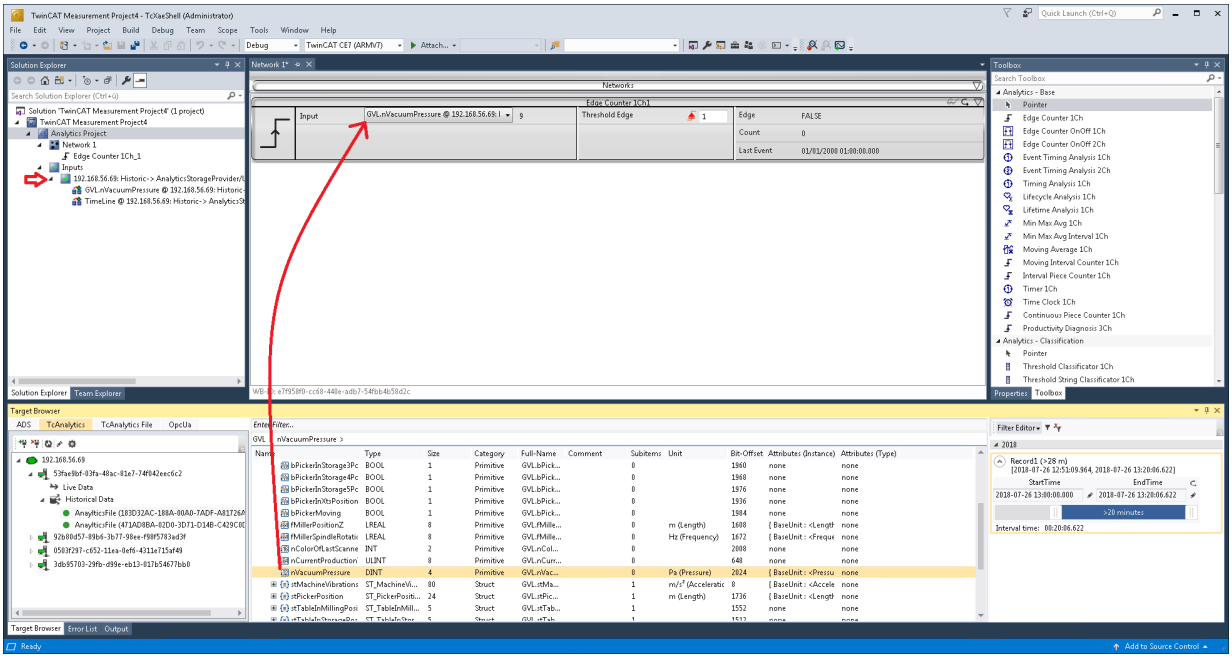
2. 作成した履歴ストリームに移動し、解析対象の記録を選択します。全ての記録は右側の**Record**ウィンドウに表示されます。デフォルトでは最後の記録が選択されています。



3. ライブ記録の場合、記録の時間範囲は数秒ごとに更新されます。デフォルトでは、記録の全時間範囲が使用されます。また、開始時間と終了時間を編集して、必要なデータ範囲を解析することもできます。これは、スライダー、テキストフィールド、またはグラフィカルカレンダー表示で設定できます。テキストフィールドの右側にあるマークをクリックすると、カレンダーが表示されます。



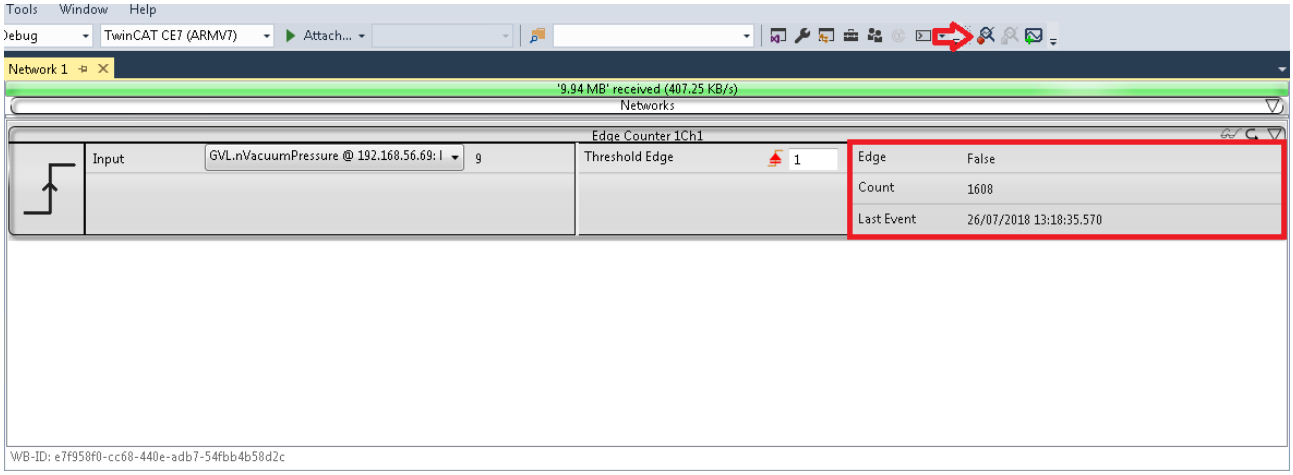
4. これらの手順の後、シンボルをアルゴリズムの入力にドラッグ＆ドロップできます。ライブデータのシンボルと同様です。



⇨ 履歴データを基に新しい入力ソースが生成され、Visual Studio®のSolution Explorerに表示されます。まず、ドラッグされたシンボルと、現在のデバイス時間のタイムスタンプが、データの下にリストされます。また、このデータで新たに追加されたシンボルもリストアップされています。

Analytics コンフィグレータで履歴データを解析する

履歴データを解析するには、Start Analyticsボタンを押してください。ライブデータの解析とは対照的に、緑色の進捗バーが表示されます。解析速度は、記録データの長さ、シンボルの数とサイズ、ブローカーとのブロードバンド接続速度に依存します。進捗バーが終了すると、解析は自動的に停止します。結果は表示されたままになります。



6.8 ランタイムへのデプロイ

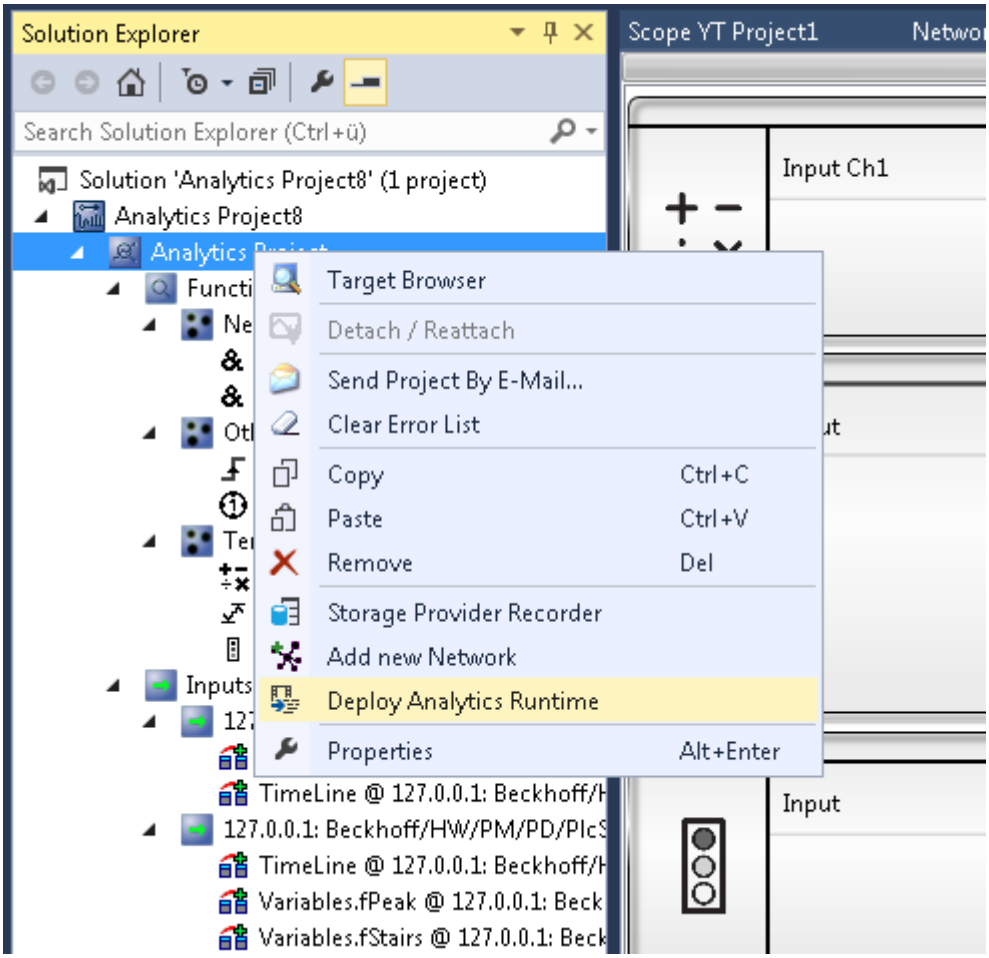
PLCコードは、TwinCAT Analytics Workbench コンフィグレータで設定した全てのモジュールとパラメータで生成できます。このコードをTwinCAT Analyticsランタイムにダウンロードすることで、24時間365日のデータ解析を実現できます。

注記

自動生成されたPLCコードの互換性

自動生成されるPLCコードは、TwinCAT Analytics ライブラリに基づいています。ライブラリの基本モデルのインターフェースは、リリース以降も旧バージョンとの互換性があります。自動生成されたコード自体は、あくまでもサンプルコードに過ぎません。このコード生成はバージョンごとに変更される可能性があります。可能な限り、コード生成バージョンによってこの問題は解決されています。

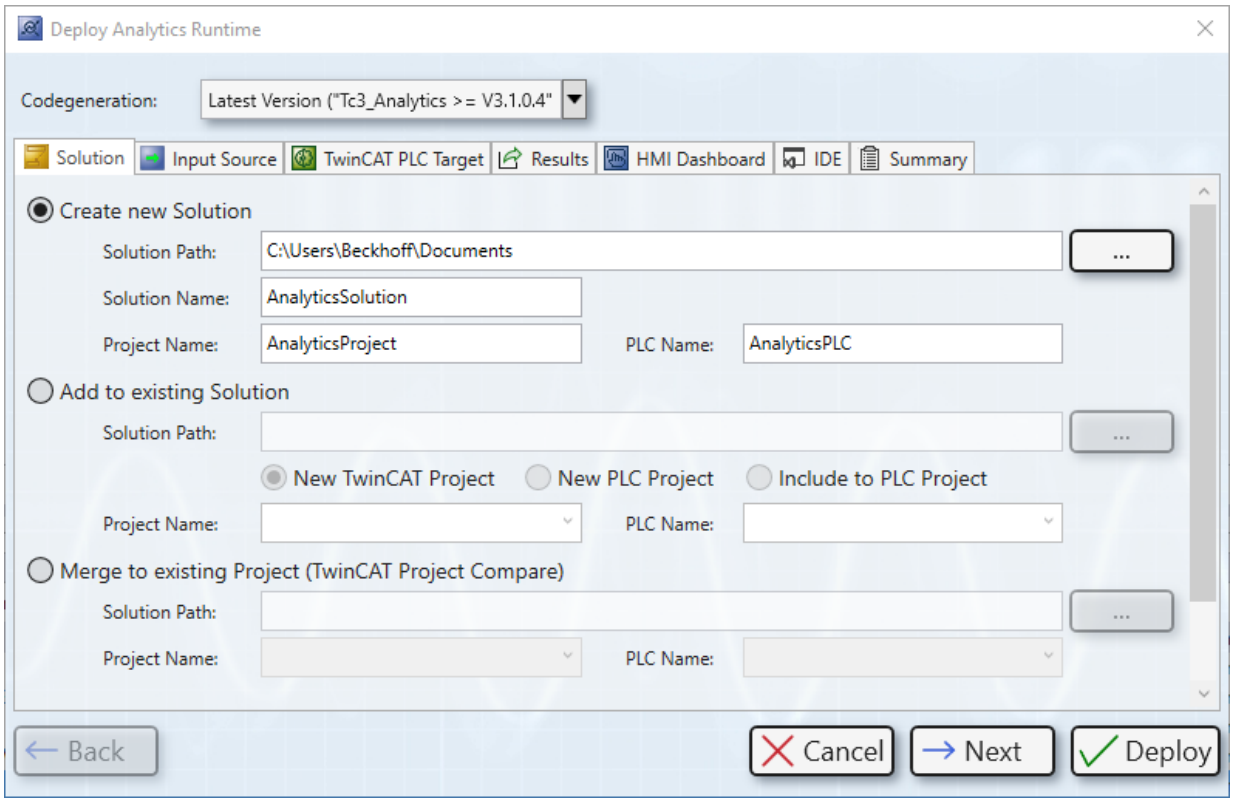
✓ 設定後、コンテキストメニューの**Deploy Analytics Runtime**コマンドをクリックします。



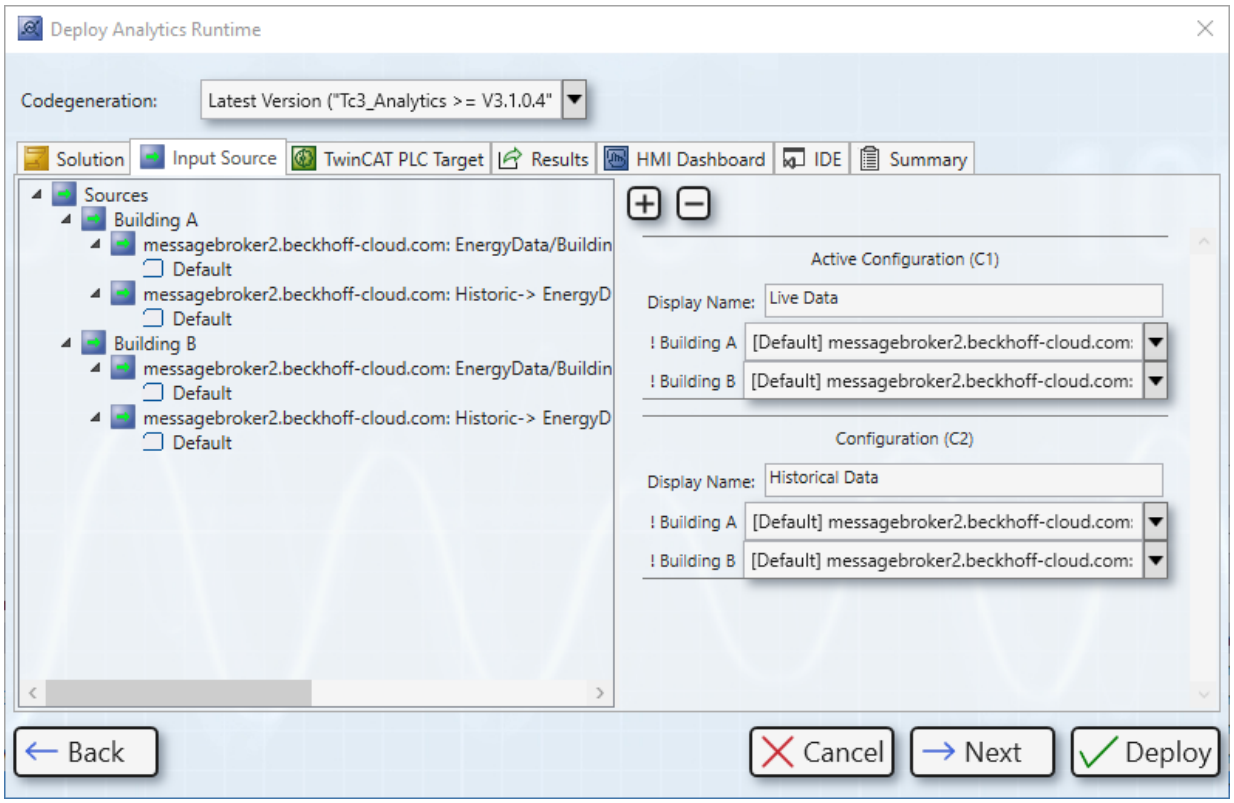
✓ Deployウィザードが起動し、必要な設定を段階的に設定して使用できます。

1. 最初のタブ**Solution**では、新規ソリューションを作成するか（**Create new Solution**）、解析用PLCコードを既存のソリューションに新しいプロジェクト、新しいPLC、または既存のPLCとして統合または追加するか（**Add to existing Solution**）、あるいはTwinCATプロジェクト比較ツールを使用し

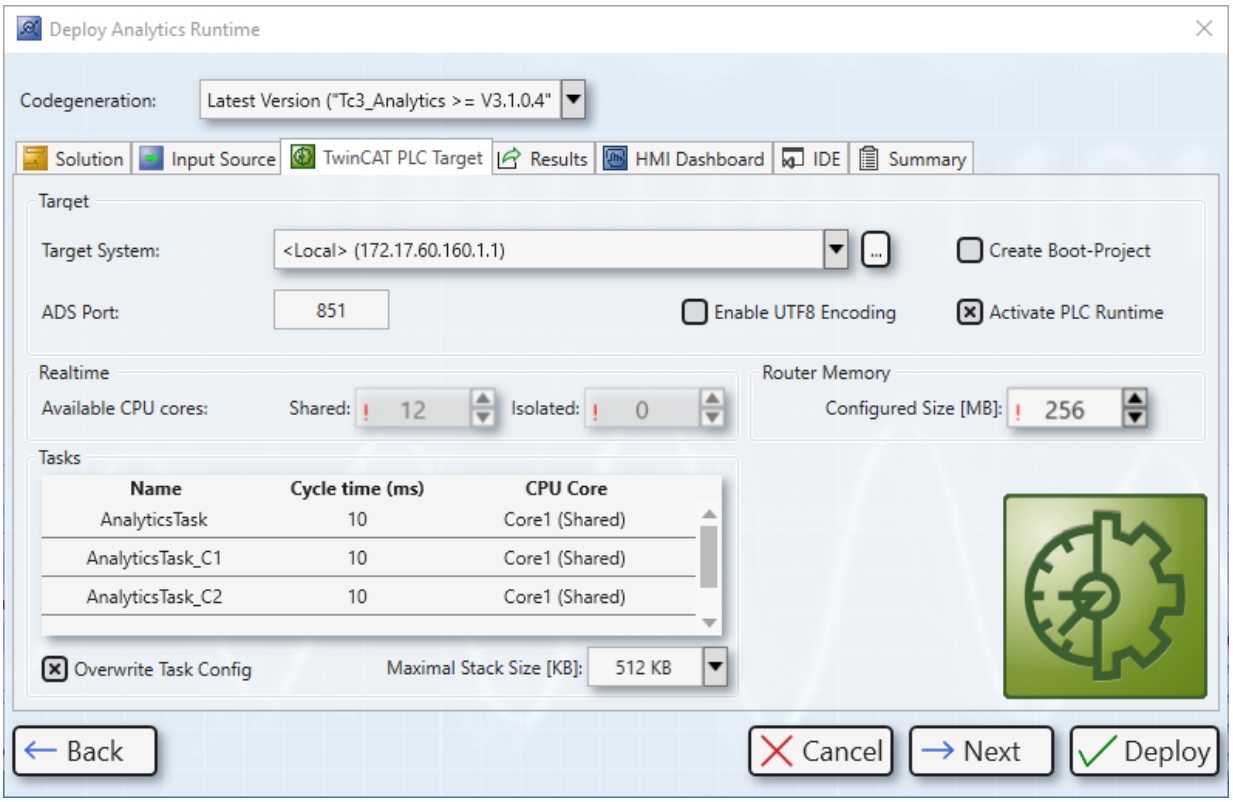
て、新しいプロジェクトを既存のソリューションにマージするか（**Merge to existing Project**）を選択します。



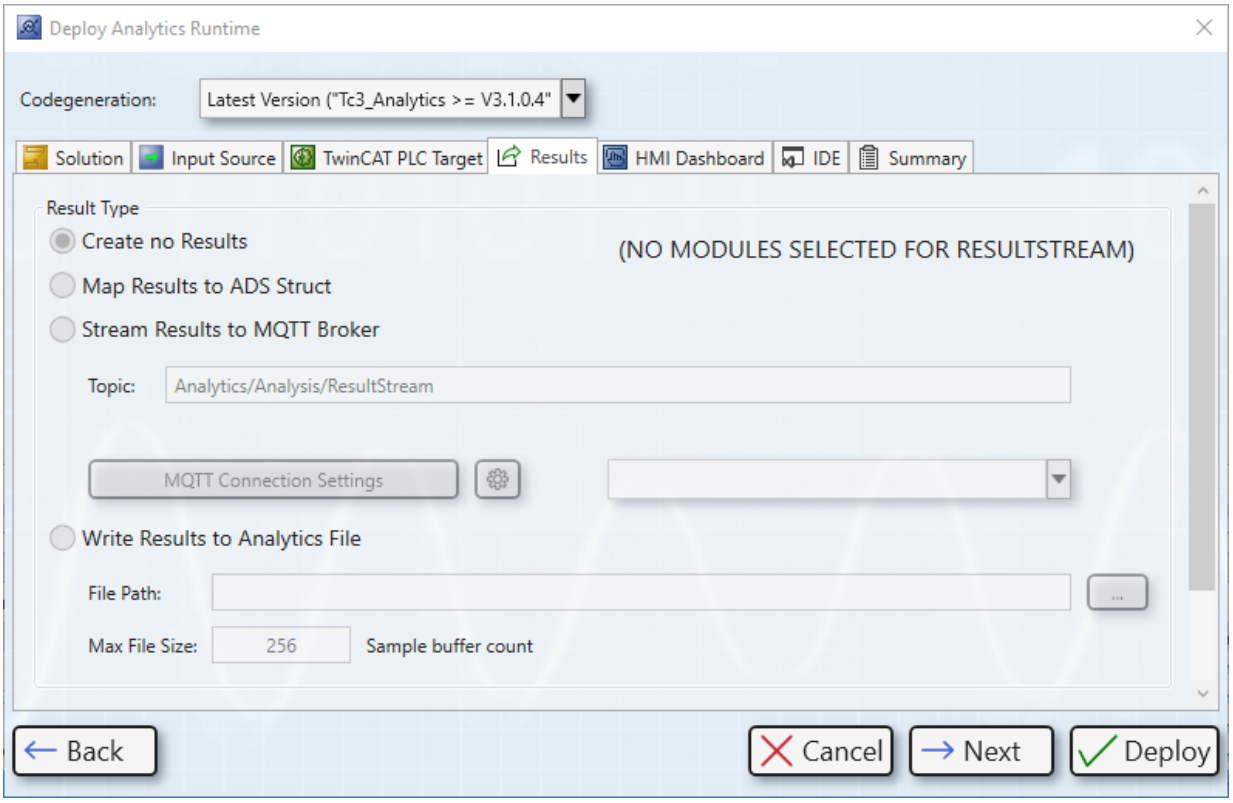
2. 2番目のタブ**Input Source**で複数のデータソース設定を選択します。これは例えば、ライブデータと履歴データの両方を解析する場合に必要です。



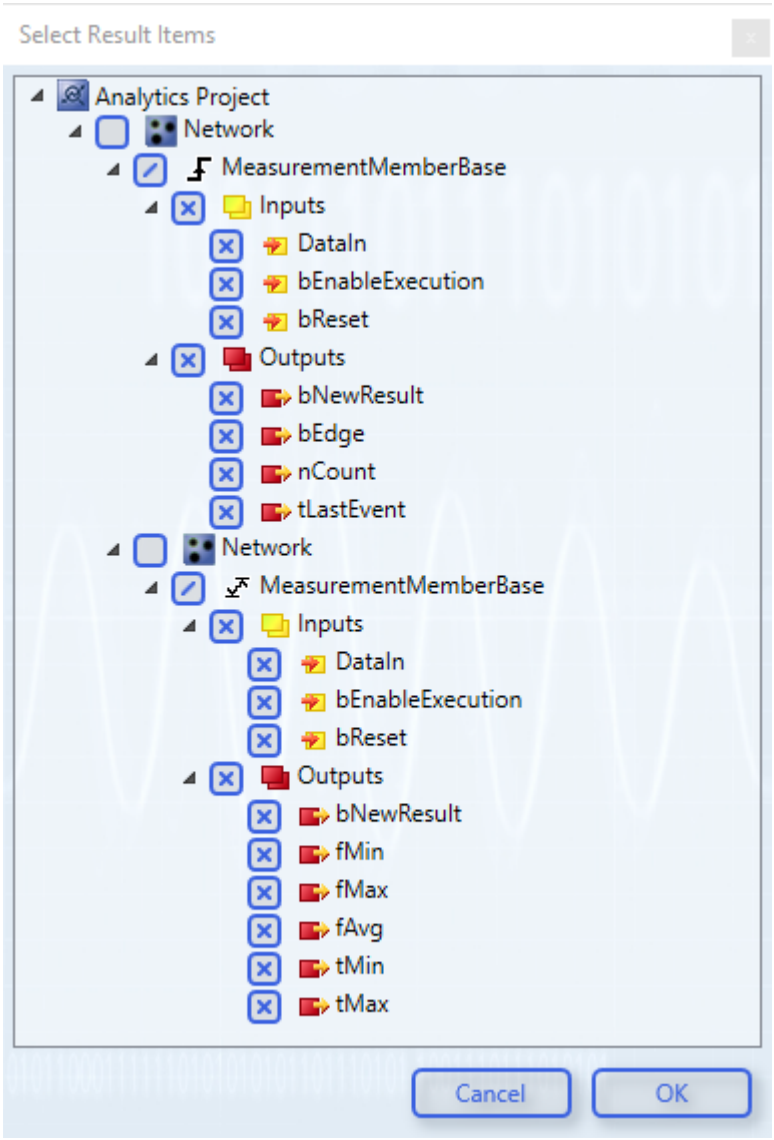
3. 3番目のタブ**TwinCAT PLC Target**では、ターゲットシステム、タスクサイクルタイム、対応するCPUコアへのタスク割り当てなど、PLC固有のパラメータを指定できます。



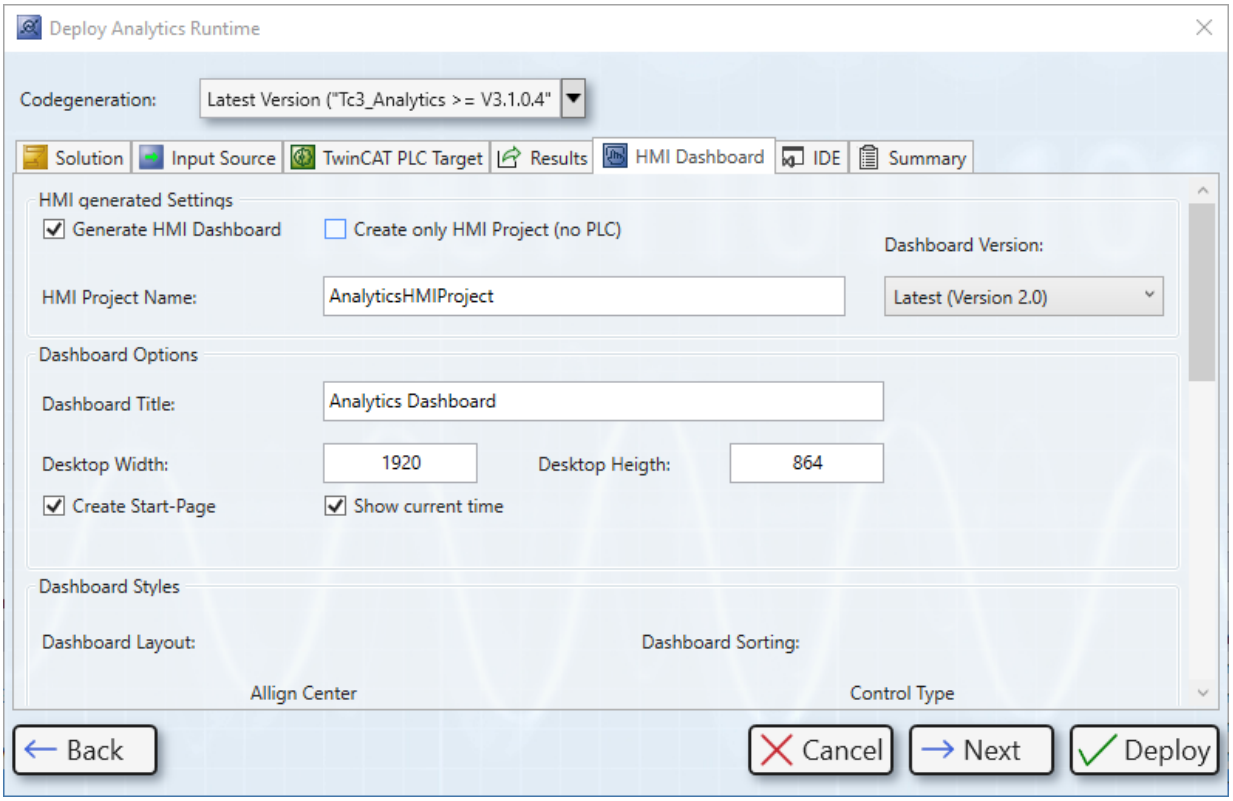
4. コンフィグレータで一部のファンクションの**PLC Result**プロパティを設定すると、**Result**タブが有効になります。
結果をストリーミングまたは保存する場所を指定します。



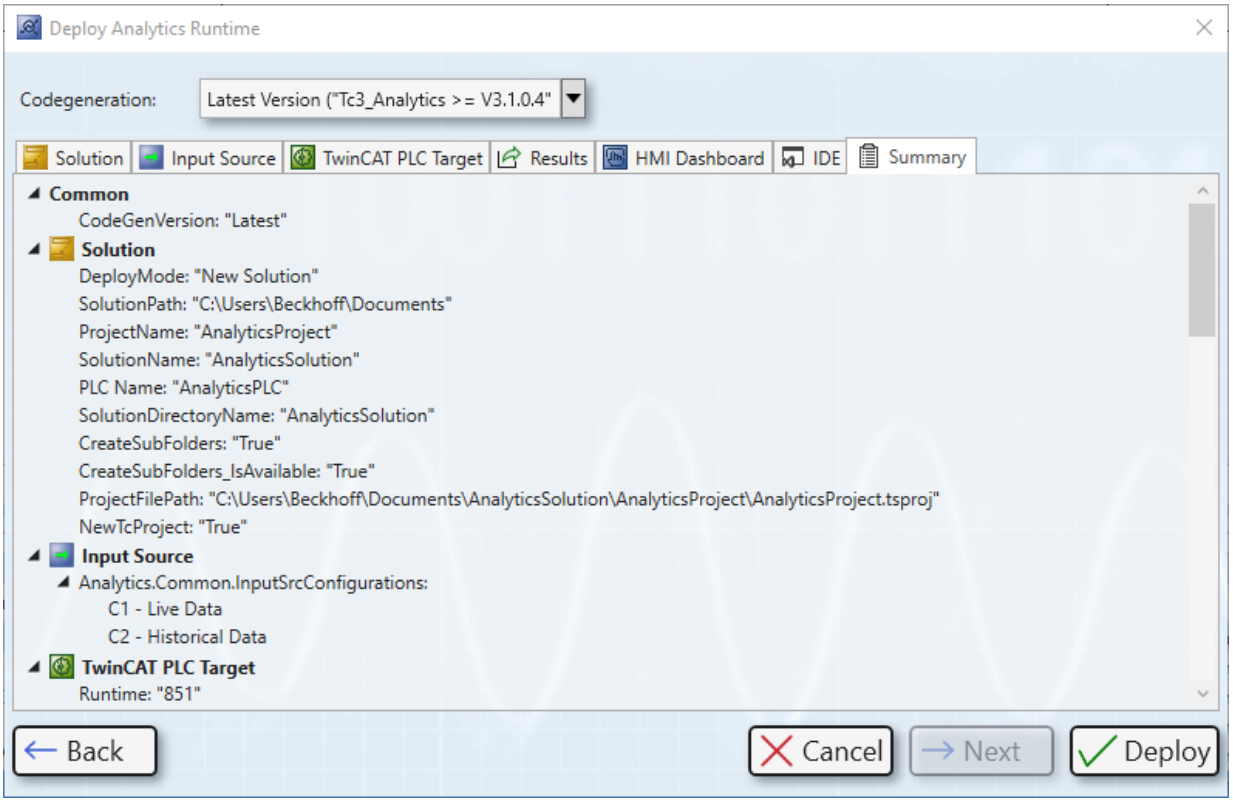
5. **Select Result Items**をクリックすると、必要な値だけを選択できます。



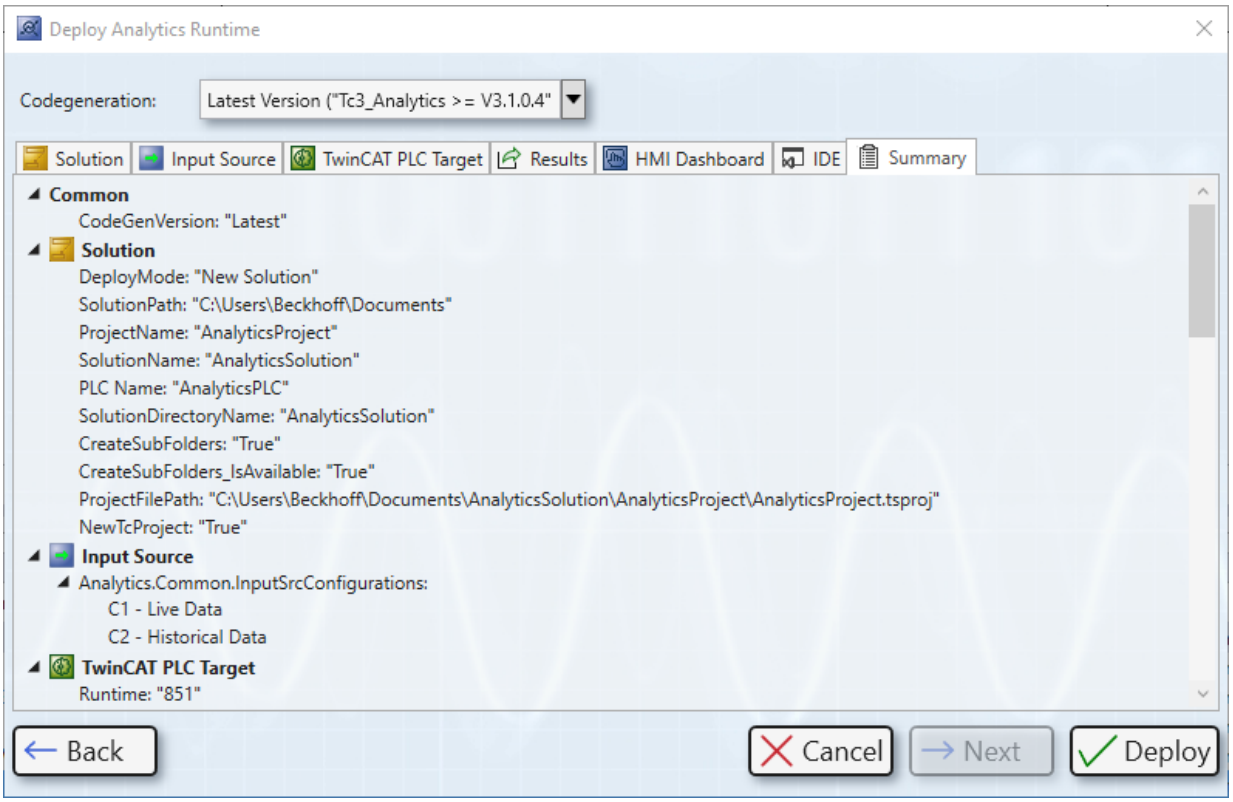
6. HMIコントロールを1つ、または複数の機能に割り当てている場合、**HMI Dashboard**タブが有効になります。ここで様々な設定を行い、ニーズに合わせてカスタマイズしたダッシュボードを作成できます。



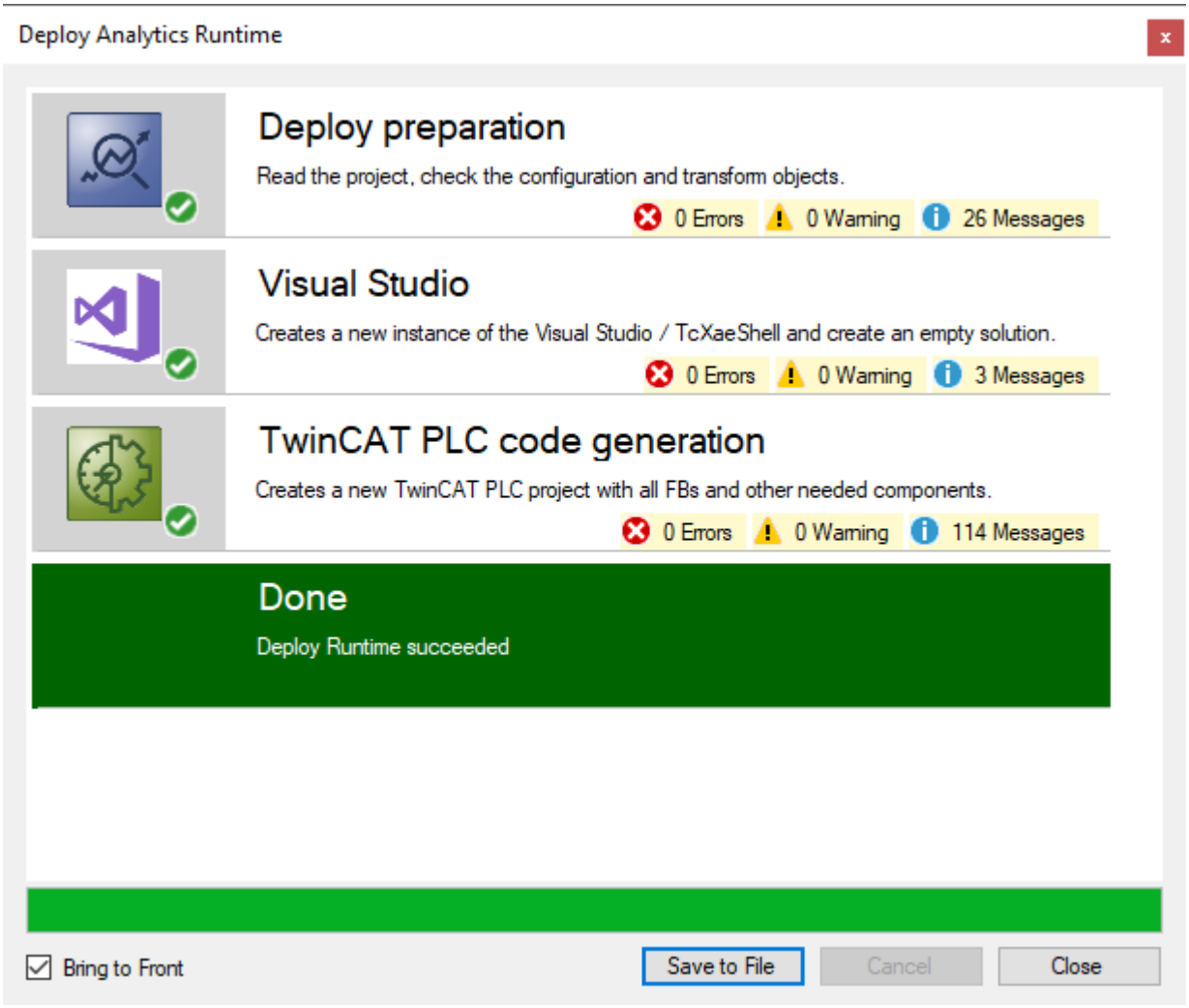
7. **IDE**タブでは、複数のバージョンがインストールされている場合に、生成に使用するVisual Studio®バージョンまたはTwinCAT XAE Shellを選択します。



8. **Summary**タブには、生成のための全ての設定が表示されます。**Deploy**をクリックして生成プロセスを開始します。

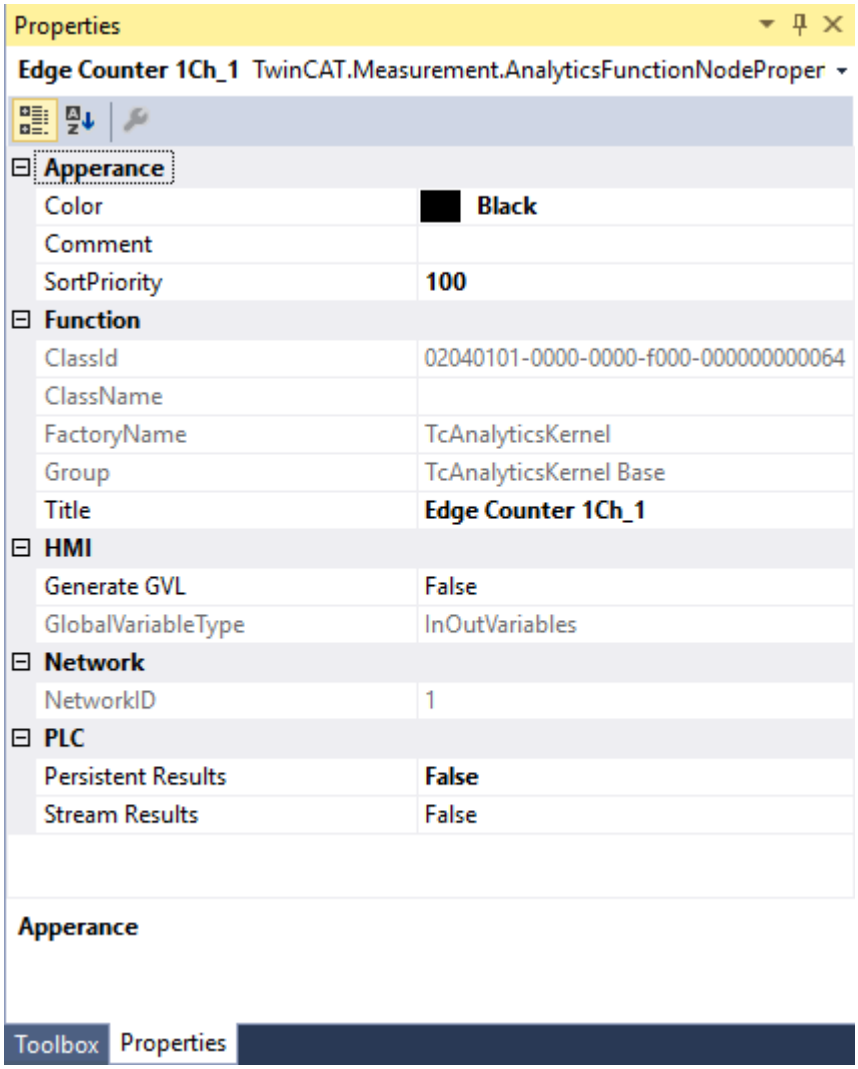


⇨ 概要ウィンドウには、生成プロセス中の各ステップが、カテゴリ別にわかりやすく表示されます。



6.8.1 アルゴリズムの特性

Analytics コンフィグレータの各アルゴリズムには、様々な特性があります。HMIとPLCのセクションは、コードの自動生成に必要です。



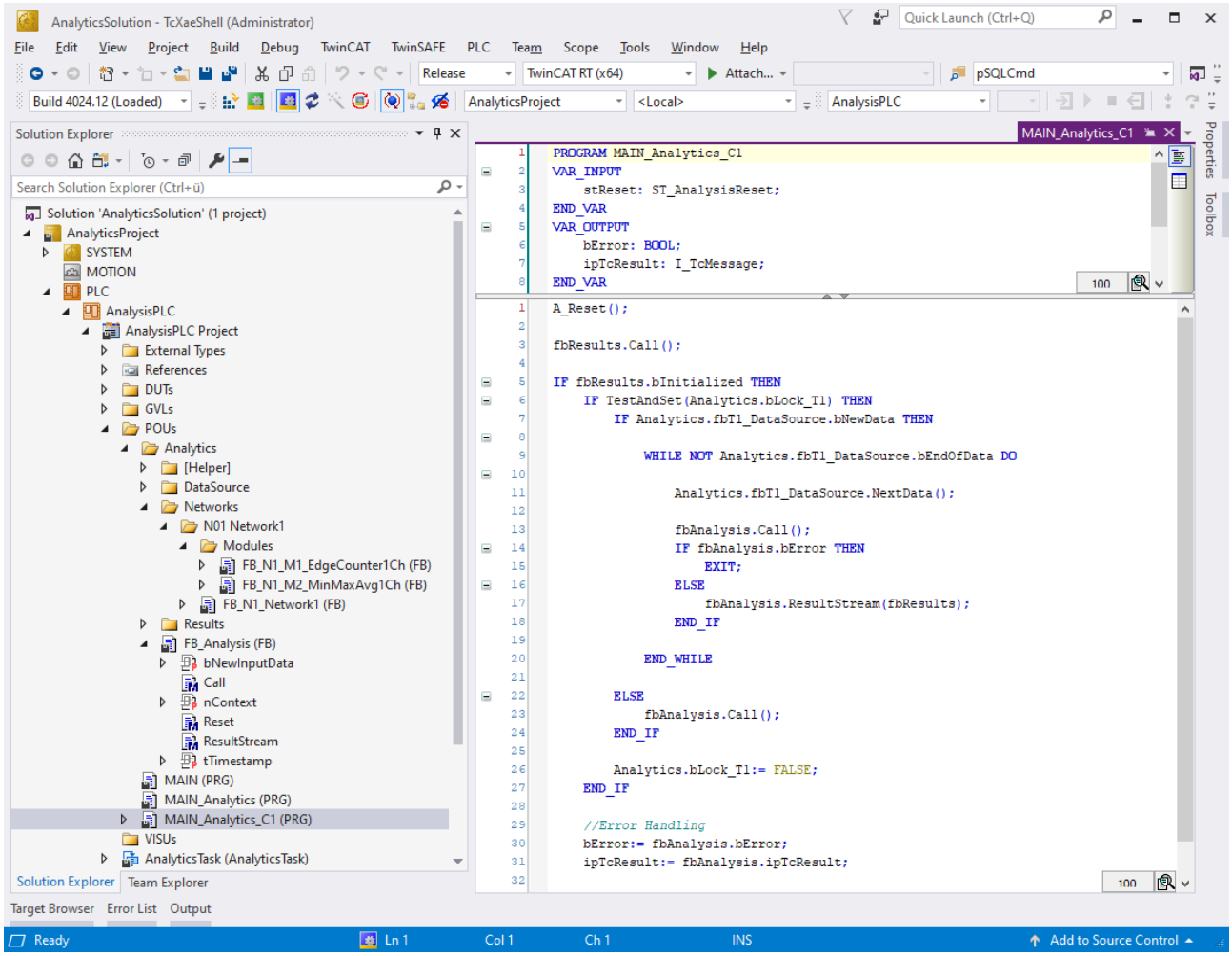
HMI

- **Generate GVL:**TwinCAT HMI用の変数と対応するデータ型のマッピングを含む、グローバル変数リストの生成を可能にします。
- **GlobalVariableType :** アルゴリズムの入出力のみを対象とする場合は InOutVariables を、テーブルへの一般的な STRING マッピングを行う場合は KeyValuePairs を選択します。
-

PLC

- **Persistent Results:**アルゴリズムの結果をAnalyticsランタイムのターゲットシステムに永続的に保存するには、このフラグを有効にします。
- **Stream Results :** このフラグを有効にすると、アルゴリズムの入出力がコード生成による結果ストリームに追加されます。

6.8.2 PLCコード



注記

自動生成されたPLCコードの互換性

自動生成されるPLCコードは、TwinCAT Analytics ライブラリに基づいています。ライブラリの基本モデルのインターフェースは、リリース以降も旧バージョンとの互換性があります。自動生成されたコード自体は、あくまでもサンプルコードに過ぎません。このコード生成はバージョンごとに変更される可能性があります。

コード生成バージョンの互換性

	バージョン 2.1（廃止）	バージョン 3.0（廃止）	バージョン 4.0	バージョン 4.1	バージョン 5.0
Analyticsアルゴリズム	(X)	(X)	X	X	X
フィルタアルゴリズム	(X)	(X)	X	X	X
HMIサポート	(X)	(X)	X	X	X
高度な入力データ処理機能を備えたHMI	-	(X)	X	X	X
配列入力をサポート	(X)	(X)	X	X	X
オーバーサンプリング入力をサポート	-	-	X	X	X
ネットワークテンプレート	(X)	(X)	X	X	X
「Closed」ネットワークテンプレート	-	-	X	X	X
入力、出力、パラメータを持つネットワーク	-	-	X	X	X

	バージョン 2.1（廃止）	バージョン 3.0（廃止）	バージョン 4.0	バージョン 4.1	バージョン 5.0
「Condition Monitoring」ネットワークテンプレート	-	-	X	X	X
ラムダアルゴリズム	-	-	X	X	X
HMIによるパラメータの書き込み	-	-	-	X	X
「Power Monitoring」アルゴリズム	-	-	X	X	X
スコープサポート	-	-	-	-	X

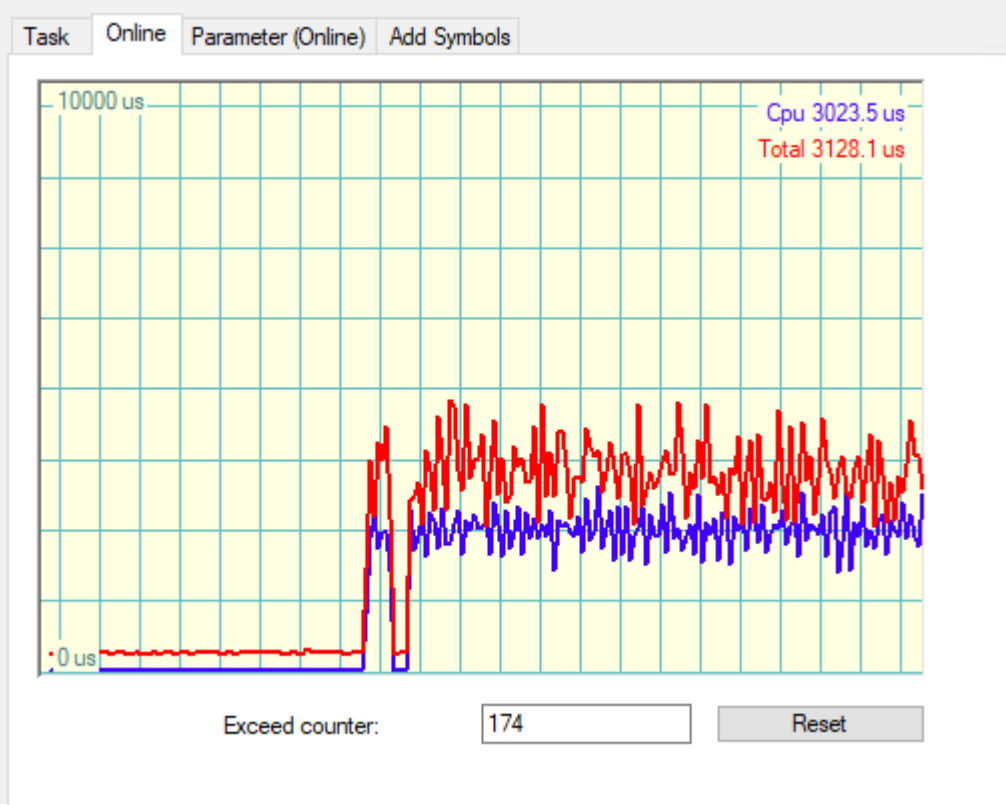
注記

PLC コードの手動による最適化の可能性

自動生成される PLC プロジェクトにはそれぞれ固有の特性があり、スムーズでエラーのない実行のための要件があります。これらの特性の中には、生成時に予測できないものもあります。生成コードを手動で最適化できる箇所は、Manual optimizations [▶ 367]にリストアップされています。

6.8.2.1 手動による最適化

履歴データの処理におけるサイクルオーバーラン

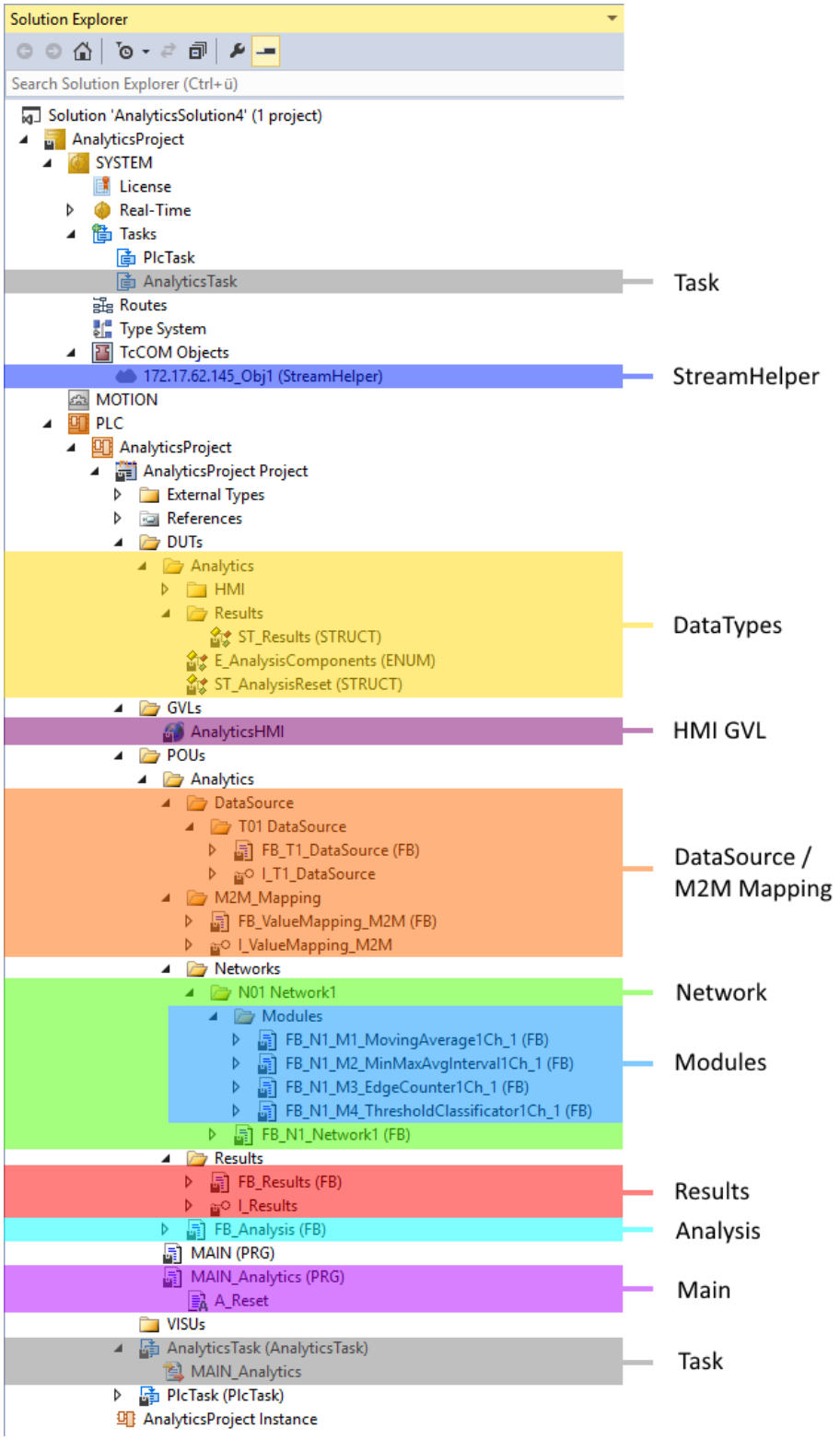


これは、Analytics.GVLの以下のパラメータを使用して最適化できます。

```
{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
```

- パラメータ nHistMaxInputBuffer は、1サイクルで処理するサンプル数の最大値を指定します。この値が高いほど、サイクルオーバーランのリスクが高くなります。この値が小さすぎると、StreamHelperバッファがオーバーフローし、データが失われる可能性があります。
- パラメータnHistMaxSampleCountは、ASPからのメッセージが格納できるサンプル数を指定します。値が高いほど、履歴データの転送が速くなります。ただし、これによりStreamHelperバッファのオーバーフローを早期に引き起こす可能性があります。

6.8.2.2 Code version 2.1



Task:

解析用に各タスクが作成されます。

StreamHelper :

1つ以上のデータソースがMQTTバイナリストリーム型である場合、コード生成は、受信するバイナリストリームパターンを処理するために、StreamHelperオブジェクトのインスタンスを作成します。

DataTypes :

データ型は解析用に作成されます。これらには、リセット機能や結果処理のためのSTRUCT、および各種コンポーネントを選択するためのENUMが含まれています。

HMI GVL :

モジュールの入出力を HMI ダッシュボードに簡単にマッピングするために、選択した変数がグローバル変数として生成されます。

DataSource/M2M Mapping :

FB DataSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。FB ValueMapping_M2Mは、INPUTモジュールとOUTPUTモジュール間の値のマッピング（M2M - Module to Module）を管理します。

Network :

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

Modules :

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

Results:

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルに保存します。

Analysis :

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

MAIN :

Analytics FBはMAIN_Analyticsプログラムで呼び出されます。プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics バージョン 3.1.0.0以上

6.8.2.2.1 FB_DataSource

FB_DataSourceは、異なるソースの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力があります。

構文

定義:

```
FUNCTION_BLOCK FB_T[n]_DataSource IMPLEMENTS I_T[n]_DataSource
VAR
END_VAR
```

メソッド

名前	定義場所	説明
Call [▶ 371]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
GetData [▶ 371]	ローカル	指定した要素のデータを取得するメソッド

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.2.1.1 GetData

構文

```
METHOD GetData : BOOL
VAR_INPUT
    nElement : UDINT;
END_VAR
```

入力

名前	データ型	説明
nElement	UDINT	特定のサンプルを取得するための要素ID

戻り値

名前	データ型	説明
GetData	BOOL	新しい要素が選択された場合にTRUE

6.8.2.2.1.2 Call

構文

```
METHOD Call : BOOL
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.2.2 FB_Network

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に整理されます。

構文

定義:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_INPUT
    [module FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

入力

名前	データ型	説明
Module FBs		設定されたモジュールのFB

出力

名前	データ型	説明
bError	BOOL	エラーが発生するとすぐに TRUE になります。
ipTcResult	I_TcMessage	戻り値の詳細を提示するTwinCAT 3 EventLoggerのメッセージインターフェース。

メソッド

名前	定義場所	説明
Call [▶ 376]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 377]	ローカル	全てのサブモジュールでネットワークをリセットします。
ValueMapping [▶ 372]	ローカル	入力値を各種モジュール入力にマッピングします。
SetHMIValues [▶ 373]	ローカル	オプション：モジュールの入出力をグローバルHMI変数にマッピングします。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.2.2.1 ValueMapping

構文

```
METHOD ValueMapping : BOOL
VAR_INPUT
    ipDataSource : I_T[n]_DataSource;
END_VAR
```

入力

名前	データ型	説明
ipDataSource	I_T[n]_DataSource	特定のデータソースからの解析用データ

戻り値

名前	データ型	説明
ValueMapping	BOOL	

6.8.2.2.2.2 SetHMIValues

構文

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバルHMI構造体へのポインタ

戻り値

名前	データ型	説明
SetHMIValues	BOOL	完了した場合にTRUE

6.8.2.2.2.3 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.2.2.4 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipDataSource: I_T[n]_DataSource;
    [ipValueMapping_M2M: I_ValueMapping_M2M;]
END_VAR
```

入力

名前	データ型	説明
ipDataSource	I_T[n]_DataSource	解析のためのデータ
ipValueMapping_M2M	I_ValueMapping_M2M	オプション：モジュール間の値のマッピングに必要

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.2.3 FB_Module

FBモジュールには、Workbenchコンフィグレータで設定したモジュールの全ての入出力が含まれます。また、実行時にモジュールを再設定することも可能です。パラメータを変更し、bReconfigure INPUTの立ち上がりエッジで再設定プロセスを開始するだけです。

構文

定義:

```
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
```

入力

名前	データ型	説明
Module inputs		選択したモジュールの入力

出力

名前	データ型	説明
bError	BOOL	エラーが発生するとすぐに TRUE になります。
ipTcResult	I_TcMessage	戻り値の詳細を提示するTwinCAT 3 EventLoggerのメッセージインターフェース。
Module outputs		選択したモジュールの出力

メソッド

名前	定義場所	説明
Call [▶ 375]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 375]	ローカル	モジュールをリセットします。
SetHMI [▶ 375]	ローカル	オプション：入出力をグローバルHMI変数に設定します。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.2.3.1 Call

構文

```
METHOD Call : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.2.3.2 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.2.3.3 SetHMI

構文

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバルHMI構造体へのポインタ

戻り値

名前	データ型	説明
SetHMI	BOOL	完了した場合にTRUE

6.8.2.2.4 FB_Analysis

FB_Analysisでは、解析ルーチン全体が定義されます。モジュールとエラー処理が設定された全てのネットワークが作成されます。

構文

定義:

```
FUNCTION_BLOCK FB_Analysis
VAR_INPUT
    [network FBs]
END_VAR
VAR_OUTPUT
```

```
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

入力

名前	データ型	説明
Network FBs		設定されたネットワークのFB

出力

名前	データ型	説明
bError	BOOL	エラーが発生するとすぐに TRUE になります。
ipTcResult	I_TcMessage	戻り値の詳細を提示するTwinCAT 3 EventLoggerのメッセージインターフェース。

メソッド

名前	定義場所	説明
Call [▶ 376]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 377]	ローカル	解析全体をリセットします。
ResultStream [▶ 377]	ローカル	オプション：結果データを作成する必要がある場合

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.2.4.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipDataSource: I_T[n]_DataSource;
END_VAR
```

入力

名前	データ型	説明
ipDataSource	I_T[n]_DataSource	解析のためのデータ。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.2.4.2 Reset

構文

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

入力

名前	データ型	説明
stReset	ST_AnalysisReset	どのモジュールまたはネットワークをリセットするかを定義する構造体。

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.2.4.3 ResultStream

構文

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

入力

名前	データ型	説明
ipResults	I_Results	Result FB へのインターフェースポインタ

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.2.5 FB_Results

解析結果を保存またはストリーミングする必要がある場合は、Result FBがこれを管理し、選択した変数をメッセージブローカーにストリーミングするか、解析バイナリファイルにデータを保存します。

構文

定義:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    stResults: ST_Results;
END_VAR
```

出力

名前	データ型	説明
stResults	ST_Results	結果データの全アイテムを含むResult構造体

メソッド

名前	定義場所	説明
Call [▶ 378]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
AddResult [▶ 378]	ローカル	結果データにサンプルを追加します。
SendResults [▶ 379]	ローカル	結果データのバッファされたサンプルを全て送信します。
Release [▶ 379]	ローカル	結果データのストリームまたはファイルを閉じます。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.2.5.1 Call

構文

```
METHOD Call : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.2.5.2 結果の追加

構文

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

入力

名前	データ型	説明
tTimestamp	ULINT	サンプルのタイムスタンプ
stSample	ST_Results	サンプル構造

戻り値

名前	データ型	説明
AddResult	BOOL	

6.8.2.2.5.3 SendResults

構文

```
METHOD SendResults : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
SendResults	BOOL	

6.8.2.2.5.4 Release

構文

```
METHOD Release : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Release	BOOL	

6.8.2.2.6 MAIN_Analytics

MAIN_Analyticsプログラムでは、Analysis FBが呼び出されます。プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットしたいコンポーネントを選択します。次にbReset INPUTの立ち上がりエッジがリセット処理を開始します。

A_Reset Actionの内部では、全てのリセットコールが定義されています。

構文

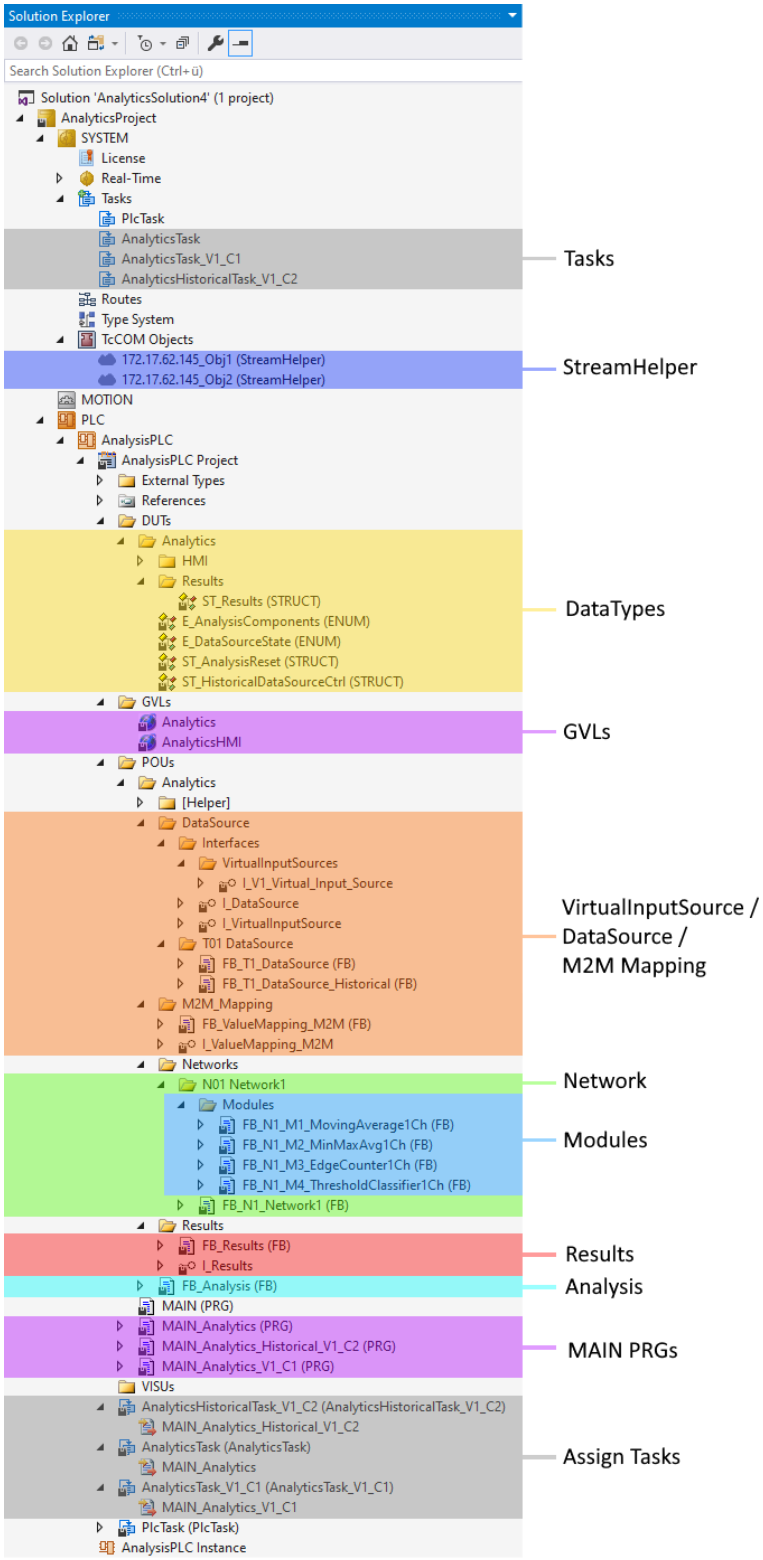
定義:

```
PROGRAM MAIN_Analytics
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3 Code version 3.0



Tasks:

解析とVirtual Input Sourceの各設定ごとに、個別のタスクが作成されます。

StreamHelper :

MQTTバイナリストリームの各データソースに対して、コード生成はストリームヘルパーオブジェクトのインスタンスを作成し、受信するバイナリデータのパターンを処理します。

DataTypes :

データ型は解析用に作成されます。これらには、リセット機能や結果処理のためのSTRUCT、および各種コンポーネントを選択するためのENUMが含まれています。

GVLs:

モジュールの入出力を HMI ダッシュボードに簡単にマッピングするために、選択した変数がグローバル変数として生成されます。さらに、Data Sourceファンクションブロックインスタンスと各種パラメータがグローバル変数として生成されます。

VirtualInputSource / DataSource / M2M Mapping :

Virtual Input Sourceインターフェースは、解析データからData Sourceシンボルを抽象化します。FB DataSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。FB ValueMapping_M2Mは、INPUTモジュールとOUTPUTモジュール間の値のマッピング（M2M - Module to Module）を管理します。

Network :

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

Modules :

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

Results:

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をバイナリまたはJson形式でメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルにローカルに保存します。

解析 :

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

MAIN PRGs:

MAIN_Analyticsプログラムでは、DataSource FBが呼び出され、リセット機能が管理され、必要に応じて値がHMIダッシュボードにマッピングされます。プログラムは独立したタスクに割り当てられます。

FB解析は、MAIN_Analytics_Vx_Cxプログラムで呼び出されます。各プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics バージョン 3.1.0.0以上

6.8.2.3.1 FB_DataSource

FB DataSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。

構文

定義:

```
FUNCTION_BLOCK FB_T[n]_DataSource IMPLEMENTS I_DataSource, I_V[n]_Virtual_Input_Source
VAR_OUTPUT
eDataState: E_DataSourceState;
END_VAR
```

メソッド

名前	定義場所	説明
Call [▶ 383]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
GetData [▶ 382]	ローカル	指定した要素のデータを取得するメソッド。
GetDataOversampling [▶ 383]	ローカル	指定した要素のオーバーサンプリングデータを取得するメソッド。
NewDataAvailable [▶ 383]	ローカル	新しいデータが利用可能かどうかをチェックするメソッド。
HistoricalCtrl [▶ 384]	ローカル	履歴データを取得するためのメソッド。
UpdateRecordList [▶ 384]	ローカル	履歴データリストを更新するためのメソッド。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3.1.1 GetData

構文

```
METHOD GetData : BOOL
VAR_INPUT
    nElement : UDINT;
END_VAR
```

入力

名前	データ型	説明
nElement	UDINT	具体的な例を得るためのElement ID。

戻り値

名前	データ型	説明
GetData	BOOL	新しい要素が選択された場合に TRUE

6.8.2.3.1.2 Call

構文

```
METHOD Call : BOOL
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.3.1.3 GetDataOversampling

構文

```
METHOD GetDataOversampling : BOOL
VAR_INPUT
nElement : UDINT;
nSample: : UDINT;
END_VAR
```

入力

名前	データ型	説明
nElement	UDINT	特定のサンプル要素を取得するためのElement ID。
nSample	UDINT	特定のサンプルを取得するためのSample ID。

戻り値

名前	データ型	説明
GetDataOversampling	BOOL	新しい要素が選択された場合に TRUE

6.8.2.3.1.4 NewDataAvailable

構文

```
METHOD NewDataAvailable : BOOL
VAR_INPUT
    nLastDataHandle : ULINT;
END_VAR
```

入力

名前	データ型	説明
nLastDataHandle	ULINT	最後に取得したデータパケットのハンドル。

戻り値

名前	データ型	説明
NewDataAvailable	BOOL	新しいデータが利用可能であればTRUE。

6.8.2.3.1.5 HistoricalCtrl

構文

```
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
```

入力

名前	データ型	説明
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
stRecordInfo	ST_HMI_DataSourceHistRecordInfo	

戻り値

名前	データ型	説明
HistoricalCtrl	BOOL	

6.8.2.3.1.6 UpdateRecordList

構文

```
METHOD UpdateRecordList : BOOL
VAR_INPUT
    stCtrl : REFERENCE TO ST_HMI_DataSourceCtrl;
    stHistStreamInfo : REFERENCE TO ST_HMI_DataSourceHist;
    sStreamSystemID : GUID;
END_VAR
```

入力

名前	データ型	説明
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
sStreamSystemID	GUID	

戻り値

名前	データ型	説明
UpdateRecordList	BOOL	

6.8.2.3.2 FB_Network

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

構文

定義:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_INPUT
    [module FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

入力

名前	データ型	説明
Module FBs		設定したモジュールのファンクションブロック。

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。

メソッド

名前	定義場所	説明
Call [▶ 389]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 390]	ローカル	全てのサブモジュールを使ってネットワークをリセットします。
ValueMapping [▶ 385]	ローカル	様々なモジュールに入力値を割り当てます。
SetHMIValues [▶ 386]	ローカル	オプション：モジュールの入出力をグローバルHMI変数にマッピングします。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3.2.1 値マッピング

構文

```
METHOD ValueMapping : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	特定のデータソースからの解析用データ。

戻り値

名前	データ型	説明
ValueMapping	BOOL	

6.8.2.3.2.2 SetHMIValues

構文

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバル HMI 構造体へのポインタ。

戻り値

名前	データ型	説明
SetHMIValues	BOOL	完了した場合にTRUE

6.8.2.3.2.3 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.3.2.4 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M: I_ValueMapping_M2M;]
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_T[n]_Virtual_Input_Source	解析のためのデータ。
ipValueMapping_M2M	I_ValueMapping_M2M	オプション：モジュール間の値のマッピングに必要です。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.3.3 FB_Module

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

構文

定義:

```
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
```

入力

名前	データ型	説明
Module inputs		選択したモジュールの入力。

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。
Module outputs		選択したモジュールの出力。

メソッド

名前	定義場所	説明
Call [▶ 387]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 388]	ローカル	モジュールをリセットします。
SetHMI [▶ 388]	ローカル	オプション：グローバル HMI 構造体への入出力を設定します。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3.3.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M : I_ValueMapping_M2M;]
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	解析のためのデータ。
ipValueMapping_M2M	I_ValueMapping_M2M	オプション：モジュール間の値のマッピングに必要です。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.3.3.2 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.3.3.3 SetHMI

構文

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバル HMI 構造体へのポインタ。

戻り値

名前	データ型	説明
SetHMI	BOOL	完了した場合にTRUE

6.8.2.3.4 FB_Analysis

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

構文

定義:

```
FUNCTION_BLOCK FB_Analysis
VAR_INPUT
    [network FBs]
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
```

入力

名前	データ型	説明
Network FBs		設定されたネットワークのファンクションブロック。

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。

メソッド

名前	定義場所	説明
Call [▶ 389]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 390]	ローカル	解析全体をリセットします。
ResultStream [▶ 390]	ローカル	オプション：結果ストリームを作成する必要がある場合。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3.4.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	解析のためのデータ。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.3.4.2 Reset

構文

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

入力

名前	データ型	説明
stReset	ST_AnalysisReset	どのモジュールまたはネットワークをリセットするかを定義する構造体。

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.3.4.3 ResultStream

構文

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

入力

名前	データ型	説明
ipResults	I_Results	FB Resultsへのインターフェースポインタ。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.3.5 FB_Results

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルに保存します。

構文

定義:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    stResults: ST_Results;
END_VAR
```

出力

名前	データ型	説明
stResults	ST_Results	結果データの全要素を含む構造体。

メソッド

名前	定義場所	説明
Call [▶ 391]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
AddResult [▶ 391]	ローカル	結果データにサンプルを追加します。
SendResults [▶ 391]	ローカル	結果データのバッファされたサンプルを全て送信します。
Release [▶ 392]	ローカル	結果データのストリームまたはファイルを閉じます。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3.5.1 Call

構文

```
METHOD Call : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.3.5.2 結果の追加

構文

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

入力

名前	データ型	説明
tTimestamp	ULINT	サンプルのタイムスタンプ
stSample	ST_Results	サンプルの構造体

戻り値

名前	データ型	説明
AddResult	BOOL	

6.8.2.3.5.3 SendResults

構文

```
METHOD SendResults : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
SendResults	BOOL	

6.8.2.3.5.4 Release

構文

```
METHOD Release : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Release	BOOL	

6.8.2.3.6 MAIN_Analytics

MAIN_Analyticsプログラムでは、DataSource FBが呼び出され、リセット機能が管理され、必要に応じて値がHMIダッシュボードにマッピングされます。プログラムは独立したタスクに割り当てられます。

構文

定義:

```
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
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.3.7 MAIN_Analytics_V[n]_C[n]

FB Analysisは、MAIN_Analytics_V[n]_C[n]プログラムで呼び出されます。プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

HMI値のマッピングはアクションA_MapToHMIで行われます。

構文

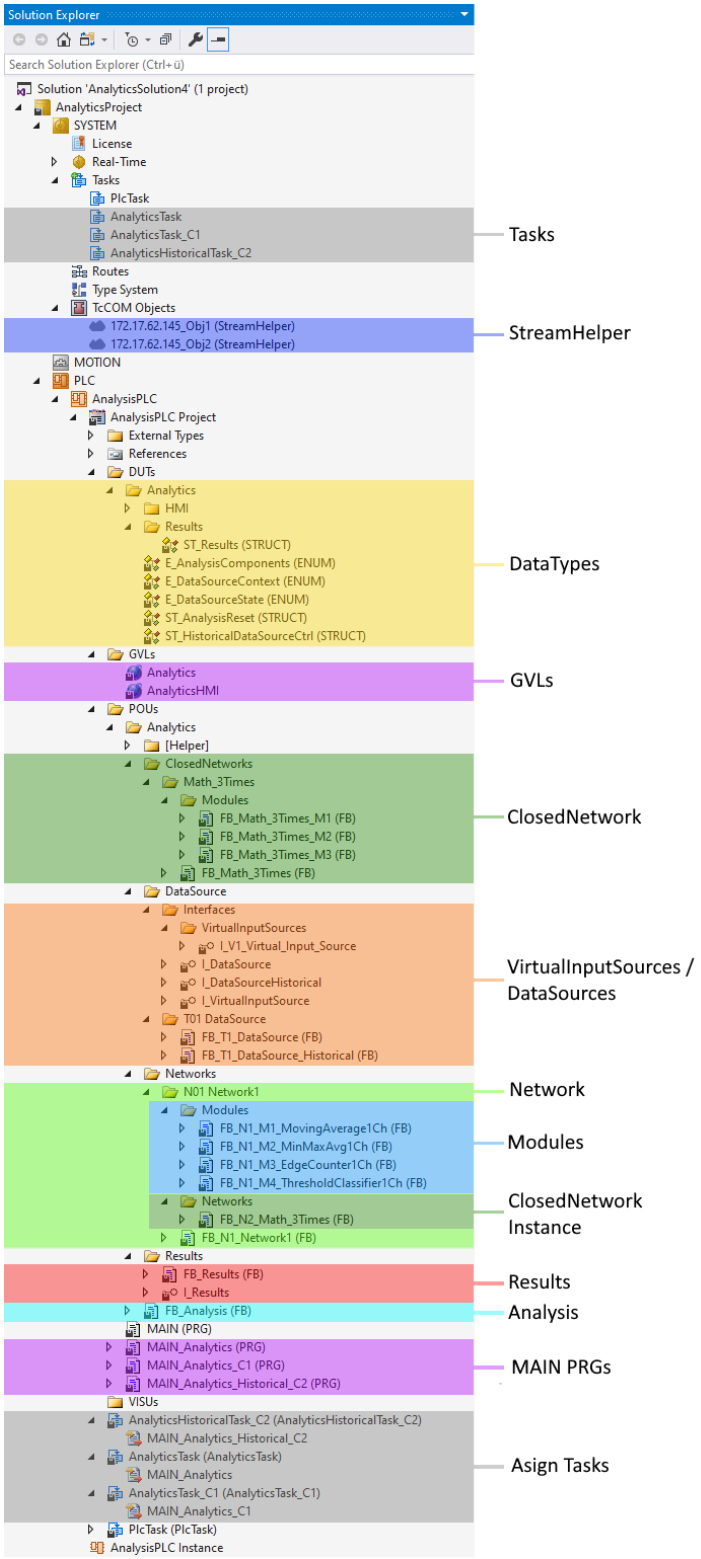
定義:

```
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
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4 Code version 4.x



Tasks:

解析とVirtual Input Sourceの各設定ごとに、個別のタスクが作成されます。

StreamHelper :

MQTTバイナリストリームの各データソースに対して、コード生成はストリームヘルパーオブジェクトのインスタンスを作成し、受信するバイナリデータのパターンを処理します。

DataTypes :

データ型は解析用に作成されます。これらには、リセット機能や結果処理のためのSTRUCT、および各種コンポーネントを選択するためのENUMが含まれています。

GVLs:

モジュールの入出力を HMI ダッシュボードに簡単にマッピングするために、選択した変数がグローバル変数として生成されます。さらに、Data Sourceファンクションブロックインスタンスと各種パラメータがグローバル変数として生成されます。

ClosedNetwork:

ClosedNetwork FBは、全てのサブネットワークとモジュールを含めて一度だけ生成されます。これらは、解析の中で複数回インスタンス化できます。こうすることで、生成されるコードを減らし、単純化できます。

VirtualInputSource / DataSource:

VirtualInputSource インターフェースは、解析データから DataSource シンボルを抽象化します。FB DataSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。

Network :

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

Modules :

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

ClosedNetwork Instance:

このFBでは、対応するClosedNetworkが解析のためにインスタンス化されます。この場合、内部で使用されるモジュールは、FBモジュールとして生成されなくなります。

Results:

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をバイナリまたはJson形式でメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルにローカルに保存します。

解析 :

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

MAIN PRGs:

MAIN_Analyticsプログラムでは、DataSource FBが呼び出され、リセット機能が管理され、必要に応じて値がHMIダッシュボードにマッピングされます。プログラムは独立したタスクに割り当てられます。

FB解析は、MAIN_Analytics_Vx_Cxプログラムで呼び出されます。各プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics バージョン 3.1.0.0以上

6.8.2.4.1 FB_DataSource

FB DataSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。

構文

定義:

```
FUNCTION_BLOCK FB_T[n]_DataSource IMPLEMENTS I_DataSource, I_V[n]_Virtual_Input_Source
VAR_OUTPUT
eDataState: E_DataSourceState;
END_VAR
```

メソッド

名前	定義場所	説明
Call [▶ 397]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
GetData [▶ 396]	ローカル	指定した要素のデータを取得するメソッド。
GetDataOversampling [▶ 397]	ローカル	指定した要素のオーバーサンプリングデータを取得するメソッド。
NewDataAvailable [▶ 397]	ローカル	新しいデータが利用可能かどうかをチェックするメソッド。
HistoricalCtrl [▶ 398]	ローカル	履歴データを取得するためのメソッド。
UpdateRecordList [▶ 398]	ローカル	履歴データリストを更新するためのメソッド。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4.1.1 GetData

構文

```
METHOD GetData : BOOL
VAR_INPUT
    nElement : UDINT;
END_VAR
```

入力

名前	データ型	説明
nElement	UDINT	特定のサンプルを取得するためのElement ID

戻り値

名前	データ型	説明
GetData	BOOL	新しい要素が選択された場合に TRUE

6.8.2.4.1.2 Call

構文

```
METHOD Call : BOOL
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.4.1.3 GetDataOversampling

構文

```
METHOD GetDataOversampling : BOOL
VAR_INPUT
nElement : UDINT;
nSample: : UDINT;
END_VAR
```

入力

名前	データ型	説明
nElement	UDINT	特定のサンプル要素を取得するためのElement ID
nSample	UDINT	特定のサンプルを取得するためのSample ID

戻り値

名前	データ型	説明
GetDataOversampling	BOOL	新しい要素が選択された場合に TRUE

6.8.2.4.1.4 NewDataAvailable

構文

```
METHOD NewDataAvailable : BOOL
VAR_INPUT
    nLastDataHandle : ULINT;
END_VAR
```

入力

名前	データ型	説明
nLastDataHandle	ULINT	最後に取得したデータパケットのハンドル

戻り値

名前	データ型	説明
NewDataAvailable	BOOL	新しいデータが利用可能であればTRUE

6.8.2.4.1.5 HistoricalCtrl

構文

```
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
```

入力

名前	データ型	説明
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
stRecordInfo	ST_HMI_DataSourceHistRecordInfo	

戻り値

名前	データ型	説明
HistoricalCtrl	BOOL	

6.8.2.4.1.6 UpdateRecordList

構文

```
METHOD UpdateRecordList : BOOL
VAR_INPUT
    stCtrl : REFERENCE TO ST_HMI_DataSourceCtrl;
    stHistStreamInfo : REFERENCE TO ST_HMI_DataSourceHist;
    sStreamSystemID : GUID;
END_VAR
```

入力

名前	データ型	説明
stCtrl	ST_HMI_DataSourceCtrl	
stHistStreamInfo	ST_HMI_DataSourceHist	
sStreamSystemID	GUID	

戻り値

名前	データ型	説明
UpdateRecordList	BOOL	

6.8.2.4.1.7 NextData

構文

```
METHOD NextData : BOOL
VAR_INPUT
END_VAR
```

6.8.2.4.1.8 NextDataOversample

構文

```
METHOD GetDataOversampling : BOOL
VAR_INPUT
nMaxOversampling : UDINT;
END_VAR
```

入力

名前	データ型	説明
nMaxOversampling	UDINT	オーバーサンプリング係数の最大値を指定します。

戻り値

名前	データ型	説明
NextDataOversample	BOOL	新しい要素が選択された場合に TRUE

6.8.2.4.2 FB_Network

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

構文

定義:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
VAR
    [module FBs]
END_VAR
```

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。

メソッド

名前	定義場所	説明
Call [▶ 389]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 390]	ローカル	全てのサブモジュールを使ってネットワークをリセットします。
ValueMapping [▶ 400]	ローカル	様々なモジュールに入力値を割り当てます。
SetHMIValues [▶ 400]	ローカル	オプション：モジュールの入出力をグローバルHMI変数にマッピングします。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4.2.1 ValueMapping

構文

```
METHOD ValueMapping : BOOL
VAR_INPUT
    pAnalysis : POINTER TO FB_Analysis;
END_VAR
```

入力

名前	データ型	説明
pAnalysis	FB_Analysis	Analysis FBのインスタンス

戻り値

名前	データ型	説明
ValueMapping	BOOL	

6.8.2.4.2.2 SetHMIValues

構文

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバルHMI構造体へのポインタ

戻り値

名前	データ型	説明
SetHMIValues	BOOL	完了した場合にTRUE

6.8.2.4.2.3 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    pAnalysis: POINTER TO FB_Analysis;
END_VAR
```

入力

名前	データ型	説明
pAnalysis	FB_Analysis	Analysis FBのインスタンス。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.4.2.4 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.4.3 FB_Module

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

構文

定義:

```
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
```

入力

名前	データ型	説明
Module inputs		選択したモジュールの入力。

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。
Module outputs		選択したモジュールの出力。

メソッド

名前	定義場所	説明
Call [▶ 402]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 402]	ローカル	モジュールをリセットします。
SetHMI [▶ 402]	ローカル	グローバル HMI 構造体への入出力を設定します。
GetHMI [▶ 403]	ローカル	オプション：グローバルHMI構造体の入力をモジュールの入力に設定します。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4.3.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M : I_ValueMapping_M2M;]
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	解析用データ
ipValueMapping_M2M	I_ValueMapping_M2M	オプション：モジュール間の値のマッピングに必要

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.4.3.2 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.4.3.3 SetHMI

構文

```
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
```

入力

名前	データ型	説明
nContent	DINT	HMI Content Index
pContent	PVOID	HMIコンテンツ構造体へのポインタ
bHMIReinit	BOOL	HMIコンテンツ構造体を初期化します。

戻り値

名前	データ型	説明
SetHMI	BOOL	完了した場合にTRUE

6.8.2.4.3.4 GetHMI (4.1)

構文

```
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
```

入力

名前	データ型	説明
nContent	DINT	HMI Content Index
pContent	PVOID	HMIコンテンツ構造体へのポインタ

戻り値

名前	データ型	説明
GetHMI	BOOL	完了した場合にTRUE

6.8.2.4.4 FB_Analysis

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

構文

定義:

```
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
```

入力

名前	データ型	説明
ipV[n]_VirtualInputs	I_V[n]_Virtual_Input_Source	特定のデータソースからの解析用データ

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。

メソッド

名前	定義場所	説明
Call [▶ 404]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 405]	ローカル	解析全体をリセットします。
ResultStream [▶ 405]	ローカル	オプション：結果データを作成する必要がある場合

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4.4.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	解析のためのデータ。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.4.4.2 Reset

構文

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

入力

名前	データ型	説明
stReset	ST_AnalysisReset	どのモジュールまたはネットワークをリセットするかを定義する構造体。

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.4.4.3 ResultStream

構文

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

入力

名前	データ型	説明
ipResults	I_Results	FB Resultsへのインターフェースポインタ

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.4.5 FB_Results

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルに保存します。

構文

定義:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    stResults: ST_Results;
END_VAR
```

出力

名前	データ型	説明
stResults	ST_Results	結果データの全要素を含む構造体。

メソッド

名前	定義場所	説明
Call [▶ 406]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
AddResult [▶ 406]	ローカル	結果データにサンプルを追加します。
SendResults [▶ 406]	ローカル	結果データのバッファされたサンプルを全て送信します。
Release [▶ 407]	ローカル	結果データのストリームまたはファイルを閉じます。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4.5.1 Call

構文

```
METHOD Call : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.4.5.2 AddResult

構文

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

入力

名前	データ型	説明
tTimestamp	ULINT	サンプルのタイムスタンプ
stSample	ST_Results	サンプルの構造体

戻り値

名前	データ型	説明
AddResult	BOOL	

6.8.2.4.5.3 SendResults

構文

```
METHOD SendResults : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
SendResults	BOOL	

6.8.2.4.5.4 Release

構文

```
METHOD Release : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Release	BOOL	

6.8.2.4.6 MAIN_Analytics

MAIN_Analyticsプログラムでは、DataSource FBが呼び出され、リセット機能が管理され、必要に応じて値がHMIダッシュボードにマッピングされます。プログラムは独立したタスクに割り当てられます。

構文

定義:

```
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
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.4.7 MAIN_Analytics_C[n]

FB Analysisは、MAIN_Analytics_C[n]プログラムで呼び出されます。プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

HMI値のマッピングはアクションA_MapToHMIで行われます。

構文

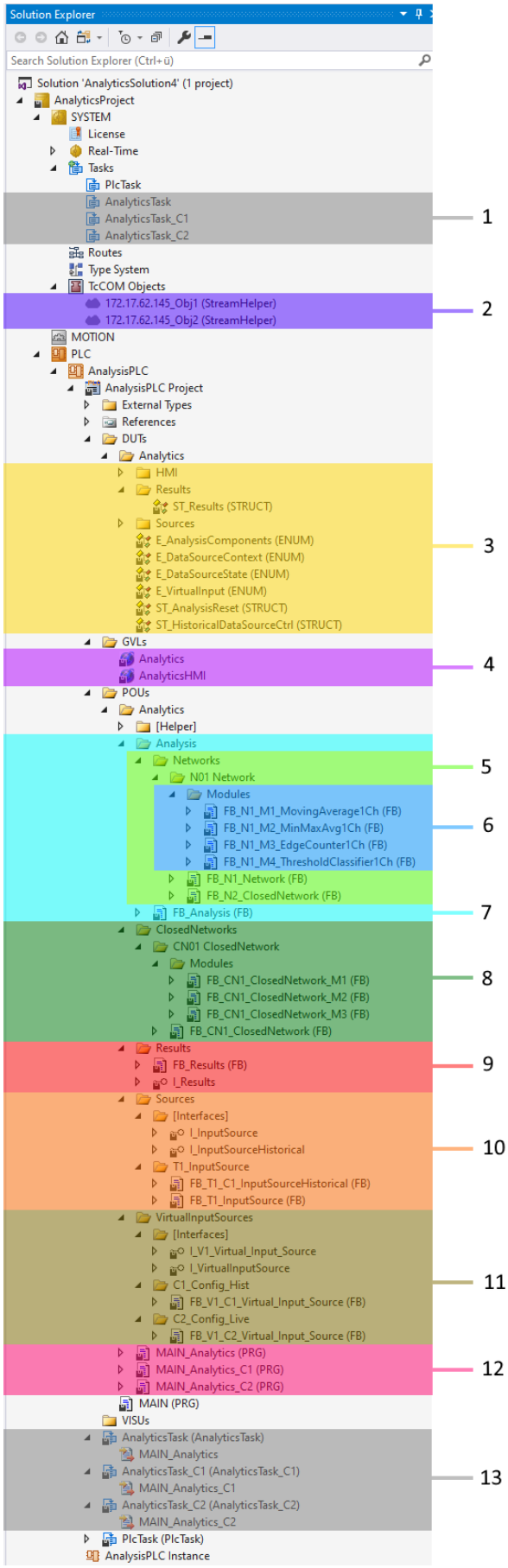
定義:

```
PROGRAM MAIN_Analytics_C[n]
VAR_INPUT
    stReset: ST_AnalysisReset;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
    nLastDataHandle: ULINT;
END_VAR
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5 Code version 5.0



1: Tasks

解析とVirtual Input Sourceの各設定ごとに、個別のタスクが作成されます。

2: StreamHelper

MQTTバイナリストリームの各データソースに対して、コード生成はストリームヘルパーオブジェクトのインスタンスを作成し、受信するバイナリデータのパターンを処理します。

3: DataTypes

データ型は解析用に作成されます。これらには、リセット機能や結果処理のためのSTRUCT、および各種コンポーネントを選択するためのENUMが含まれています。

4: GVLs

モジュールの入出力を HMI ダッシュボードに簡単にマッピングするために、選択した変数がグローバル変数として生成されます。さらに、Data Sourceファンクションブロックインスタンスと各種パラメータがグローバル変数として生成されます。

5: Network

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

6: Modules

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

7: Analysis

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

8: ClosedNetwork

ClosedNetwork FBは、全てのサブネットワークとモジュールを含めて一度だけ生成されます。これらは、解析の中で複数回インスタンス化できます。こうすることで、生成されるコードを減らし、単純化できます。

9: Results

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をバイナリまたはJson形式でメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルにローカルに保存します。

10: InputSource

FB InputSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。

11: VirtualInputSource

VirtualInputSource インターフェースは、解析データから InputSource シンボルを抽象化します。

12: MAIN PRGs

MAIN_Analyticsプログラムでは、DataSource FBが呼び出され、リセット機能が管理され、必要に応じて値がHMIダッシュボードにマッピングされます。プログラムは独立したタスクに割り当てられます。

MAIN_Analytics_Vx_Cxプログラムでは、FB-Analysisが呼び出されます。各プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics バージョン 3.1.0.0以上

6.8.2.5.1 FB_Analysis

解析ルーチン全体はAnalysis FBで定義されています。設定されたすべてのネットワークが、対応するモジュールおよびエラー処理とともに作成されます。

構文

定義:

```
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
```

入力

名前	データ型	説明
nConfigurationID	INT	設定インデックス
ipV[n]_VirtualInputs	I_V[n]_Virtual_Input_Source	特定のデータソースからの解析用データ

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。

プロパティ

名前	データ型	アクセス	説明
nContext	DWORD	Get	コンテキストのインデックス

メソッド

名前	定義場所	説明
Call [▶ 412]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 412]	ローカル	解析全体をリセットします。
ResultStream [▶ 413]	ローカル	オプション：結果ストリームを作成する必要がある場合。
SetHMIValues [▶ 413]	ローカル	HMI構造体の充填メソッド
GetHMIValues [▶ 413]	ローカル	HMIから解析パラメータを設定するためのメソッド。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.1.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source: I_V[n]_Virtual_Input_Source;
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	解析のためのデータ。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.5.1.2 Reset

構文

```
METHOD Reset : BOOL
VAR_IN_OUT
    stReset: ST_AnalysisReset;
END_VAR
```

入力

名前	データ型	説明
stReset	ST_AnalysisReset	どのモジュールまたはネットワークをリセットするかを定義する構造体。

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.5.1.3 ResultStream

構文

```
METHOD ResultStream : BOOL
VAR_INPUT
    ipResults: I_Results;
END_VAR
```

入力

名前	データ型	説明
ipResults	I_Results	FB結果へのインターフェースポインタ

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.5.1.4 SetHMIValues

構文

```
METHOD SetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバルHMI構造体へのポインタ

戻り値

名前	データ型	説明
SetHMIValues	BOOL	完了した場合にTRUE

6.8.2.5.1.5 GetHMIValues

構文

```
METHOD GetHMIValues : BOOL
VAR_INPUT
    pHMI_N[n]_[Network1] : POINTER TO ST_HMI_N[n]_[Network1];
END_VAR
```

入力

名前	データ型	説明
pHMI_N[n]_[Network1]	POINTER TO ST_HMI_N[n]_[Network1]	グローバルHMI構造体へのポインタ

戻り値

名前	データ型	説明
SetHMIValues	BOOL	完了した場合にTRUE

6.8.2.5.2 FB_Network

全てのモジュールは、設定した解析の見通しと構造をより明確にするために、特定のネットワーク内に分類されます。

構文

定義:

```
FUNCTION_BLOCK FB_N[n]_[Network1]
VAR_OUTPUT
    bError: BOOL;
    ipTcResult: I_TcMessage;
END_VAR
VAR
    [module FBs]
END_VAR
```

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。

メソッド

名前	定義場所	説明
Call [▶ 389]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 390]	ローカル	全てのサブモジュールを使ってネットワークをリセットします。
ValueMapping [▶ 415]	ローカル	様々なモジュールに入力値を割り当てます。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.2.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    pAnalysis: POINTER TO FB_Analysis;
END_VAR
```

入力

名前	データ型	説明
pAnalysis	FB_Analysis	Analysis FBのインスタンス。

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.5.2.2 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.5.2.3 値マッピング

構文

```
METHOD ValueMapping : BOOL
VAR_INPUT
    pAnalysis : POINTER TO FB_Analysis;
END_VAR
```

入力

名前	データ型	説明
pAnalysis	FB_Analysis	Analysis FBのインスタンス

戻り値

名前	データ型	説明
値マッピング	BOOL	

6.8.2.5.3 FB_Module

FBモジュールには、Workbenchコンフィグレータで設定されたモジュールの全ての入出力が含まれます。また、実行中にモジュールを再設定することも可能です。これを行うには、パラメータを変更し、INPUT bReconfigure に立ち上がりエッジを与えて設定プロセスを開始します。

構文

定義:

```
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
```

入力

名前	データ型	説明
Module inputs		選択したモジュールの入力。

出力

名前	データ型	説明
bError	BOOL	エラーが発生すると TRUE になります。
ipTcResult	I_TcMessage	TwinCAT 3 EventLoggerのメッセージインターフェースは、戻り値に関する詳細情報を提供します。
Module outputs		選択したモジュールの出力。

プロパティ

名前	データ型	アクセス	説明
nContext	DWORD	Get	コンテキストのインデックス

メソッド

名前	定義場所	説明
Call [▶ 417]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
Reset [▶ 417]	ローカル	モジュールをリセットします。
SetHMI [▶ 417]	ローカル	グローバル HMI 構造体への入出力を設定します。
GetHMI [▶ 418]	ローカル	オプション：グローバルHMI構造体の入力をモジュールの入力に設定します。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.3.1 Call

構文

```
METHOD Call : BOOL
VAR_INPUT
    ipVirtual_Input_Source : I_V[n]_Virtual_Input_Source;
    [ipValueMapping_M2M : I_ValueMapping_M2M;]
END_VAR
```

入力

名前	データ型	説明
ipVirtual_Input_Source	I_V[n]_Virtual_Input_Source	解析用データ
ipValueMapping_M2M	I_ValueMapping_M2M	オプション：モジュール間の値のマッピングに必要

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.5.3.2 Reset

構文

```
METHOD Reset : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Reset	BOOL	完了した場合にTRUE

6.8.2.5.3.3 SetHMI

構文

```
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
```

入力

名前	データ型	説明
nContent	DINT	HMI Content Index
pContent	PVOID	HMIコンテンツ構造体へのポインタ
bHMIReinit	BOOL	HMIコンテンツ構造体を初期化します。

戻り値

名前	データ型	説明
SetHMI	BOOL	完了した場合にTRUE

6.8.2.5.3.4 GetHMI

構文

```
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
```

入力

名前	データ型	説明
nContent	DINT	HMI Content Index
pContent	PVOID	HMIコンテンツ構造体へのポインタ

戻り値

名前	データ型	説明
GetHMI	BOOL	完了した場合にTRUE

6.8.2.5.4 FB_InputSource

FB DataSourceは、様々なソースからの入力値の受信を管理します。OUTPUT宣言の中に、設定された全ての入力が含まれます。

構文

定義:

```
FUNCTION_BLOCK FB_T[n]_InputSource IMPLEMENTS I_InputSource
VAR
END_VAR
```

インターフェース

データ型	説明
I_InputSource	データソースとの通信用インターフェース

メソッド

名前	定義場所	説明
Call [▶ 419]	ローカル	TwinCATドライバとのバックグラウンド通信メソッド。このメソッドは周期的に呼び出す必要があります。
GetData [▶ 419]	ローカル	指定した要素のデータを取得するメソッド。
NewDataAvailable [▶ 420]	ローカル	新しいデータが利用可能かどうかをチェックするメソッド。

名前	定義場所	説明
AddClient [▶ 420]	ローカル	Clientsデータを追加するためのメソッド。
ClientDone [▶ 420]	ローカル	クライアントが全てのデータを受信したことを通知するメソッド。

プロパティ

名前	データ型	アクセス	説明
bReadNewData	BOOL	Get	.
eDataState	E_DataSourceState	Get	
nDataHandle	ULINT	Get	
nElements	UDINT	Get	
nMaxOversamplingFactor	UDINT	Get	

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.4.1 Call

構文

```
METHOD Call : BOOL
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.5.4.2 GetData

構文

```
METHOD GetData : BOOL
VAR_INPUT
    nElement    : UDINT;
    pInputs     : PVOID;
    nInputsSize : UDINT
END_VAR
VAR_OUTPUT
    nTimestamp  : ULINT;
    nContext    : DWORD;
END_VAR
```

入力

名前	データ型	説明
nElement	UDINT	特定のサンプルを取得するためのElement ID
pInputs	PVOID	データ構造体へのポインタ
nInputsSize	UDINT	データ構造体のサイズ

出力

名前	データ型	説明
nTimestamp	ULINT	タイムスタンプのデータ
nContext	DWORD	データコンテキスト

戻り値

名前	データ型	説明
GetData	BOOL	新しい要素が選択された場合に TRUE

6.8.2.5.4.3 NewDataAvailable

構文

```
METHOD NewDataAvailable : BOOL
VAR_INPUT
    nLastDataHandle : ULINT;
END_VAR
```

入力

名前	データ型	説明
nLastDataHandle	ULINT	最後に取得したデータパケットのハンドル

戻り値

名前	データ型	説明
NewDataAvailable	BOOL	新しいデータが利用可能であればTRUE

6.8.2.5.4.4 AddClient

構文

```
METHOD AddClient : BOOL
VAR_OUTPUT
    nClientID    : DWORD;
END_VAR
```

出力

名前	データ型	説明
nClientID	LWORD	Client-ID

6.8.2.5.4.5 ClientDone

構文

```
METHOD ClientDone : BOOL
VAR_OUTPUT
    nClientID    : DWORD;
END_VAR
```

出力

名前	データ型	説明
nClientID	LWORD	Client-ID

6.8.2.5.5 FB_VirtualInputSource

FB VirtualInputSourceは、様々な解析設定のInputSourceを抽象化します。Analytics Workbenchで構成された仮想入力はここで設定します。

構文

定義:

```
FUNCTION_BLOCK FB_V[n]_C[n]_VirtualInputSource IMPLEMENTS I_VirtualInputSource,
I_V[n]_VirtualInputSource
VAR
END_VAR
```

インターフェース

データ型	説明
I_VirtualInputSource	データソースとの通信用インターフェース
I_V[n]_VirtualInputSource	定義された全ての仮想入力を提供するインターフェース

メソッド

名前	定義場所	説明
SourceSync [▶ 422]	ローカル	複数のFBインスタンスを同期するためのメソッド。
NextData [▶ 422]	ローカル	次のデータセットを承認するメソッド。
Done [▶ 422]	ローカル	データパケット全体が処理されたことを通知するメソッド。

プロパティ

名前	データ型	アクセス	説明
bEndOfData	BOOL	Get	データパケットの終了を通知します。
dtTimestamp	DCTIMESTRUCT	Get	現在、受信済みのデータセットのタイムスタンプ
nDataHandle	ULINT	Get	データハンドル
nContext	DWORD	Get	データコンテキスト
tTimestamp	ULINT	Get	現在受信しているデータセットのタイムスタンプ

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.5.1 SourceSync

構文

```
METHOD SourceSync : BOOL
```

6.8.2.5.5.2 NextData

構文

```
METHOD NextData : BOOL
VAR_INPUT
    nMaxOversampling    : UDINT;
END_VAR
```

入力

名前	データ型	説明
nMaxOversampling	UDINT	オーバーサンプリングの最大値を指定します。

戻り値

名前	データ型	説明
NextData	BOOL	新しい要素が選択された場合に TRUE

6.8.2.5.5.3 完了

構文

```
METHOD Done : BOOL
VAR_INPUT
END_VAR
```

6.8.2.5.6 FB_Results

解析結果を保存またはストリーミングする必要がある場合、FB Resultsがこれを管理し、選択した変数をメッセージブローカーにストリーミングするか、データをAnalyticsバイナリファイルに保存します。

構文

定義:

```
FUNCTION_BLOCK FB_Results
VAR_OUTPUT
    nTimestamp: ULINT;
    stResults: ST_Results;
END_VAR
```

出力

名前	データ型	説明
nTimestamp	ULINT	結果の構造体データに関連したタイムスタンプ
stResults	ST_Results	結果データの全要素を含む構造体。

プロパティ

名前	データ型	アクセス	説明
bInitialized	BOOL	Get	ファンクションブロックが正しく初期化されたかどうかを示します。
nMaxSamples	INT	Get	バッファリングされた結果の最大数
nResultCount	INT	Get	現在のバッファされた結果数

メソッド

名前	定義場所	説明
Call [▶ 423]	ローカル	バックグラウンド通信のメソッド。このメソッドは周期的に呼び出す必要があります。
AddResult [▶ 423]	ローカル	結果データにサンプルを追加します。
SendResults [▶ 424]	ローカル	結果データのバッファされたサンプルを全て送信します。
Release [▶ 424]	ローカル	結果データのストリームまたはファイルを閉じます。

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.6.1 Call

構文

```
METHOD Call : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Call	BOOL	

6.8.2.5.6.2 結果の追加

構文

```
METHOD AddResult : BOOL
VAR_INPUT
    tTimestamp: ULINT;
    stSample: ST_Results;
END_VAR
```

入力

名前	データ型	説明
tTimestamp	ULINT	サンプルのタイムスタンプ
stSample	ST_Results	サンプルの構造体

戻り値

名前	データ型	説明
AddResult	BOOL	

6.8.2.5.6.3 SendResults

構文

```
METHOD SendResults : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
SendResults	BOOL	

6.8.2.5.6.4 Release

構文

```
METHOD Release : BOOL
VAR
END_VAR
```

戻り値

名前	データ型	説明
Release	BOOL	

6.8.2.5.7 MAIN_Analytics

MAIN_Analyticsプログラムでは、InputSource FBが呼び出され、リセット機能が管理され、必要に応じて値がHMIダッシュボードとマッピングされます。プログラムは独立したタスクに割り当てられます。

構文

定義:

```
PROGRAM MAIN_Analytics
VAR
END_VAR
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.8.2.5.8 MAIN_Analytics_C[n]

FB Analysisは、MAIN_Analytics_C[n]プログラムで呼び出されます。プログラムは独立したタスクに割り当てられます。

1つの立ち上がりフラグで、単一のモジュール、ネットワーク全体、または定義済みのすべてのネットワークをリセットすることも可能です。まず、リセットするコンポーネントを選択します。その後、INPUT bResetの立ち上がりエッジがリセット処理を開始します。

全てのリセットコールはアクションA_Resetで定義されています。

「InputSources」の処理は、アクションA_InputSourcesで行われます。

HMI値のマッピングはアクションA_MapToHMIで行われます。

構文

定義:

```
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
```

要件

開発環境	ターゲットプラットフォーム	必要なPLCライブラリ
TwinCAT v3.1.4024.0	PCまたはCX (x64、x86)	Tc3_Analytics

6.9 HMIワンクリックダッシュボード

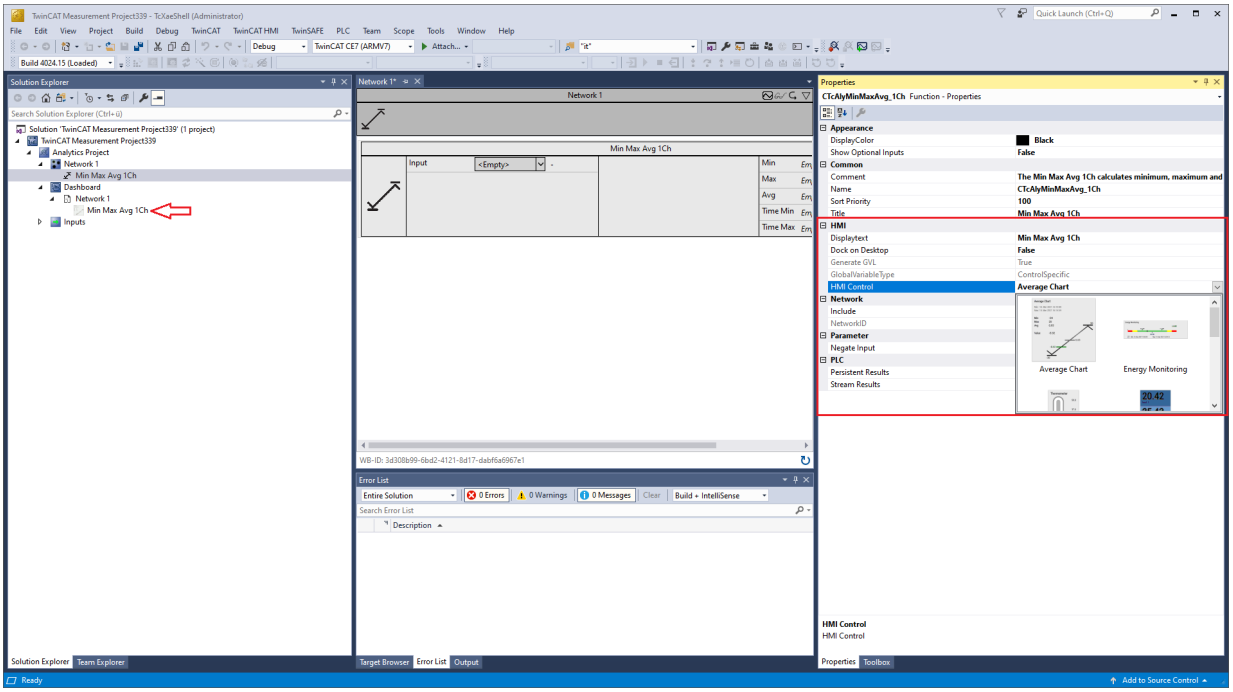
TwinCAT Analytics Workbench Configurator で設定されたすべてのモジュールおよびパラメータに対して、HMI コントロールを含む HMI ダッシュボードを自動生成できます。HMIダッシュボードはTwinCAT HMIをベースにしています。ランタイムへのデプロイ [▶ 357]によりPLCデータを表示します。

自動生成されるワンクリックダッシュボードは、新しいHMIバージョン1.12でのみ使用できます。Analytics HMI Controlsを使用するには、Analytics Runtimeライセンスが必要です。

✓ Analytics Workbenchプロジェクトを設定した後、各アルゴリズムごとにHMIコントロールを選択できます。

1. モジュールの**Properties** ウィンドウを開き、ドロップダウンリストから**HMI Control**を選択します。HMIダッシュボードのタイトルの表示テキストを変更できます（display text）。また、コントロールをスタートページに固定するかどうかも選択できます（Dock on Desktop）。**Solution Explorer**で

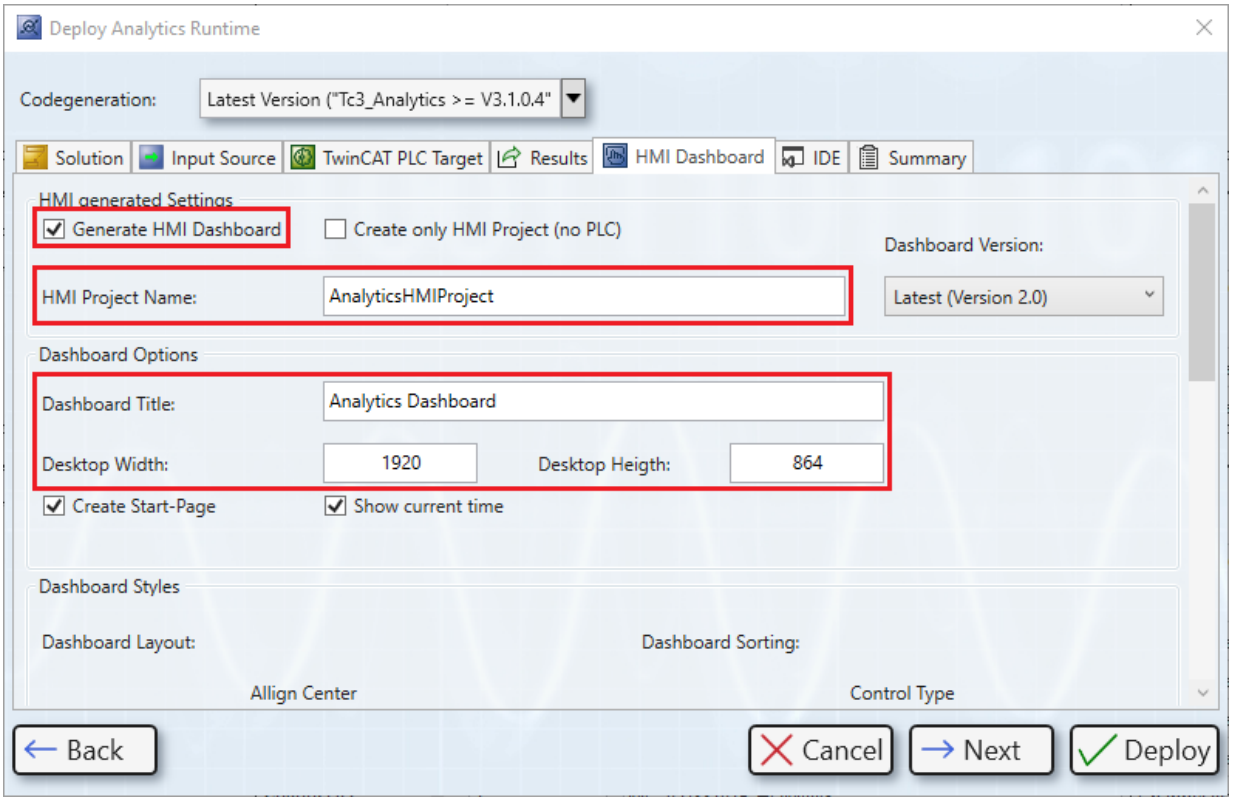
は、生成される全てのコントロールは Analyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理 [▶ 428] 以下に保存されます。



2. 設定が完了したら、コンテキストメニューの**Deploy Analytics Runtime**コマンドをクリックします。Deployウィザードが起動し、必要な設定を段階的に設定して使用できます。

   ⇨ HMI ダッシュボードは、**HMI Dashboard**タブで設定できます。

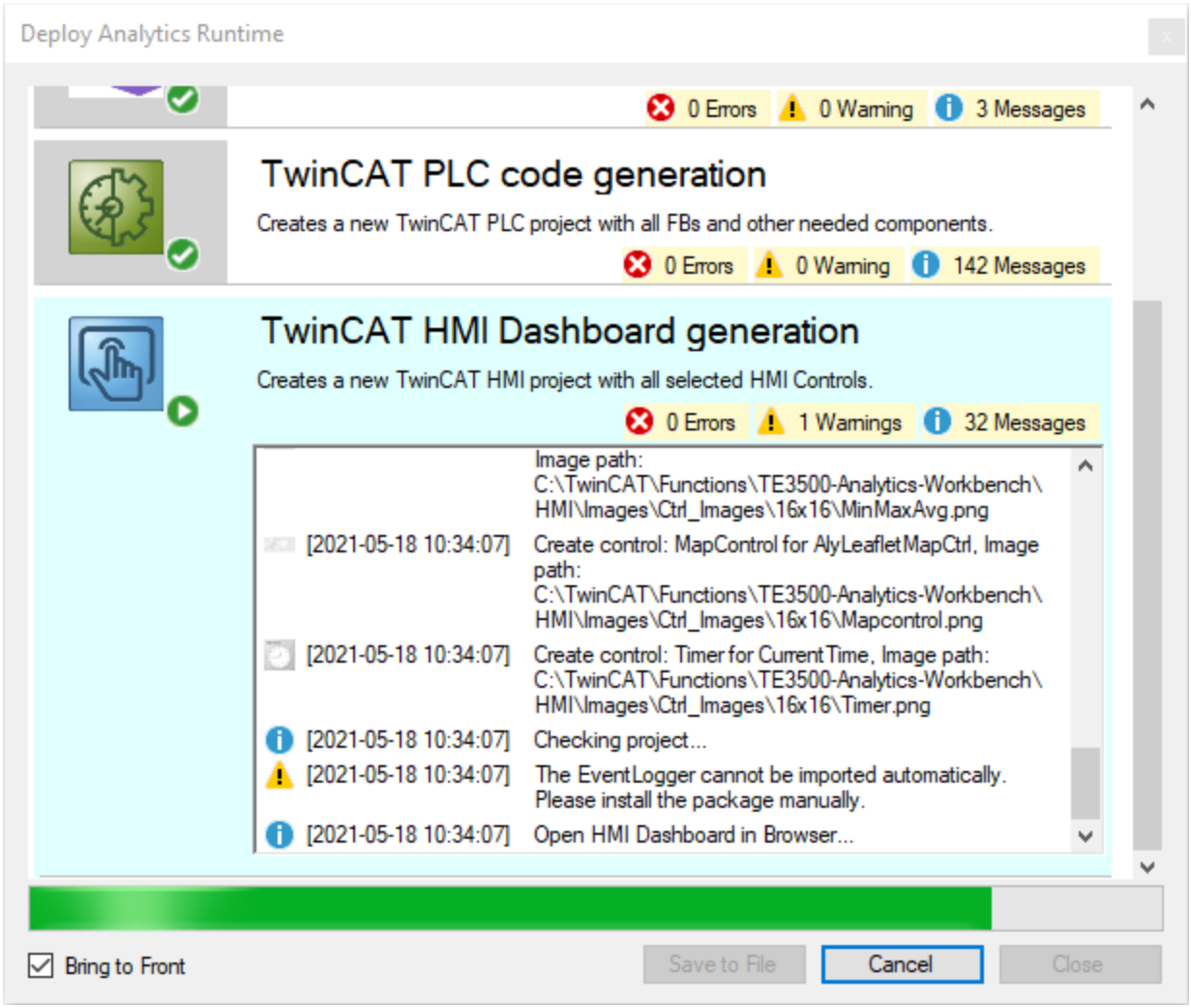
3. **Generate HMI Dashboard**チェックボックスをオンにします。また、PLCなしでHMIプロジェクトを1つのみ作成することも可能です。さらに、ダッシュボードに**HMI Project Name**を割り当て、Dashboard Title、Desktop HeightおよびDesktop Widthを設定して、ニーズに合わせてカスタマイズしたダッシュボードを作成することもできます。その他の設定については、Dashboard Configuration [▶ 460]で説明しています。



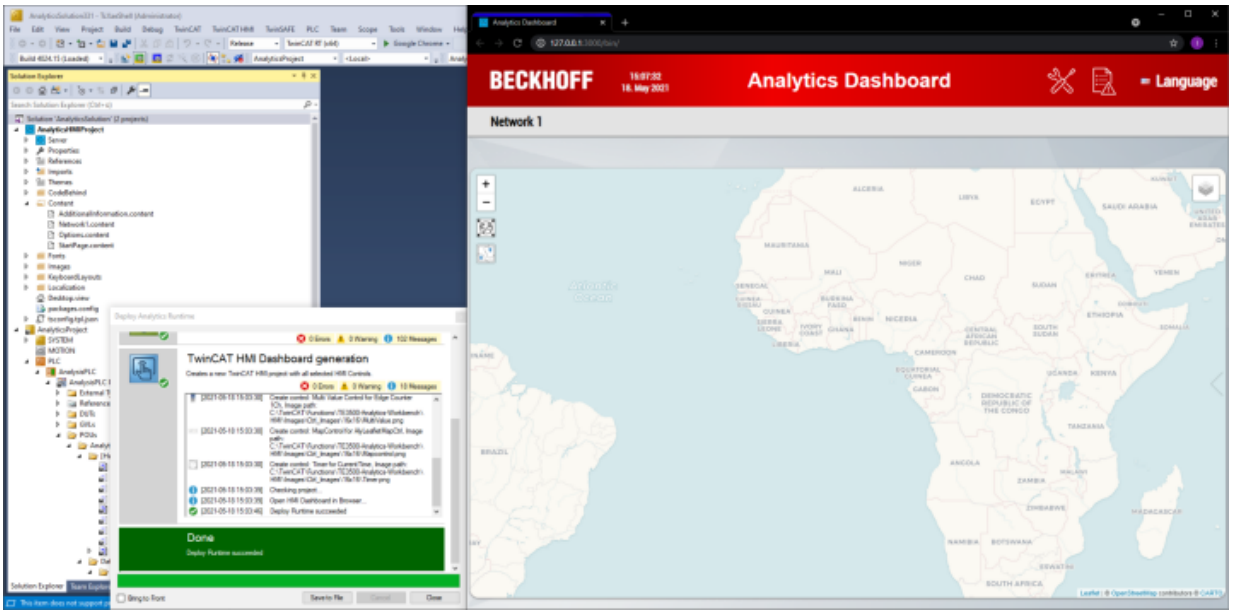
⇨ 通常どおり、最後のタブには、その世代で行った全ての設定が表示されます。

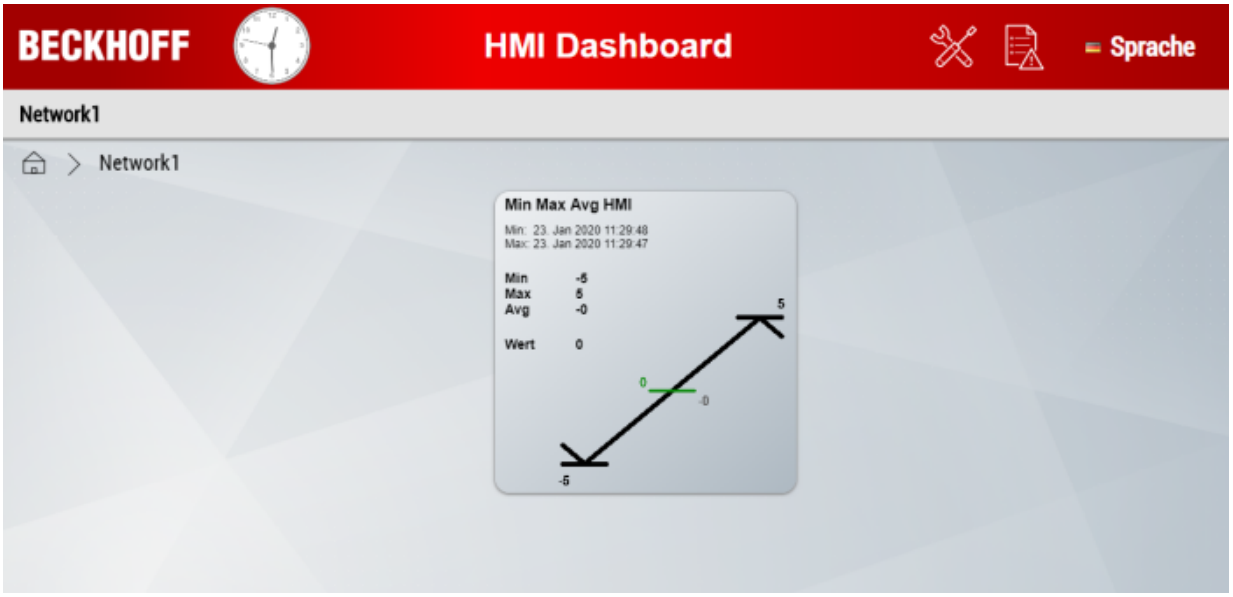
4. これで、**Deploy**をクリックして生成プロセスを開始できます。

⇨ HMIの生成は（選択されている場合）ランタイムデプロイの直後に開始されます。HMIダッシュボードを生成するための各手順は、生成プロセス中に概要ウィンドウにも表示されます。



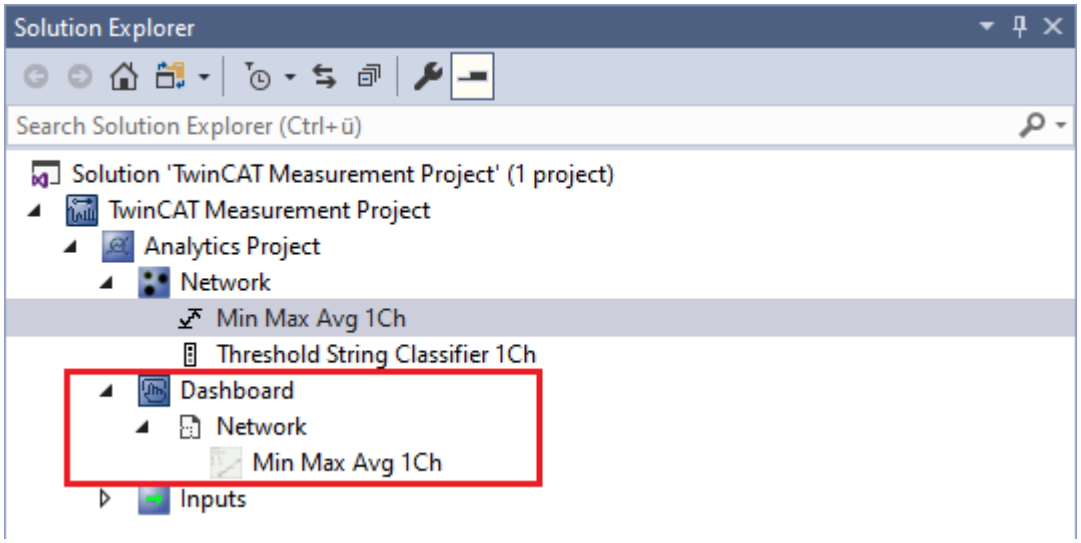
⇨ ダッシュボードはデフォルトのブラウザで自動的に開きます。



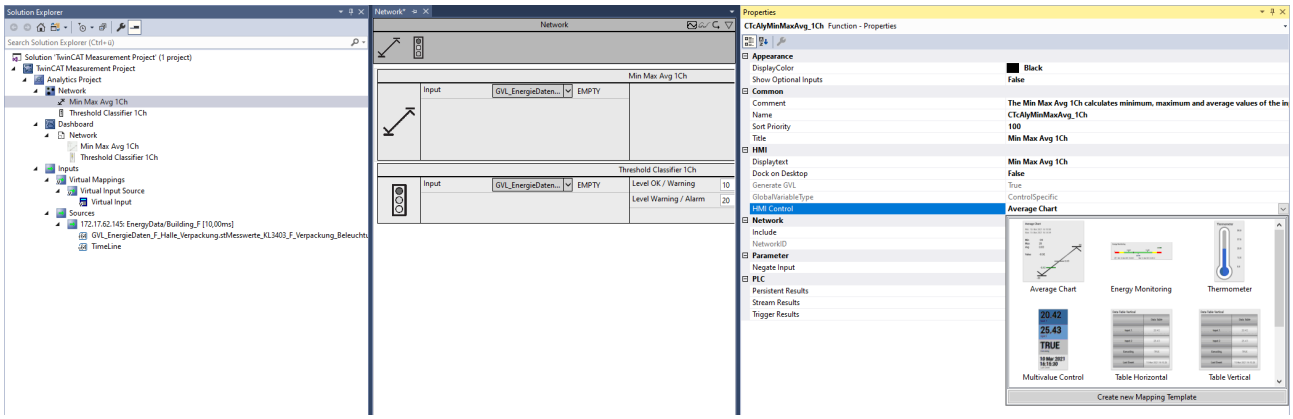


6.9.1 Analyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理

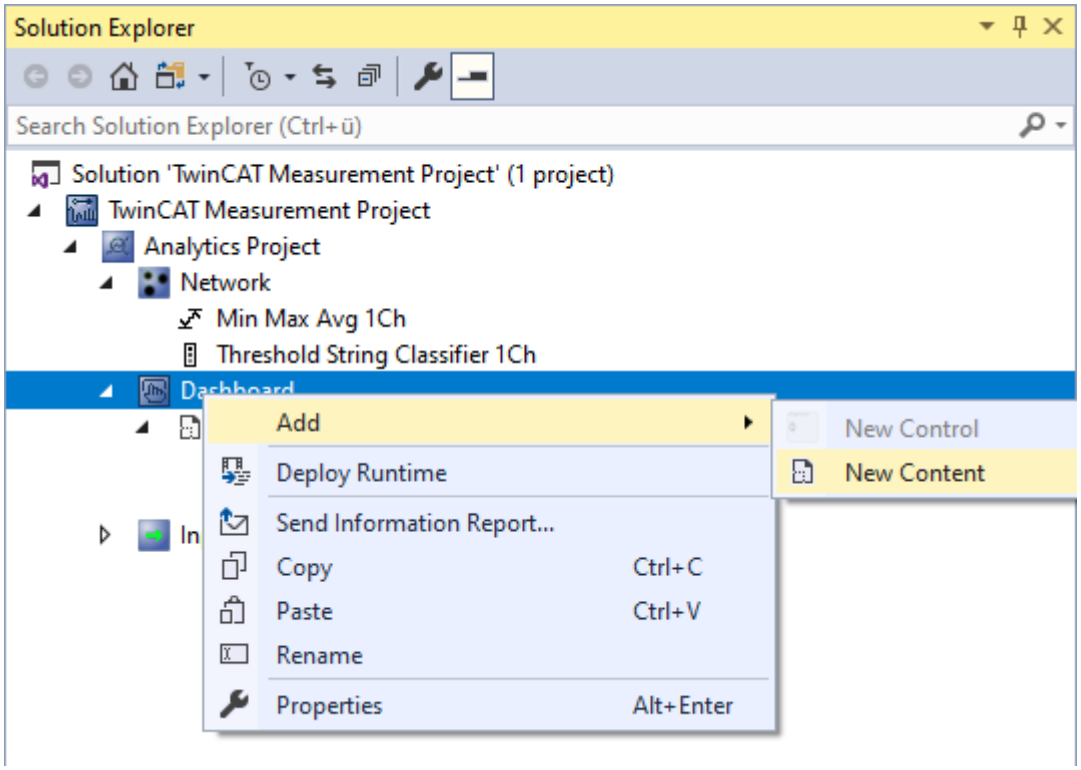
Analyticsプロジェクトには、ダッシュボード生成時に作成されるHMIコンテンツ（ページ）とHMIコントロール（表示要素）の設定があります。この設定は、Solution Explorerの**Dashboard node**で表示および変更できます。コンテンツやコントロールの名前を変更したり、ドラッグ＆ドロップで他のコンテンツに移動したり、コピー（**Ctrl-C**、**Ctrl-V**）したり、削除したりできます。コントロールをダブルクリックして編集することもできます。



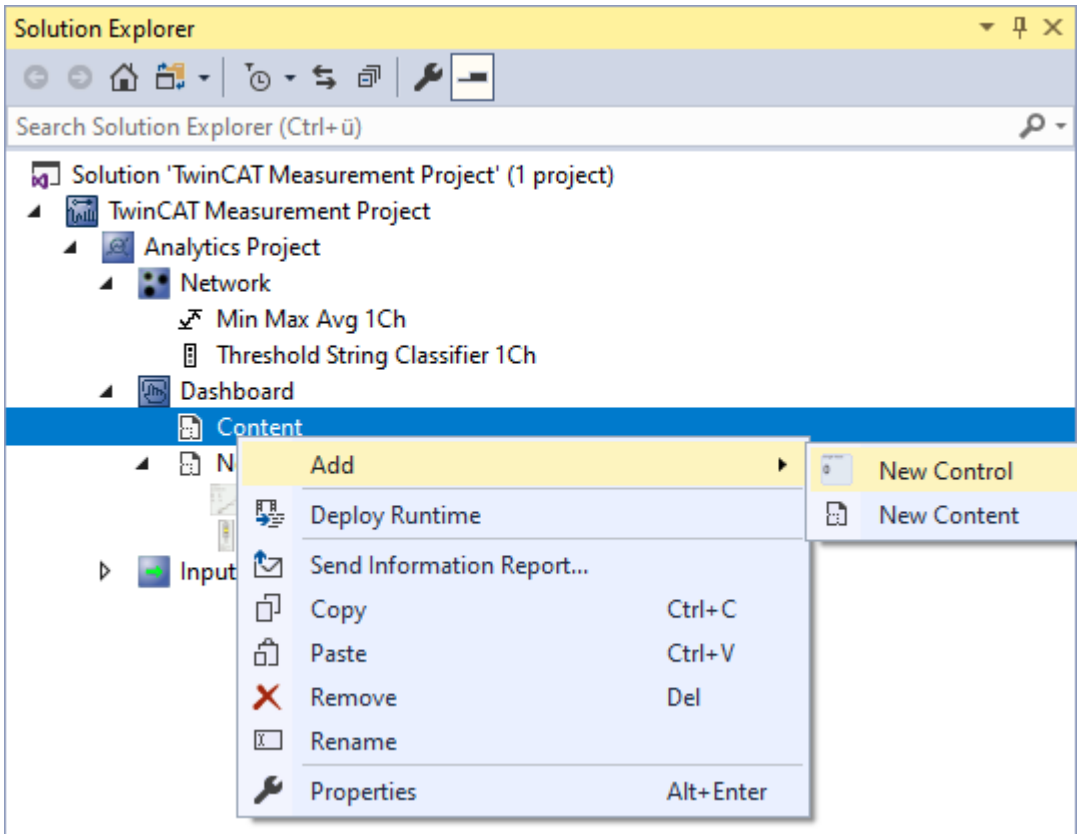
各Analyticsモジュールには既存のコントロールマッピングがあり、ウィンドウの**Properties**から選択できます（Create new Mapping Templateまたはカスタマイズされた独自コントロールの使用 [▶ 438]（項目2）を使って、自身で作成/編集することもできます）。モジュールにコントロールが選択されると、**Dashboard**ノードの下に表示されます。



右クリックして、**Dashboard**ノードと**Content**ノードに新しいコンテンツを作成できます。



同様に、**Content**ノードを右クリックして、**new controls**を追加できます。



Analytics Dashboard Wizardが開きます。このウィザードは、コントロールを追加するための設定を順を追ってガイドします。

Analytics Dashboard Wizard

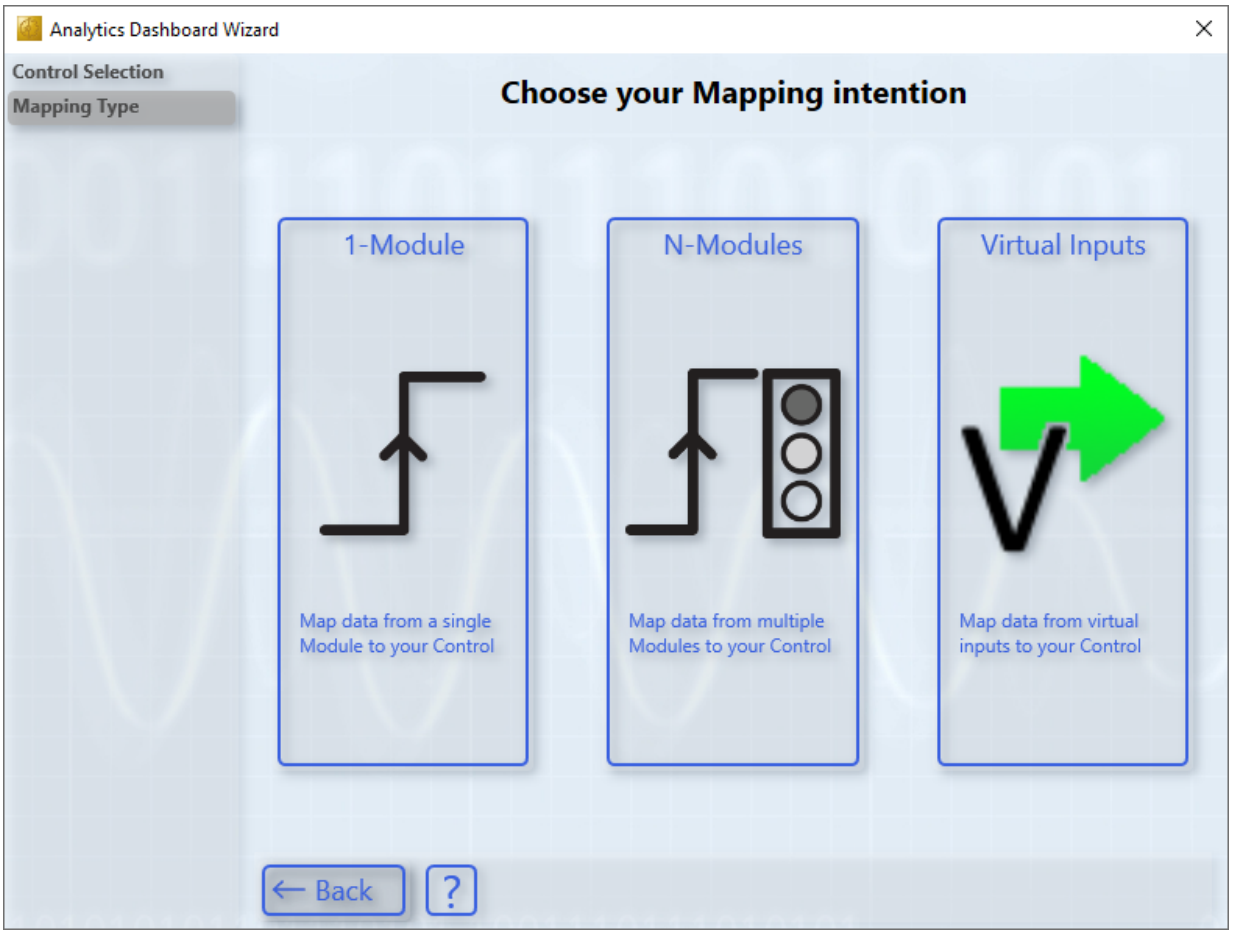
コントロールを追加してウィザード操作に慣れてください（モジュールのPropertiesウィンドウからコントロールを選択します）。その後、コントロールをダブルクリックすることで、ウィザード上で全ての設定を検索できます。そこで各設定手順に従います。

Analytics Dashboard Wizard

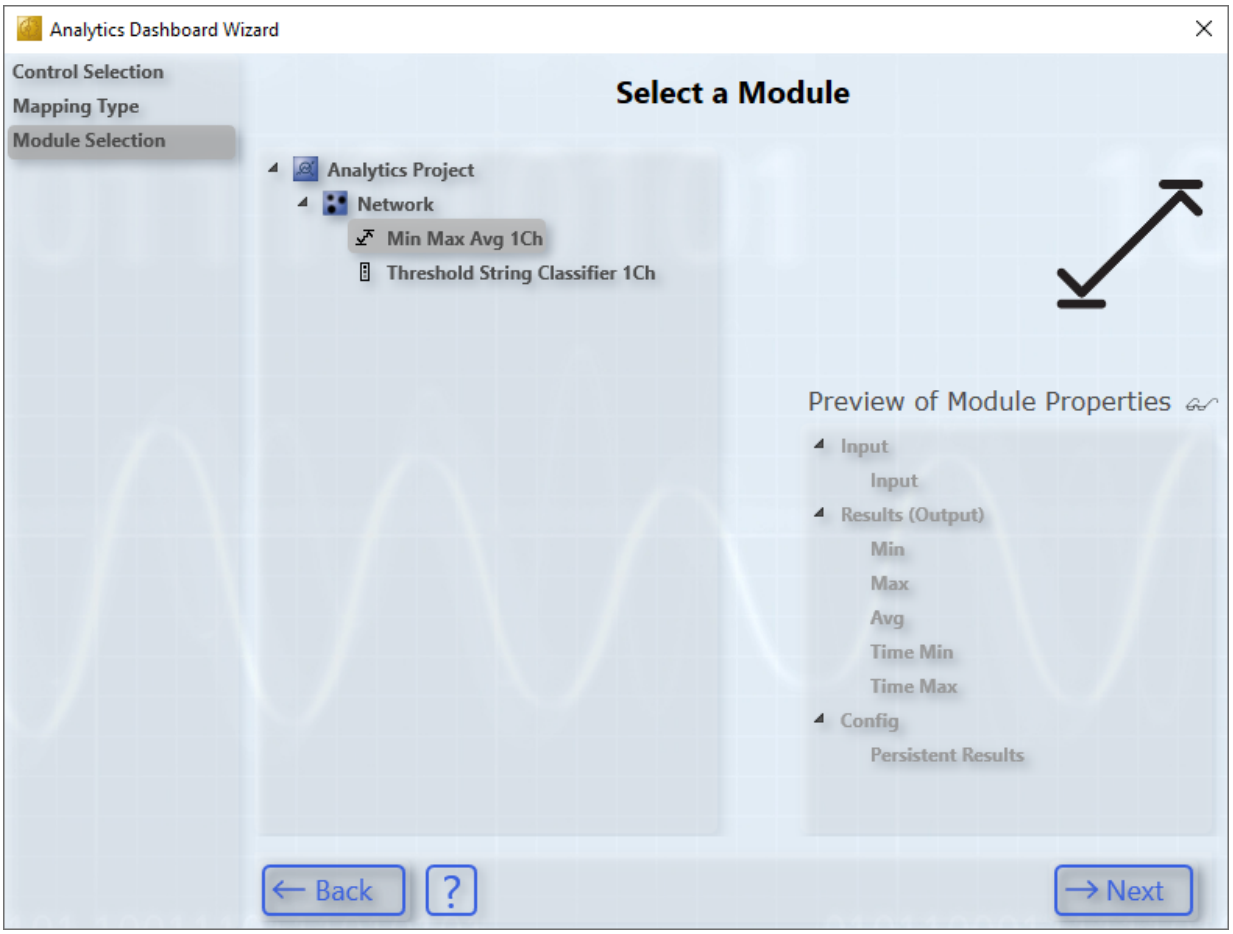
コントロールを選択します。使用可能なコントロールは全て左側にリストアップされています。デフォルトでは、アルゴリズムに依存しないコントロールのみが表示されます。**Show only default Controls**のチェックボックスをオフにすることで、全てのコントロールが使用可能になります。

1. ダッシュボードに単一の値を表示する場合には、**Single Value**を選択します。**Next**をクリックして続行します。

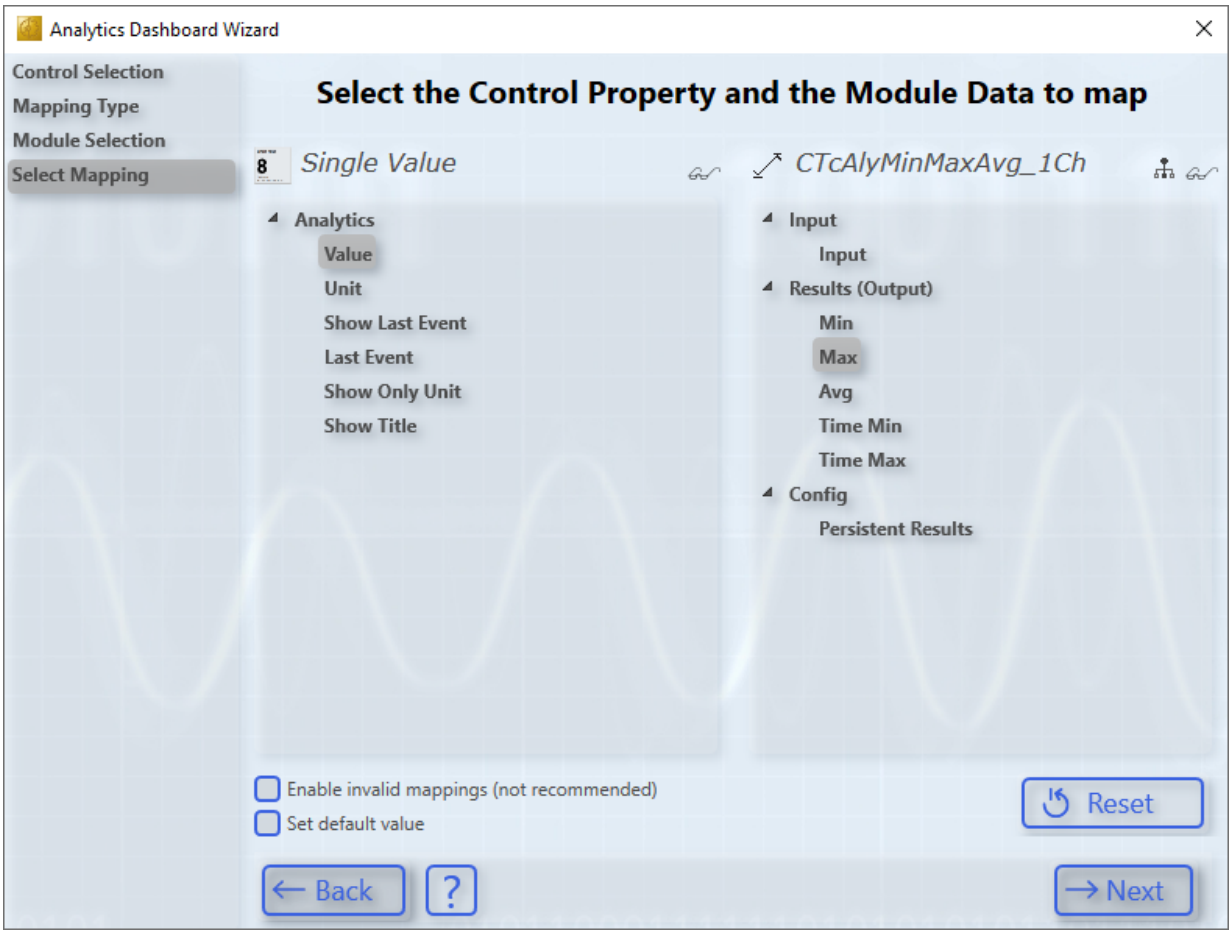
2. 1つのモジュール（**1-Module**）、複数のモジュール（**N-Module**）、または仮想入力（**Virtual Inputs**）からデータをリンクできるようになりました。この例では、**1-Module**を選択します。



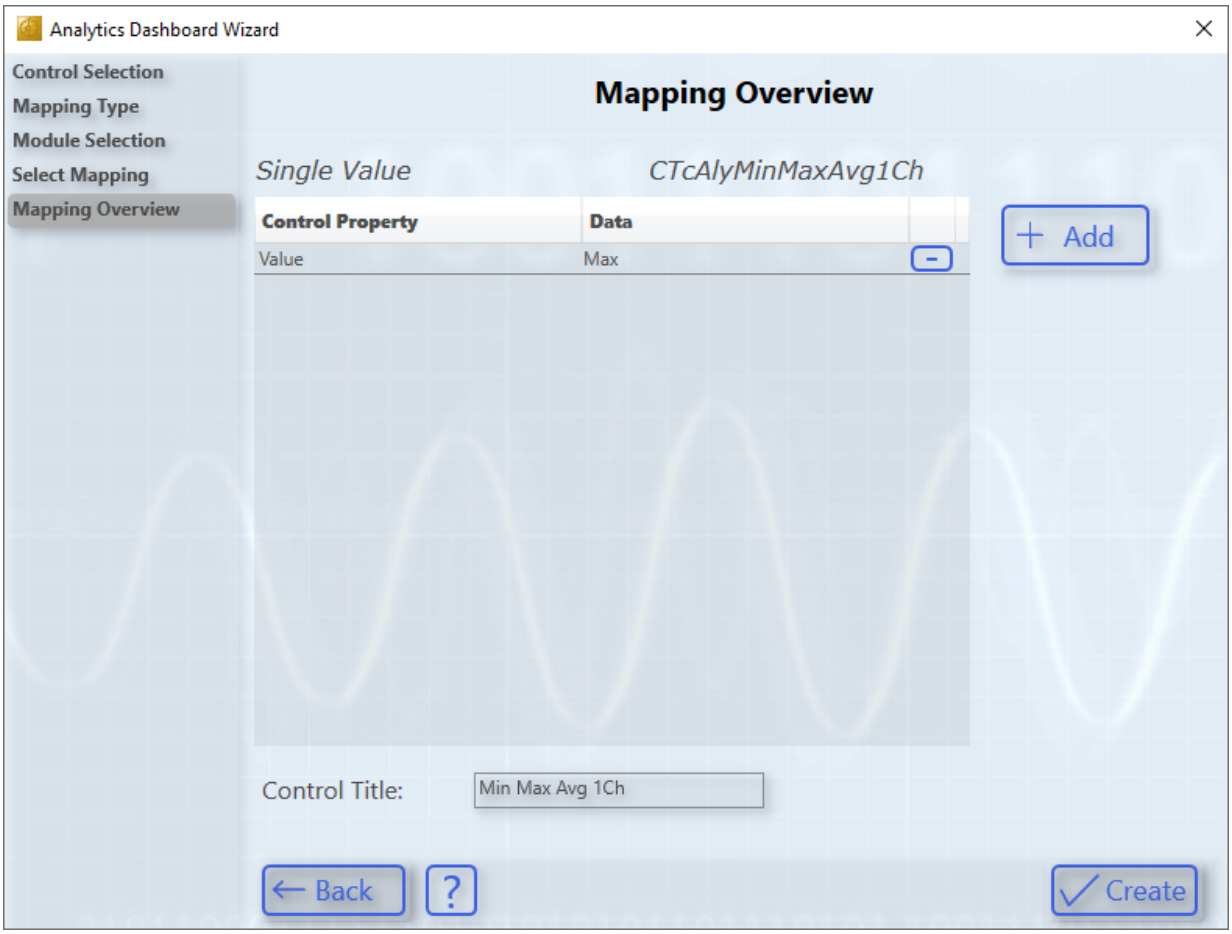
3. Analyticsプロジェクトの全モジュールがリストアップされています。データを表示したいモジュールを選択します。選択したモジュールのプレビューが右側に表示されます。



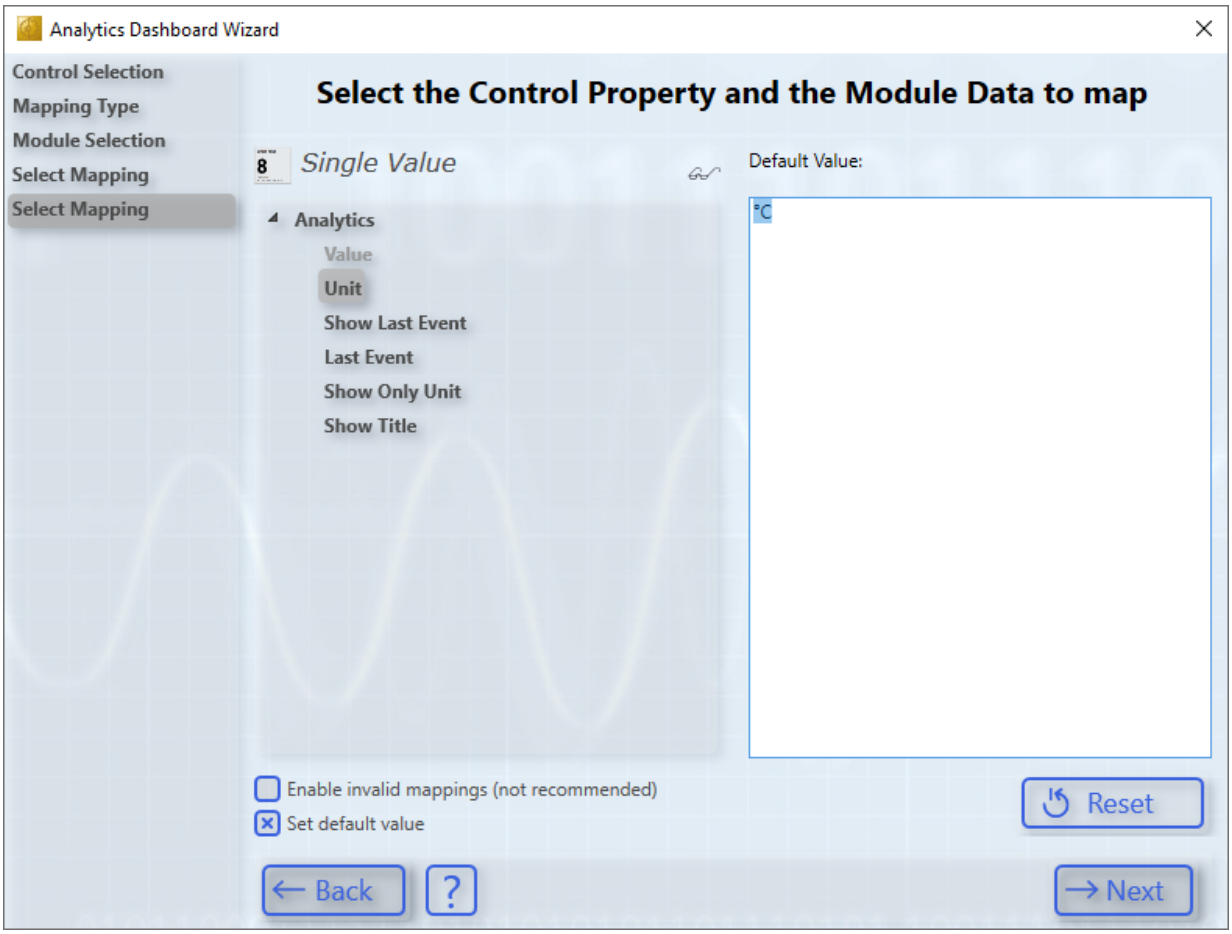
4. ここでは、最初のコントロールプロパティがモジュール変数にリンクされています。これを行うには、変数**Max**で表示する**Value** を選択します。



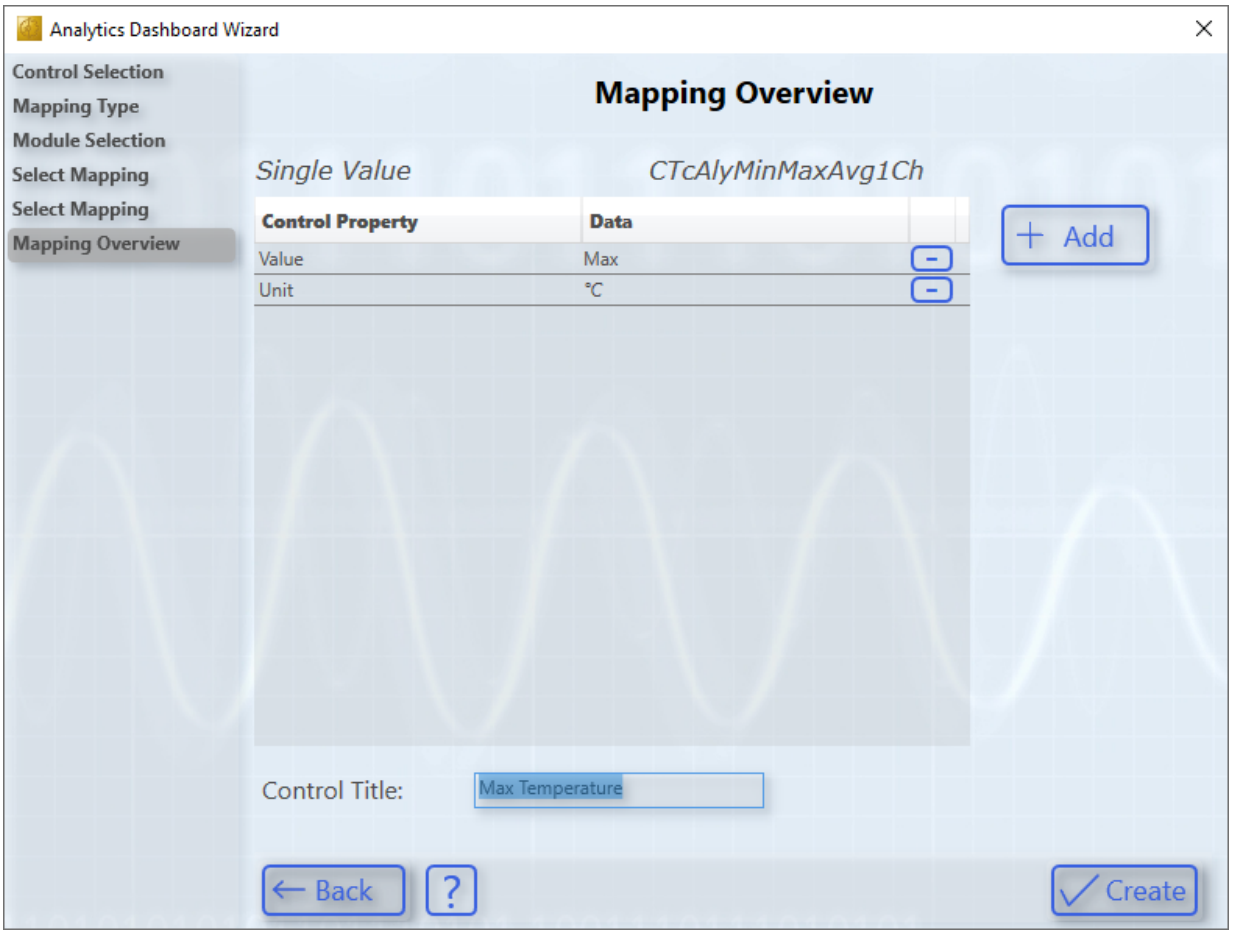
5. この概要ページには、既存のリンクが全てリストアップされています。**Add**で別のリンクを追加します。



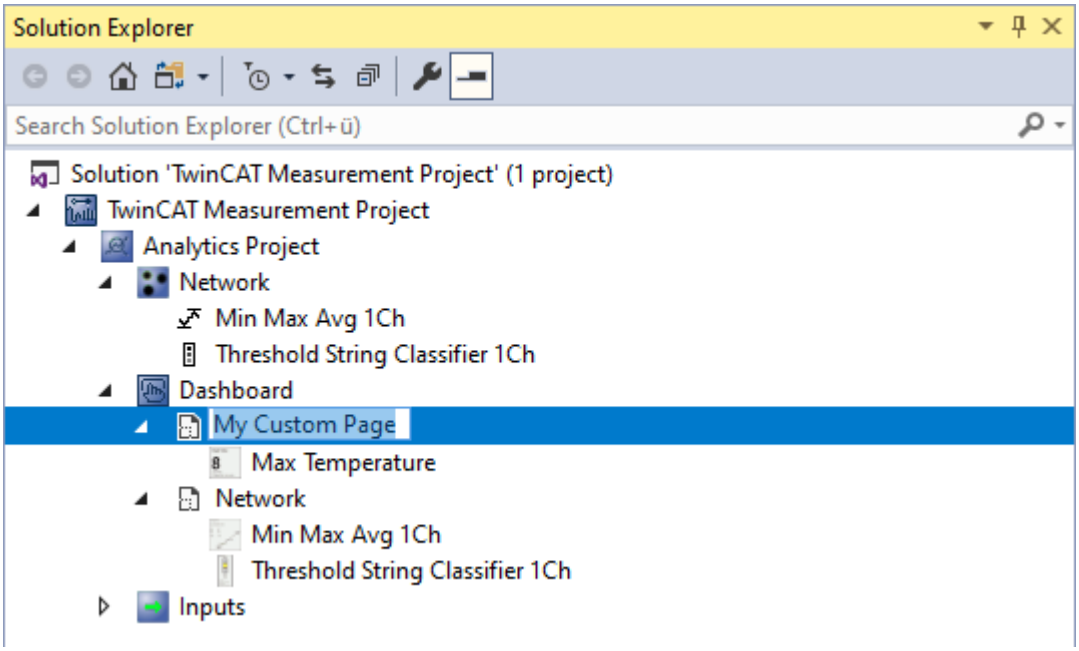
6. **Unit**を選択し、**Set default value**をチェックして固定値を割り当てます。右側にテキストフィールドが開くので、**°C**と入力します。



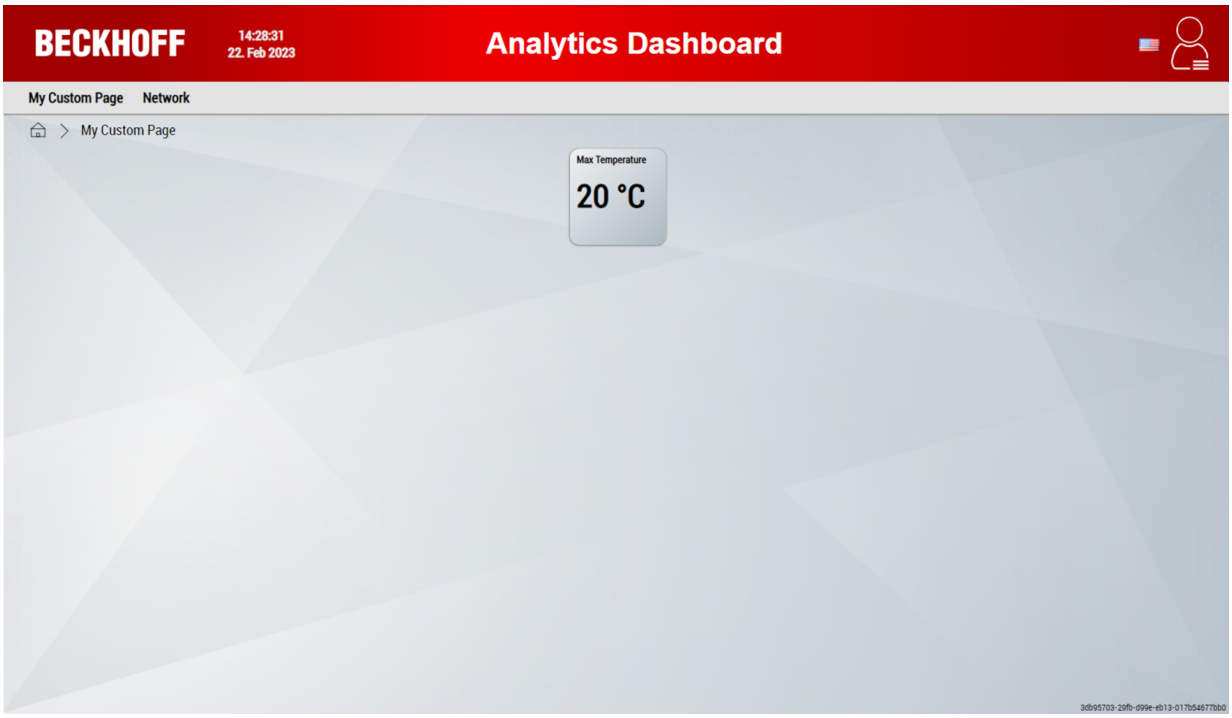
7. コントロールのタイトルを**Max Temperature**に変更し、**Create**でマッピングを追加します。



8. 作成された**Max Temperature**コントロールが**Dashboard**ノードに表示されます。コンテンツ名を変更するには、**Right-click** > **Rename**を選択します。



⇨ HMI と PLC の生成に成功すると、ナビゲーションから以前に作成した**My Custom Page**を開くことができます。マニュアルで作成したコントロールが表示されます。



6.9.2 カスタマイズされた独自コントロールの使用

HMI Control Mappingウィザード

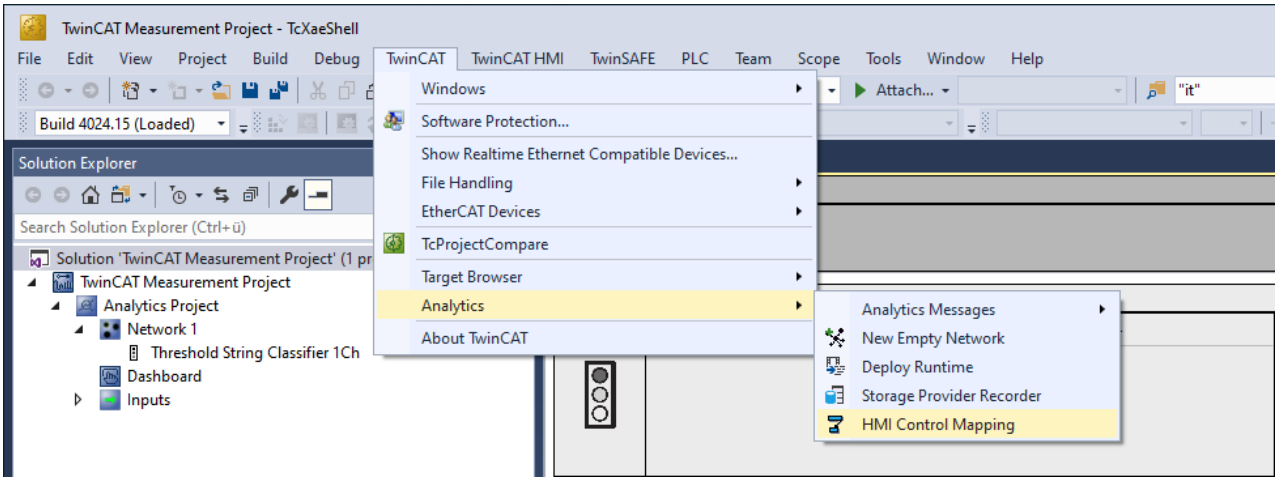
HMI Control Mappingウィザードは、以下を可能にします：

1.独自のコントロールを追加できます。

2.コントロールをモジュールクラスにマッピングする、または既存のマッピングを変更できます。

3.コントロールをモジュールインスタンスまたは仮想入力にマッピングできます（Analyticsプロジェクトが開いている必要があります）。

TwinCAT > Analytics > HMI Control Mappingタブからウィザードを開きます。

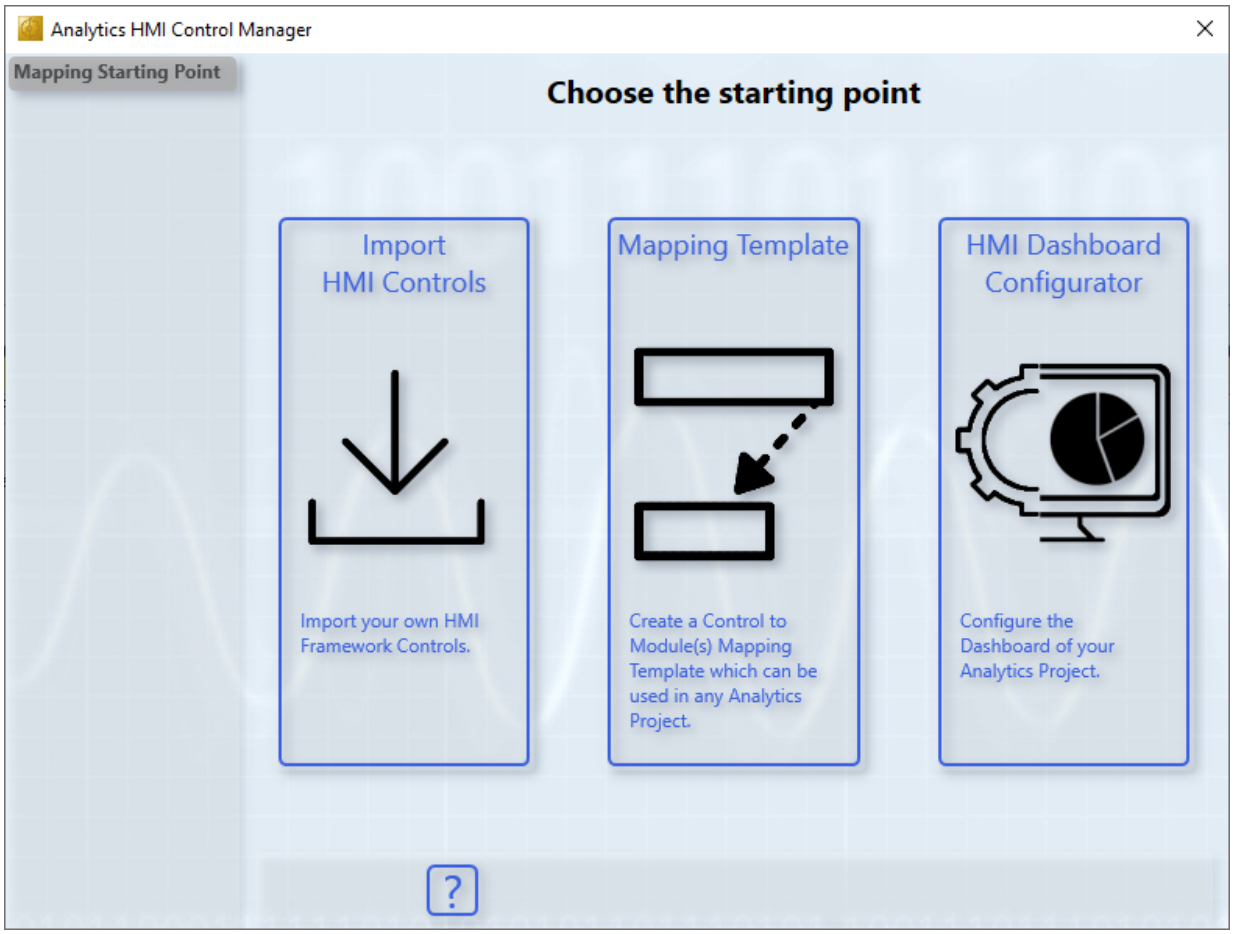


その他のヘルプ情報は、ウィザード内の**?**マークから確認できます。

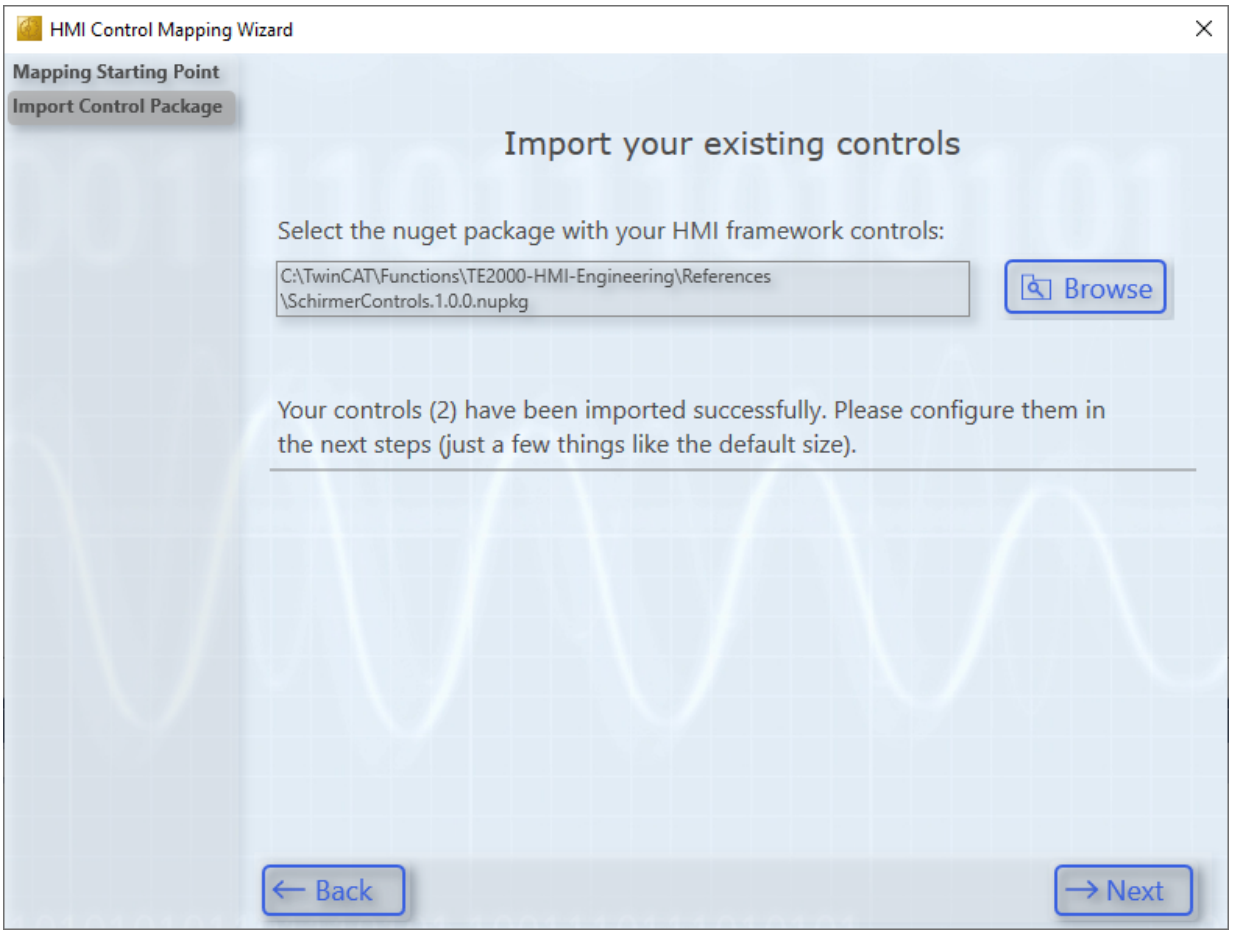
1.独自のコントロールを追加

TwinCAT 3 HMIでは、独自のHMIフレームワークコントロールを作成し、NuGetパッケージとしてエクスポートできます。

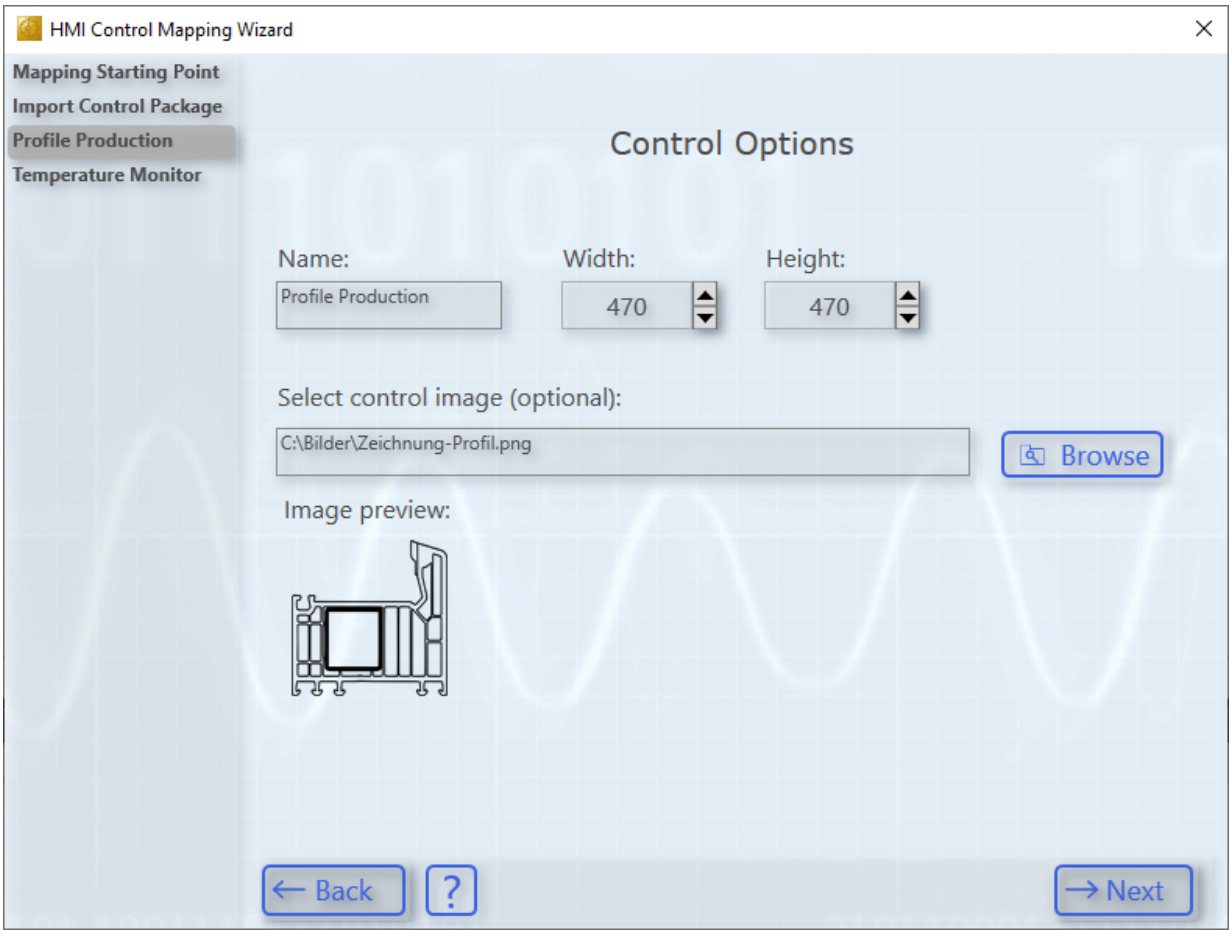
1. 独自のフレームワークコントロールを Analytics モジュールに割り当てるには、**Import HMI Controls**をクリックします。



2. **Browse**からNuGetパッケージを選択します。



3. 次に、コントロールの名前、サイズ、画像を選択します。次に、**Next**をクリックし、最後のコントロールで**Create**をクリックします。

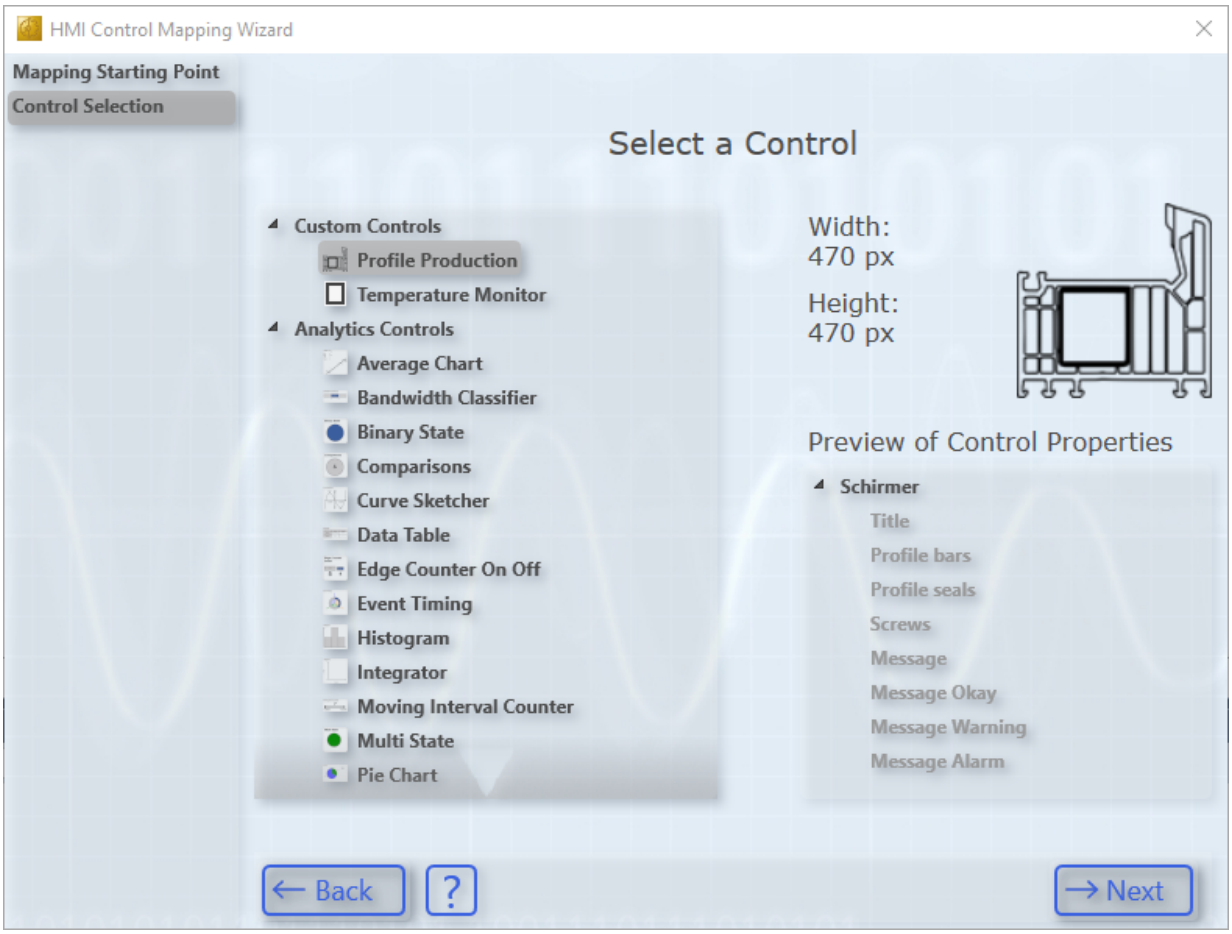


⇨ 自動でリダイレクトされ、コントロールとAnalyticsモジュール間のマッピングを作成できます。

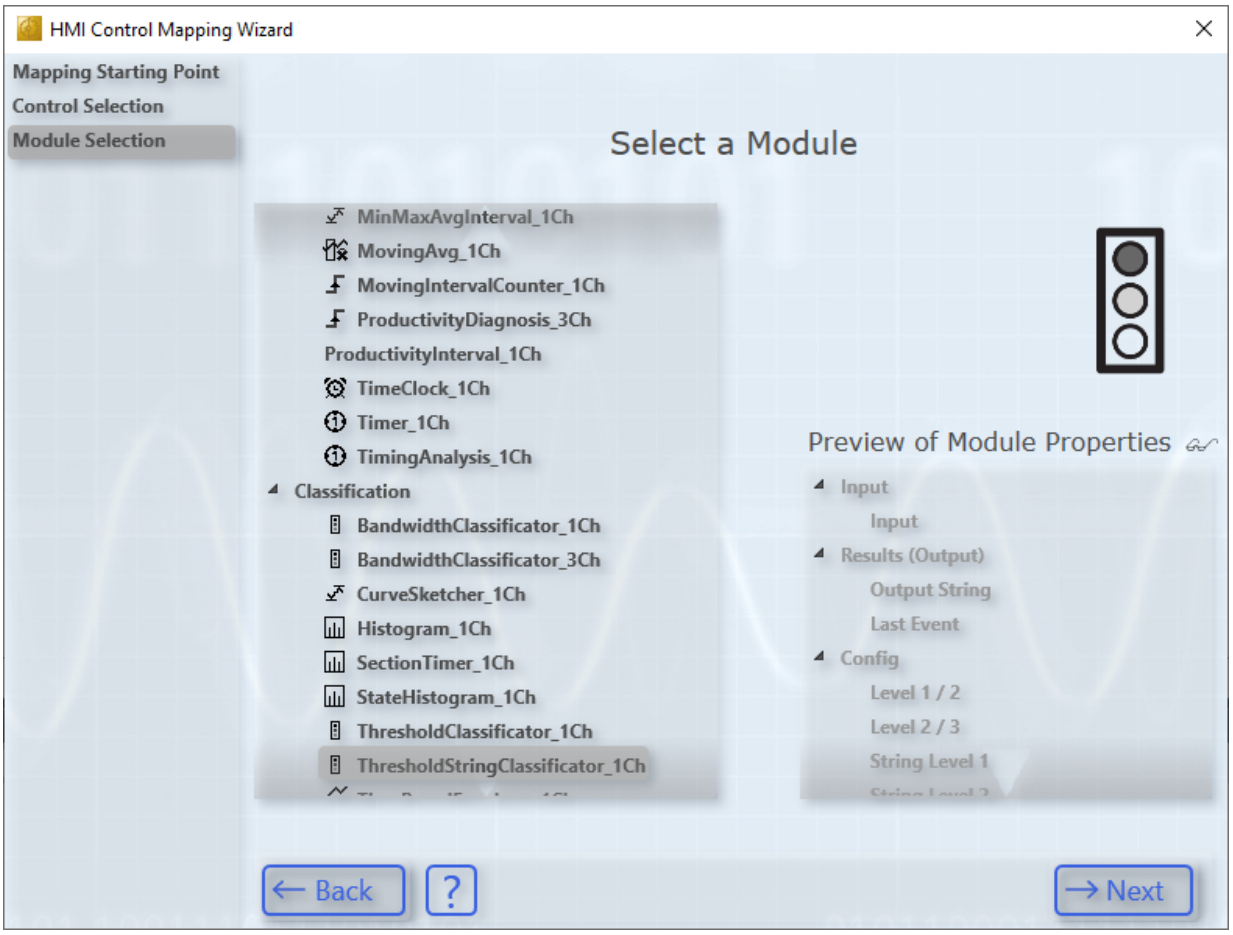
2. a) コントロールとモジュールのマッピング（1.の続き）

2.b)では、このステップを「バイナリ状態」制御を使ってより詳しく説明します。

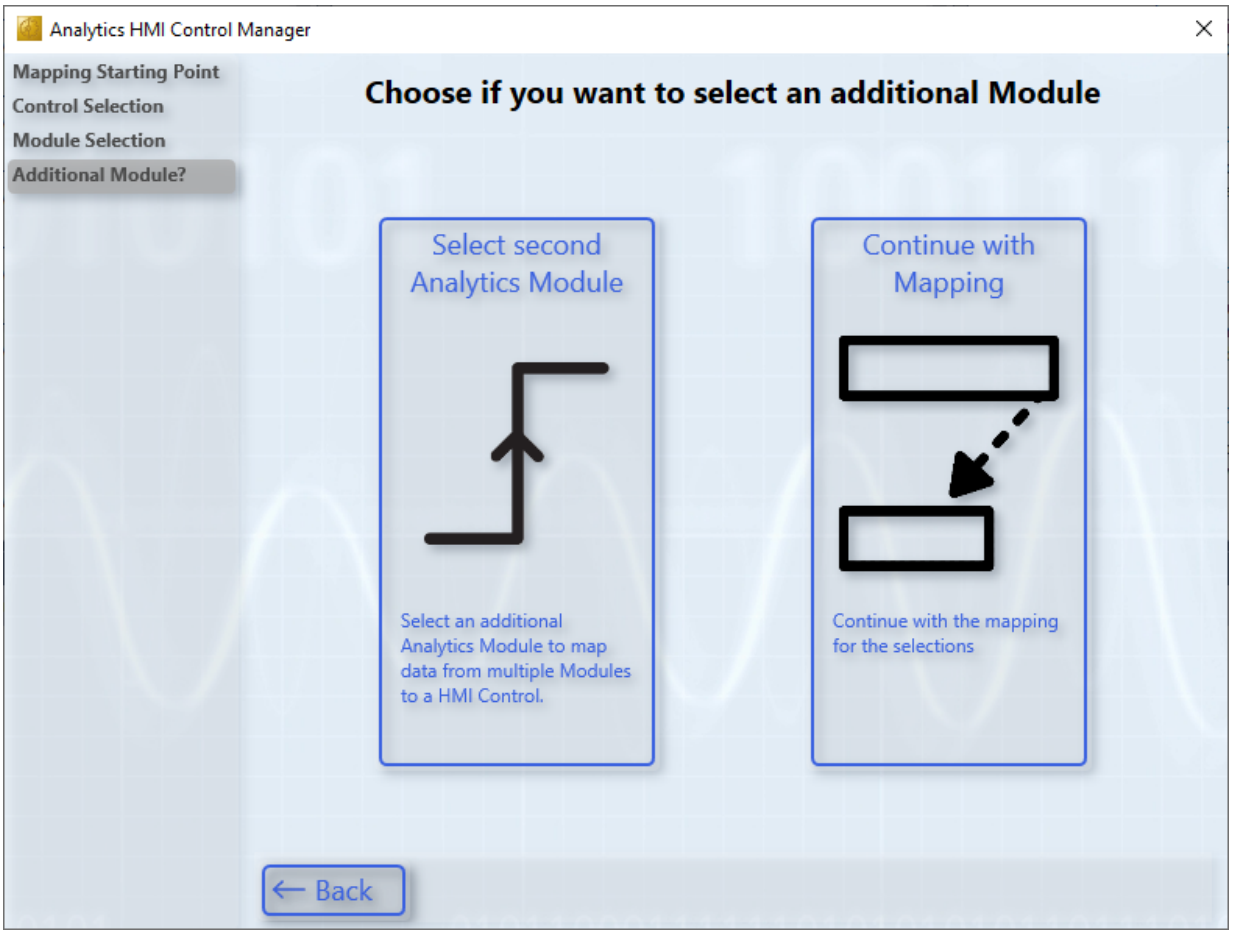
1. まずコントロールを選択します。



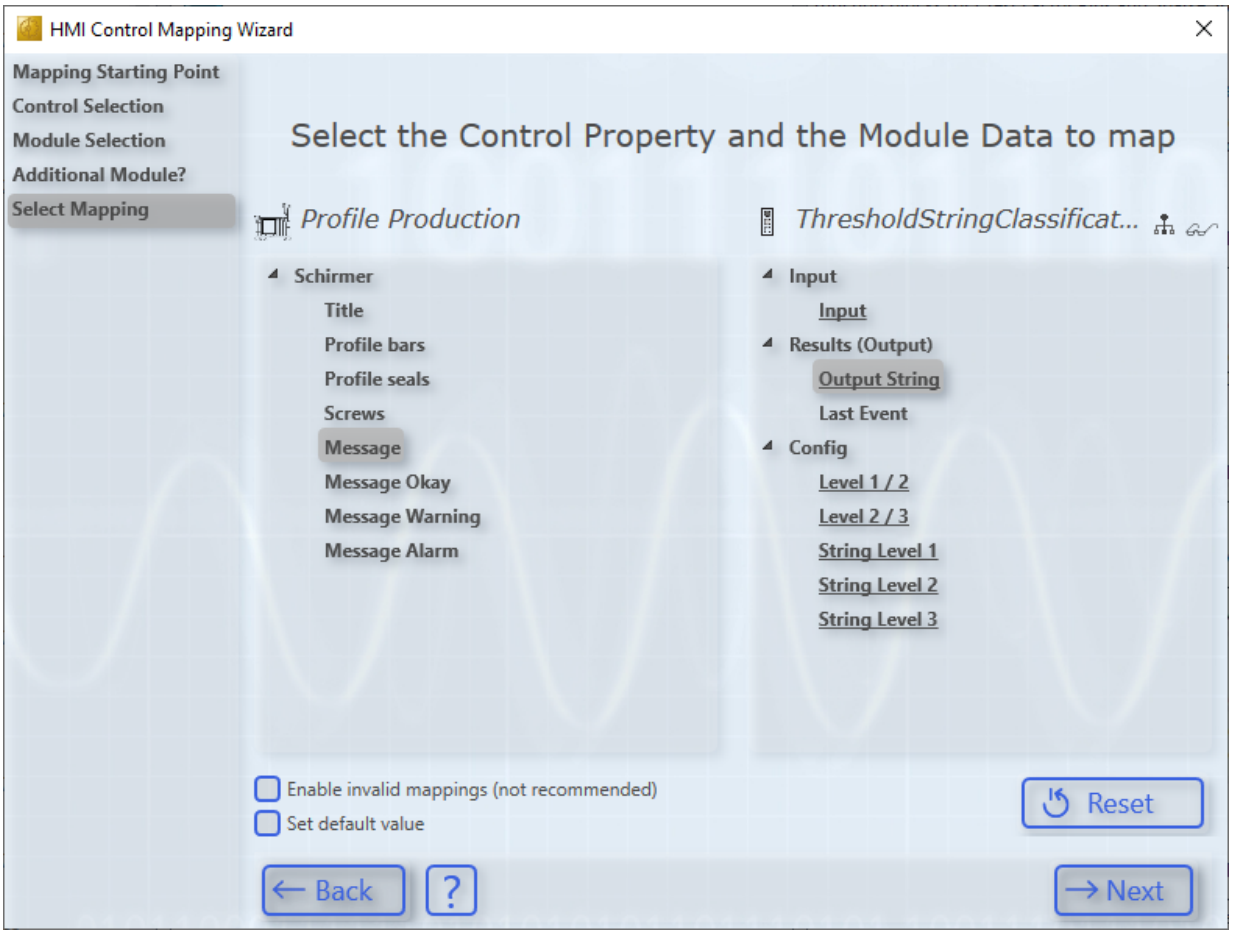
2. 次に、コントロールをアサインするモジュールを選択します。



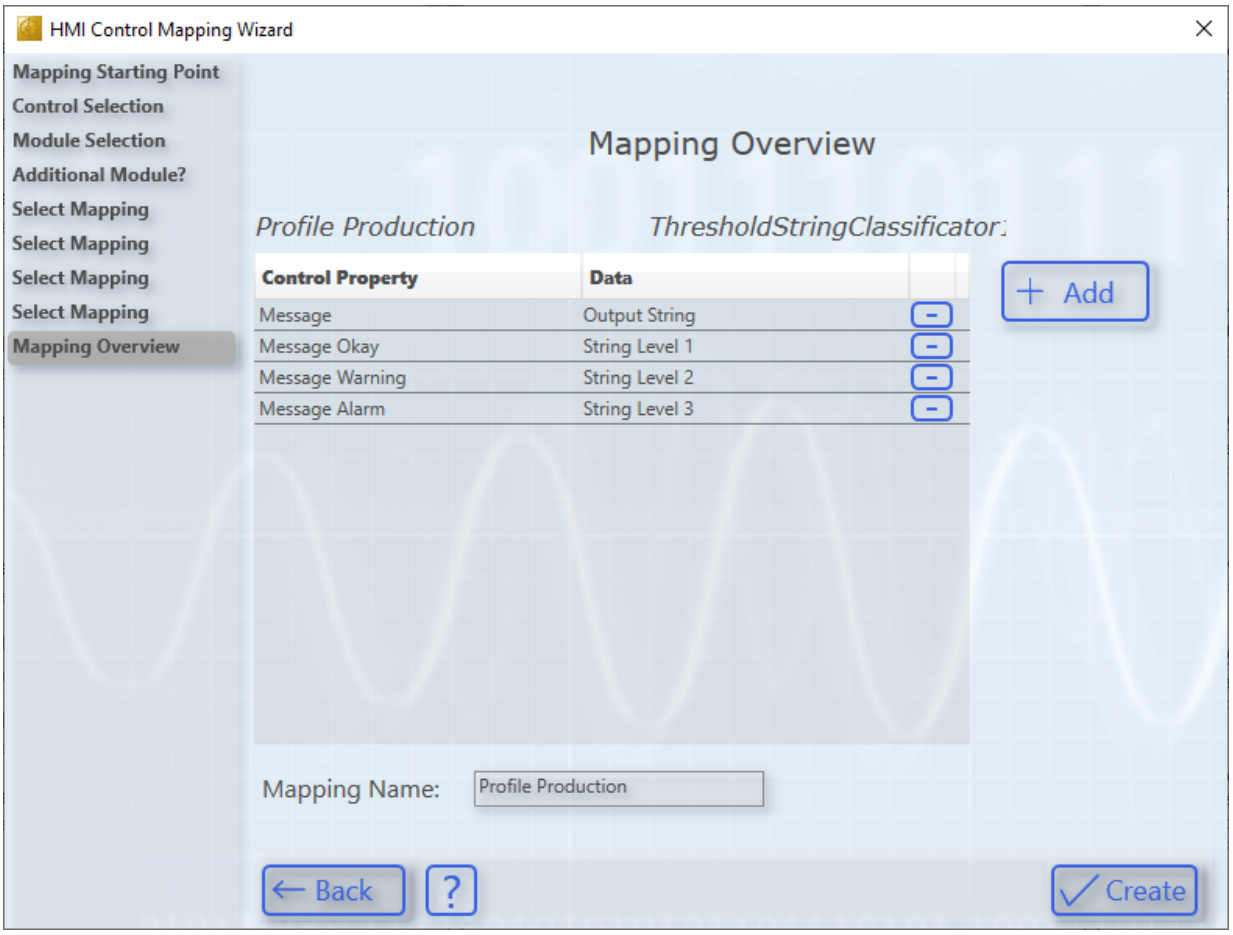
3. **Continue with Mapping**を選択します。



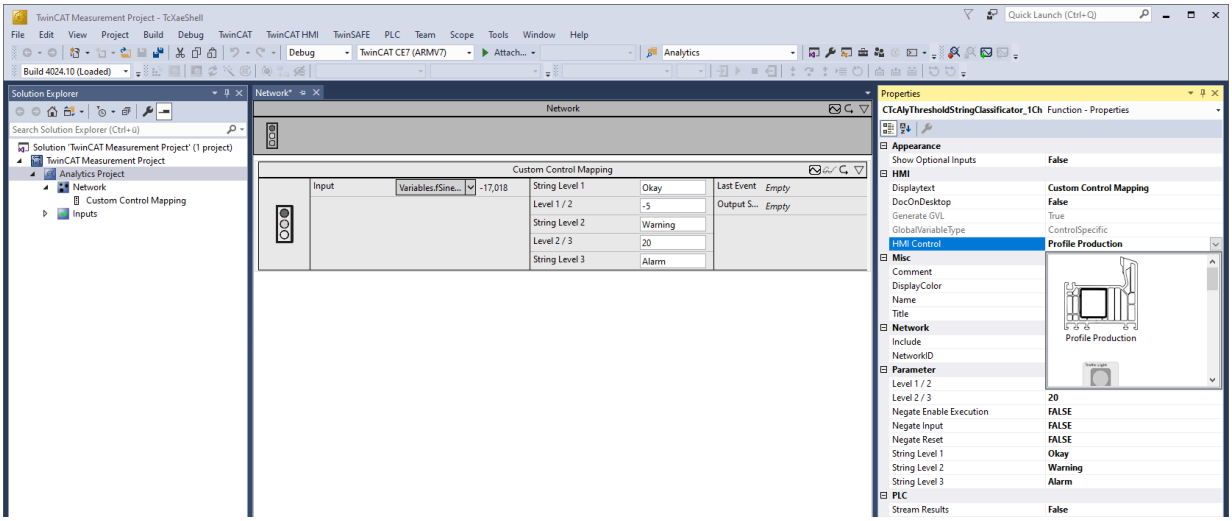
4. 次のステップでは、制御入力をモジュールデータに接続します。

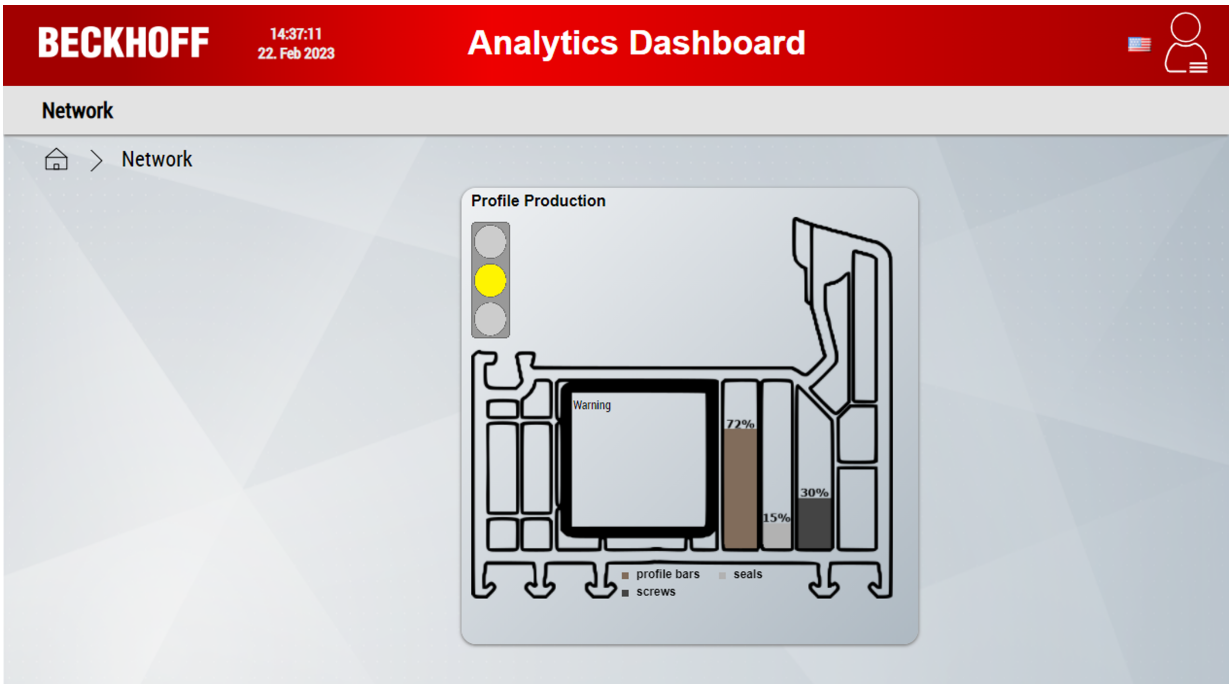


5. 最後に、**Create**をクリックしてAnalyticsワークベンチにマッピングを追加します。



6. これでウィザードを閉じれば、自動的にマッピングがモジュールで利用できるようになります。HMIコントロールを選択し、ダッシュボードを作成します。



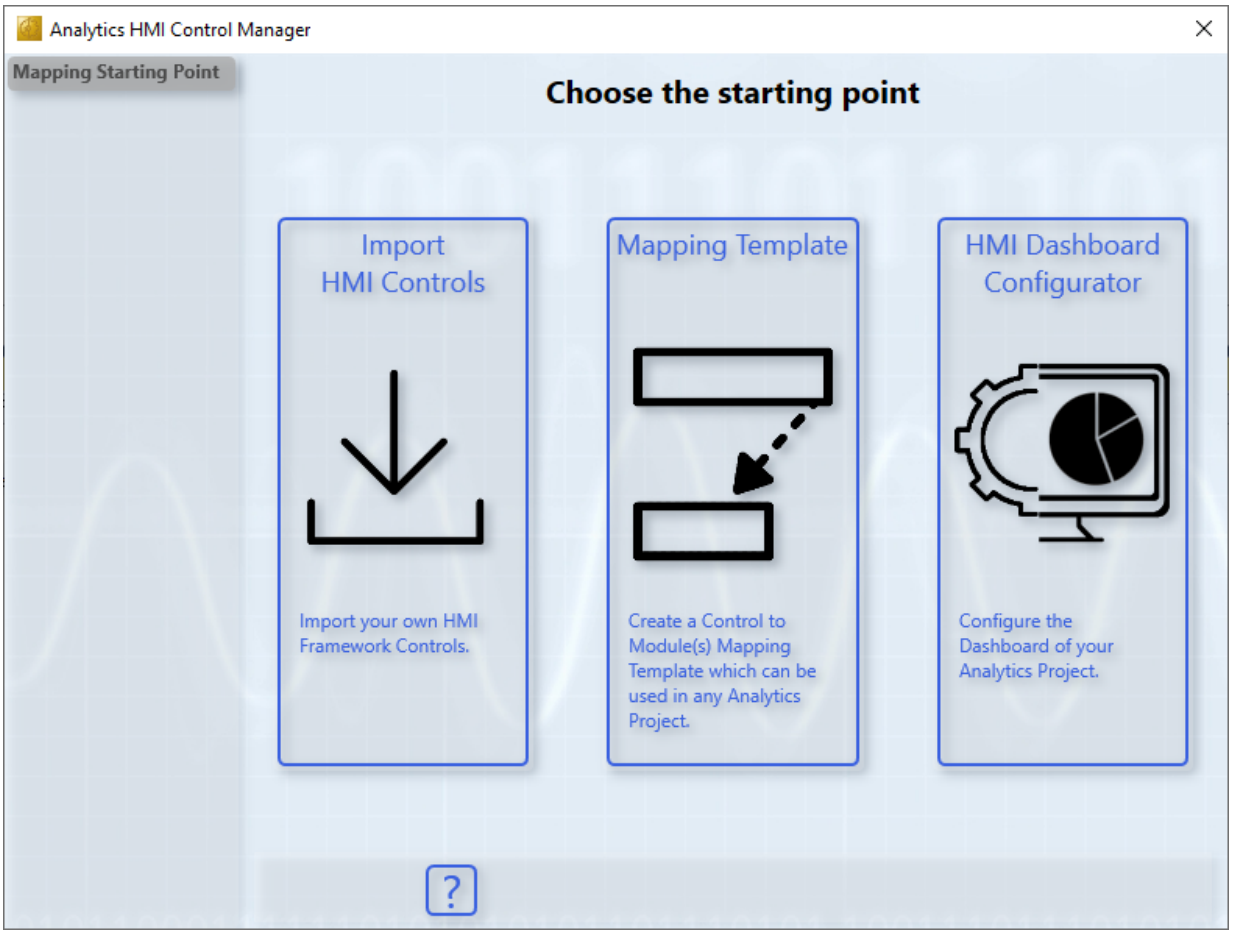


⇨ ダッシュボードの準備が整いました。

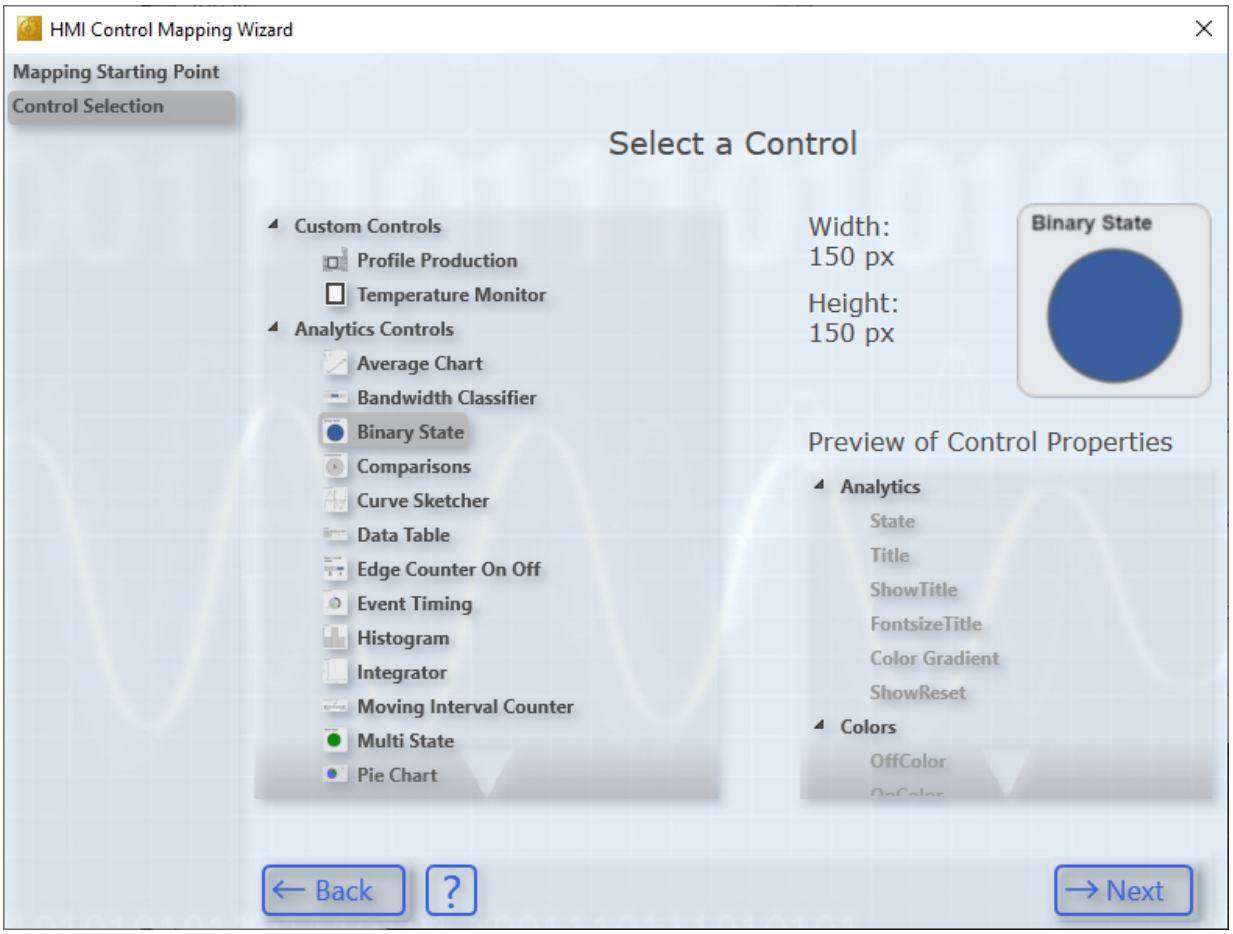
2.b) コントロールとモジュールのマッピング（独自のコントロールなし）

2.a)と同様に、本項ではコントロールとモジュールをマッピングする方法を説明します。既存のAnalyticsコントロールを例にしています。この例は、実際にテストできます。

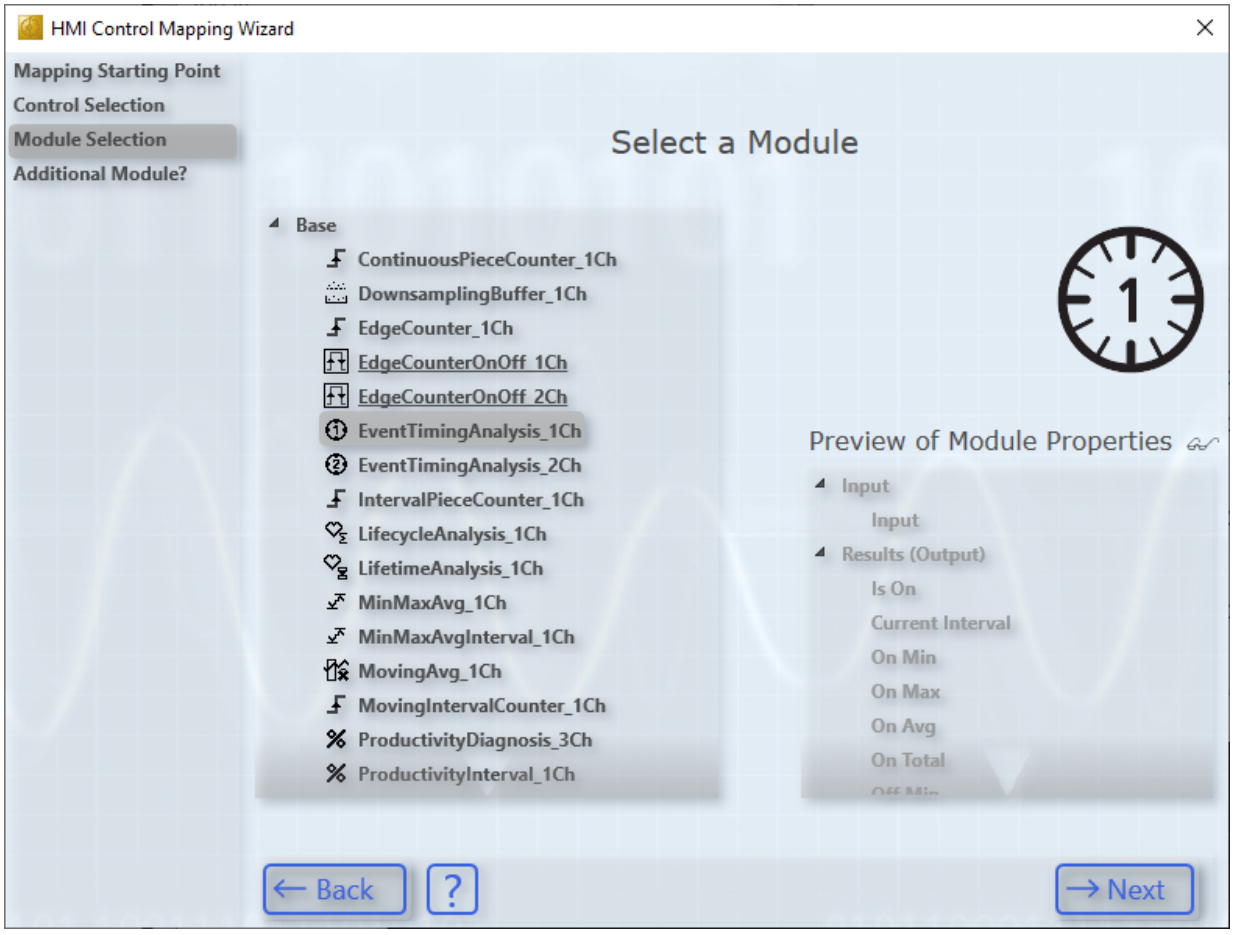
1. まず、**Mapping Template**の項目を選択します。



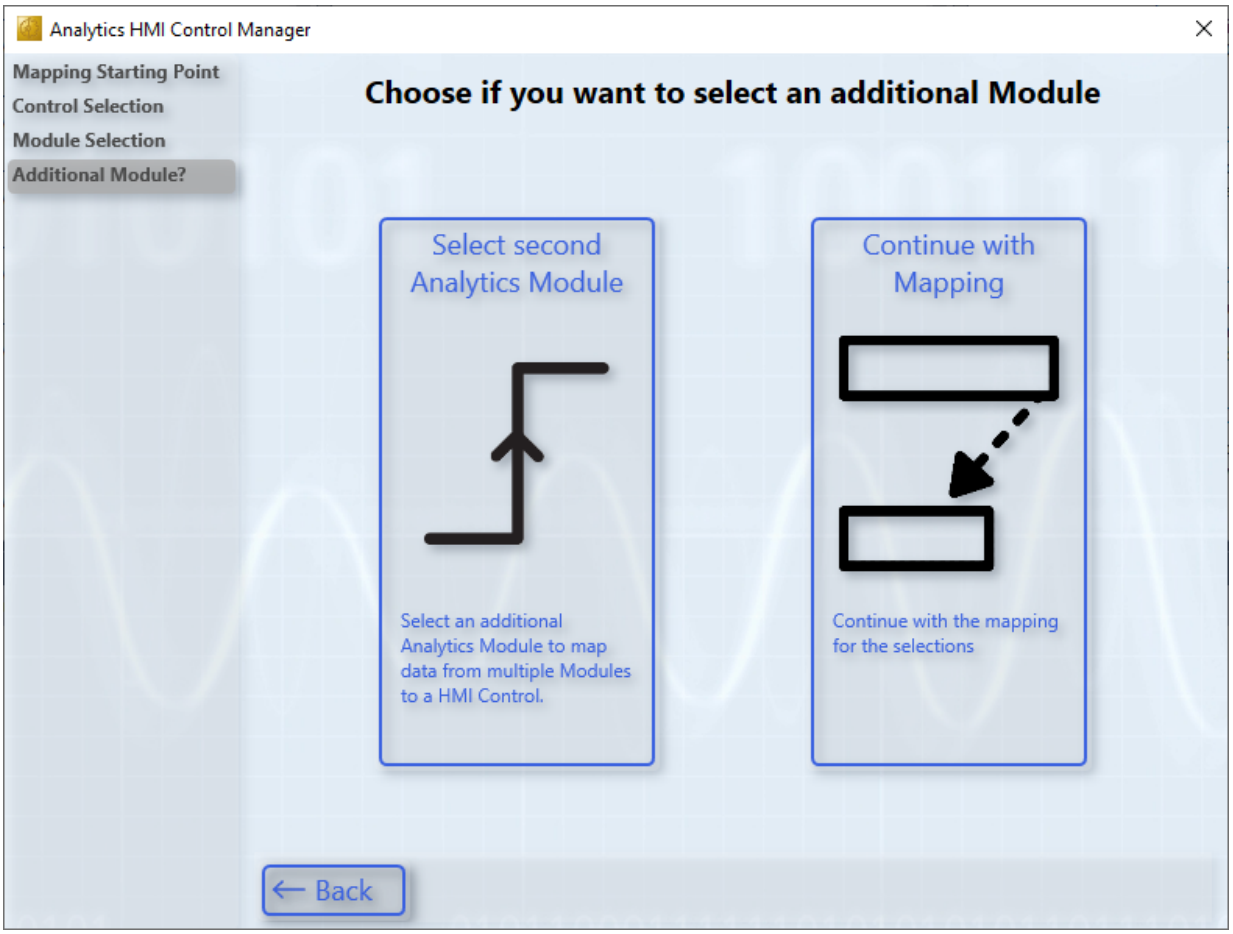
2. コントロールを選択します。コントロールのプロパティは右側に表示されます。また、ダッシュボード上のコントロールのサイズも表示されます。



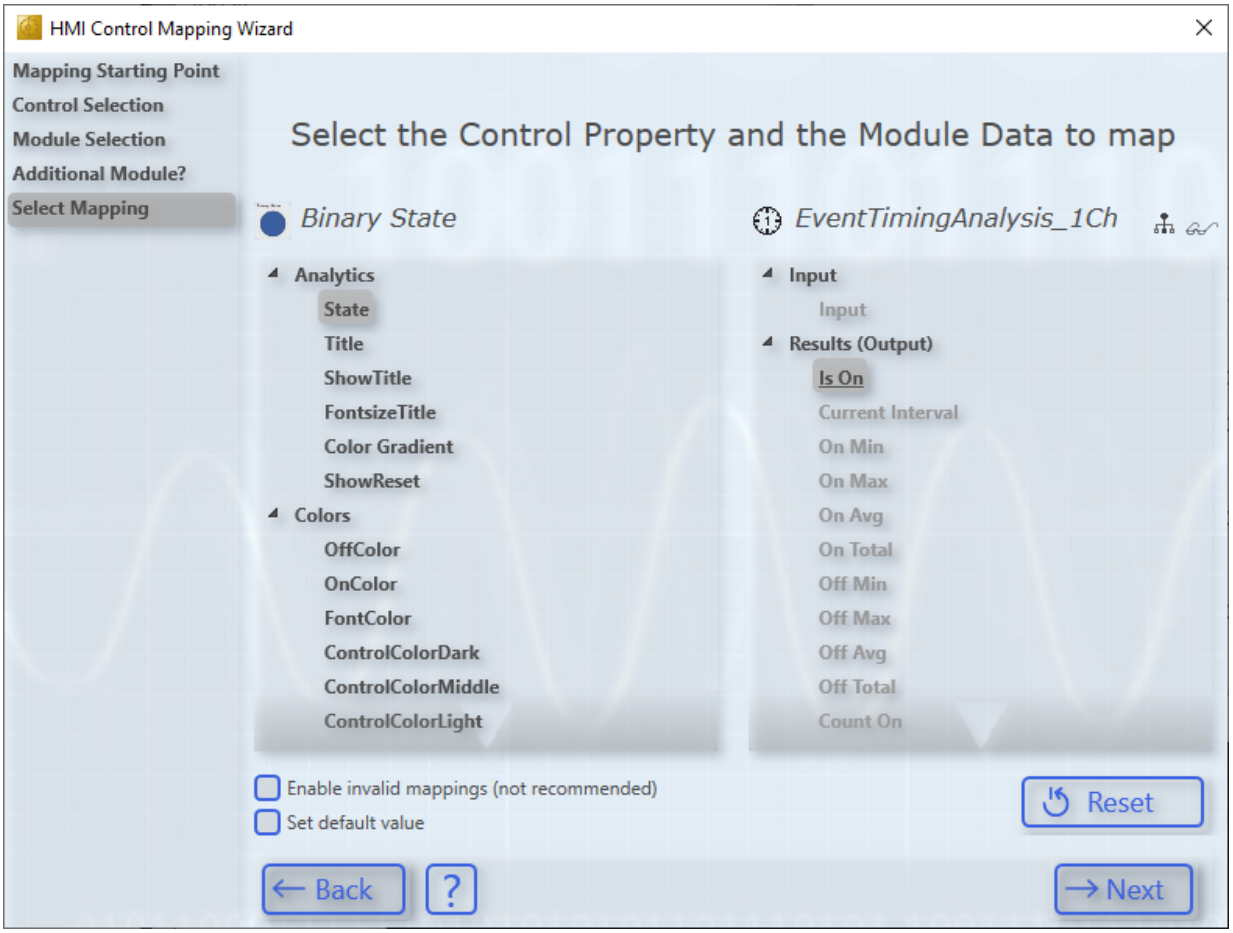
3. データを表示したいモジュールを選択します。入力、出力、パラメータはプレビューとして表示され、選択したコントロールのプロパティにリンクできます。
既にマッピングされているモジュールには下線が引かれています。これらのマッピングは編集することもできます。



4. 将来的には、コントロールで異なるモジュールのデータを表示できます。現段階では不可能なので、**Continue with Mapping**を選択してください。

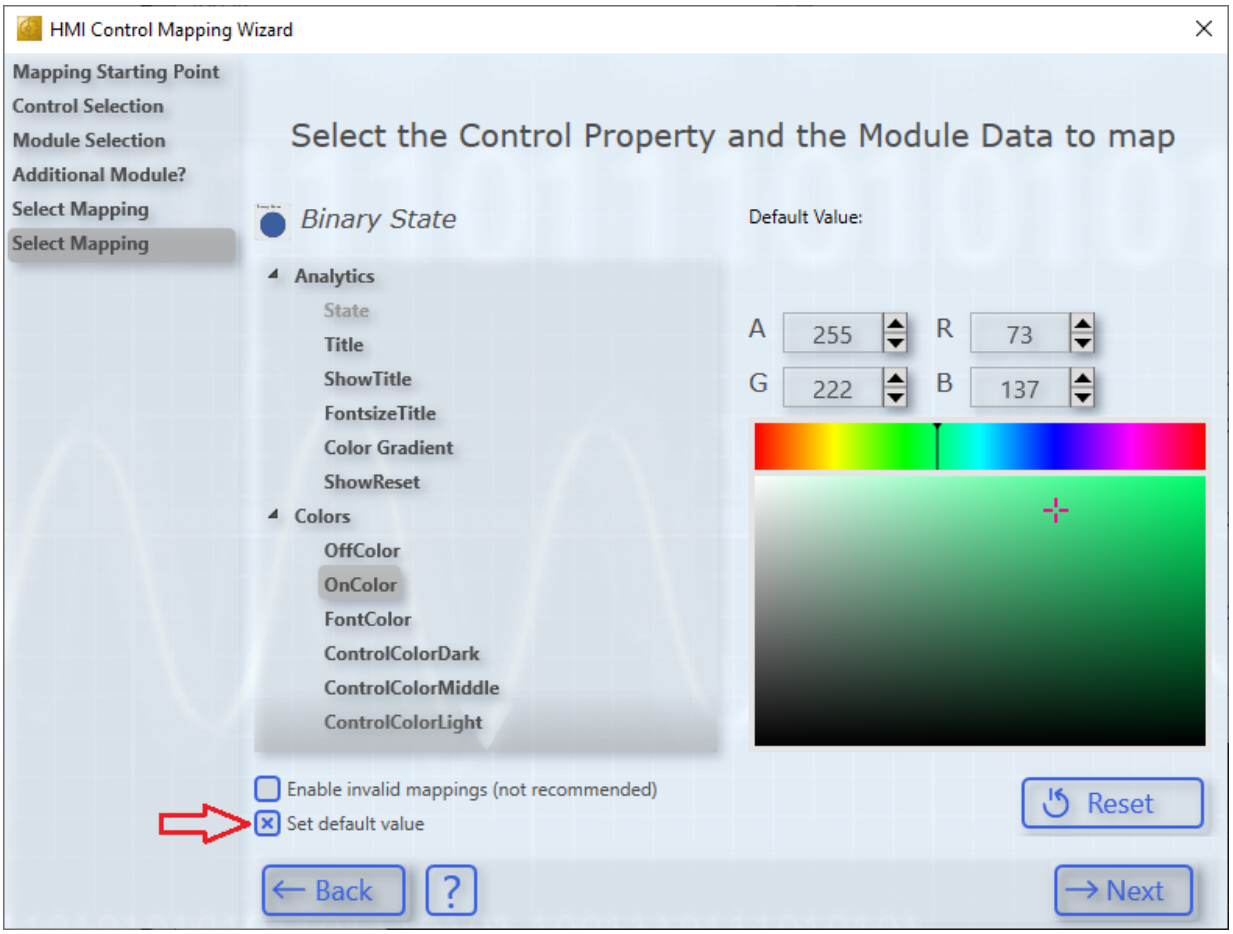


5. 次に、マッピングするコントロールプロパティを選択します。データ型によって代入できない変数は、全てグレー表示されます。それ以外は全てマッピングできます。
   次に、コントロールプロパティにマッピングしたいモジュール変数を選択します。

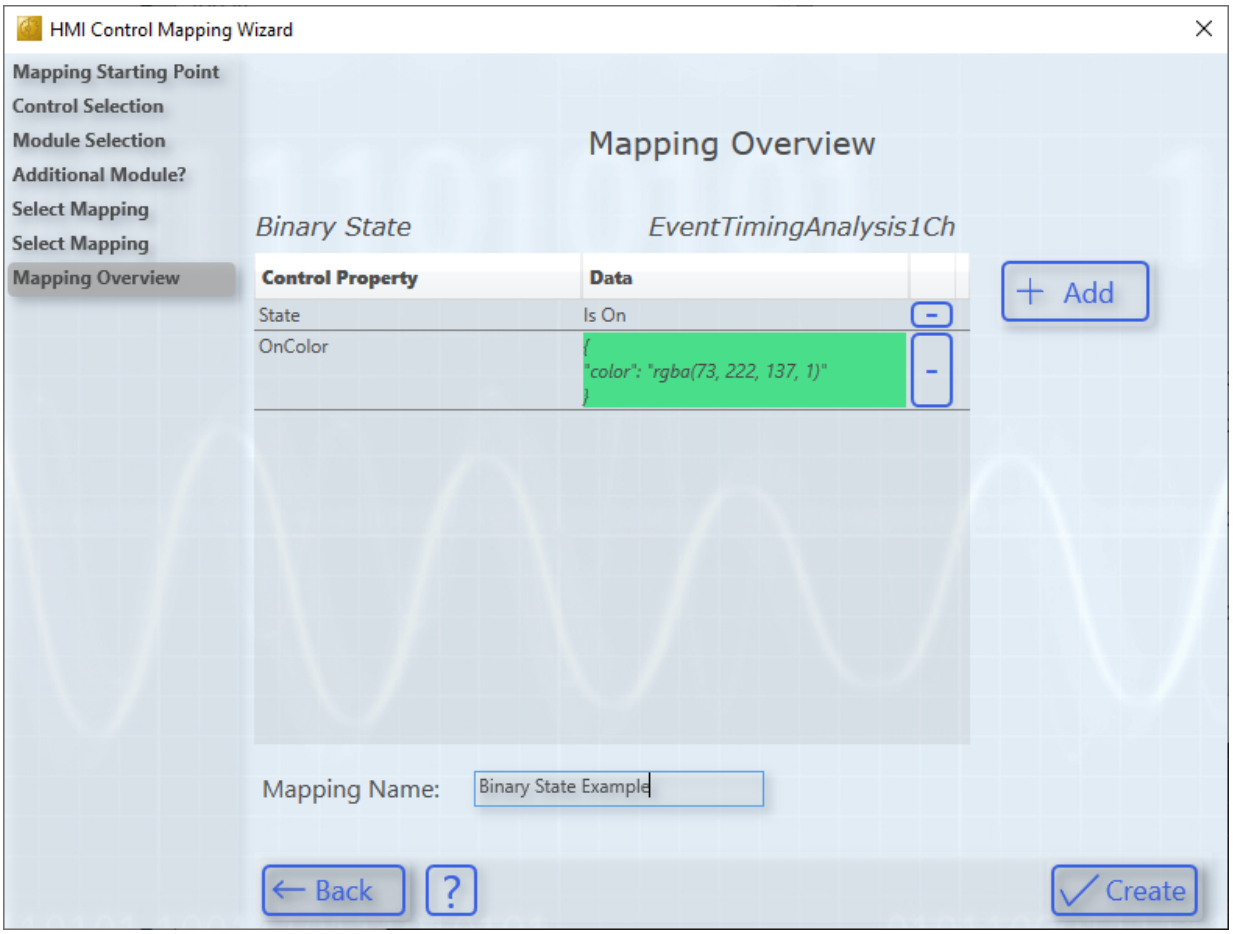


6. 同様に、**Set default value**チェックボックスをチェックすると、コントロールにデフォルト値を割り当てることができます。これは、例えば、デフォルトの色を変更したり、Analyticsコントロールにタイトルを表示しない場合に「ShowTitle」のブール値を「False」に設定するためなどに使用できます。

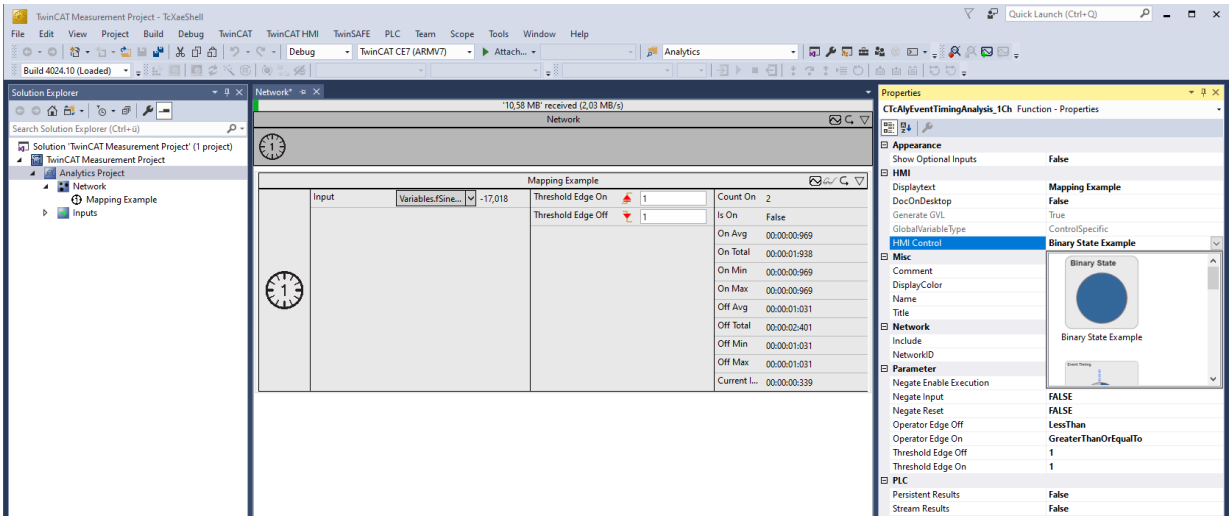
この例では、「OnColor」を変更しています。データ型に基づき、特定の選択オプションが表示されます。

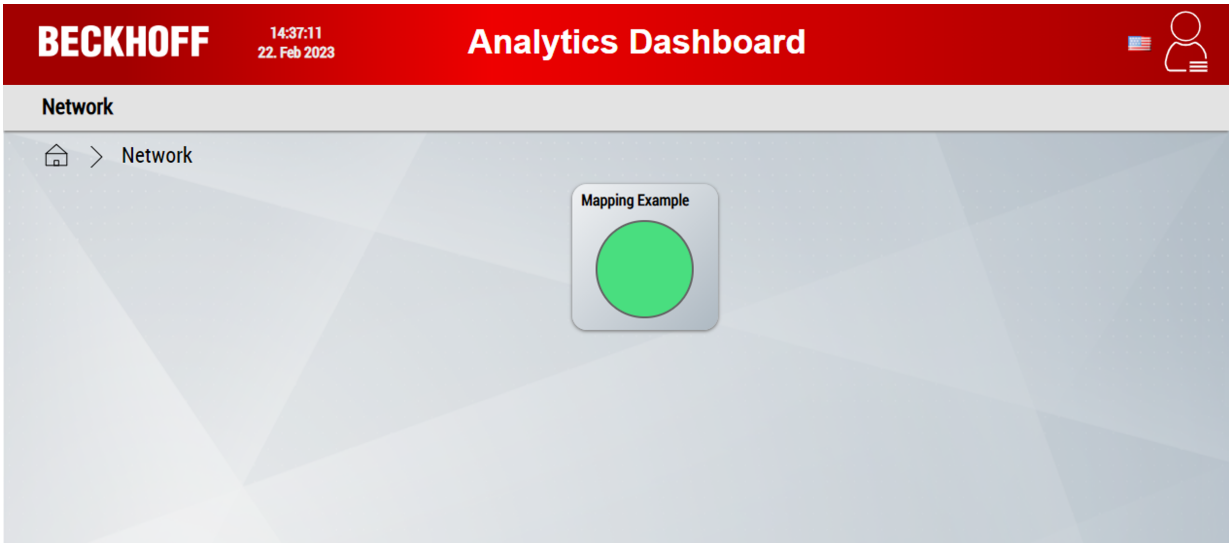


7. **Add**で全てのエントリー追加が完了したら、**Create**でマッピングを完了します。分かりやすいマッピング名を付けることを推奨します。



8. ウィザードを閉じると、マッピングは自動的にモジュールで利用できるようになります。HMIコントロールを選択し、ダッシュボードを作成します。



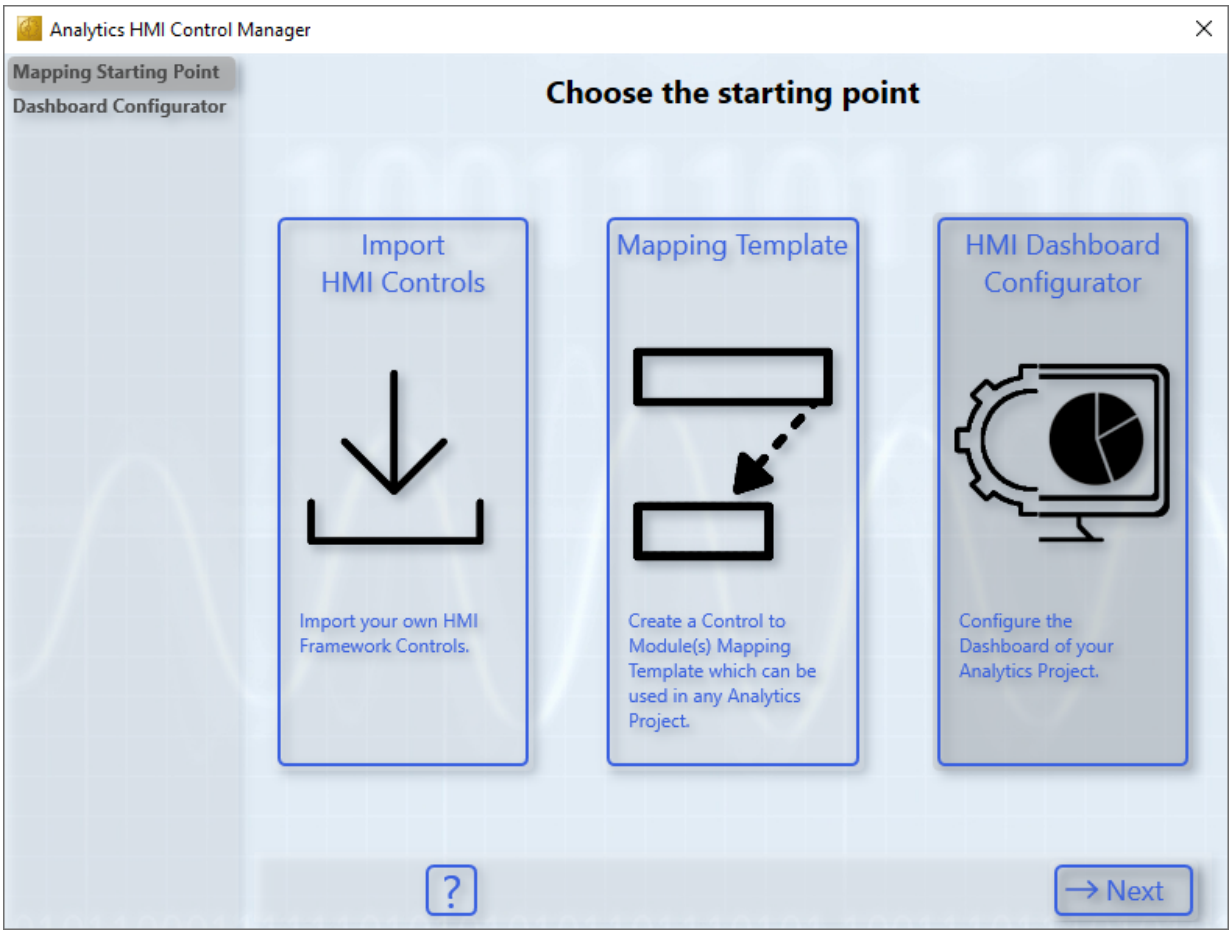


⇨ ダッシュボードの準備が整いました。

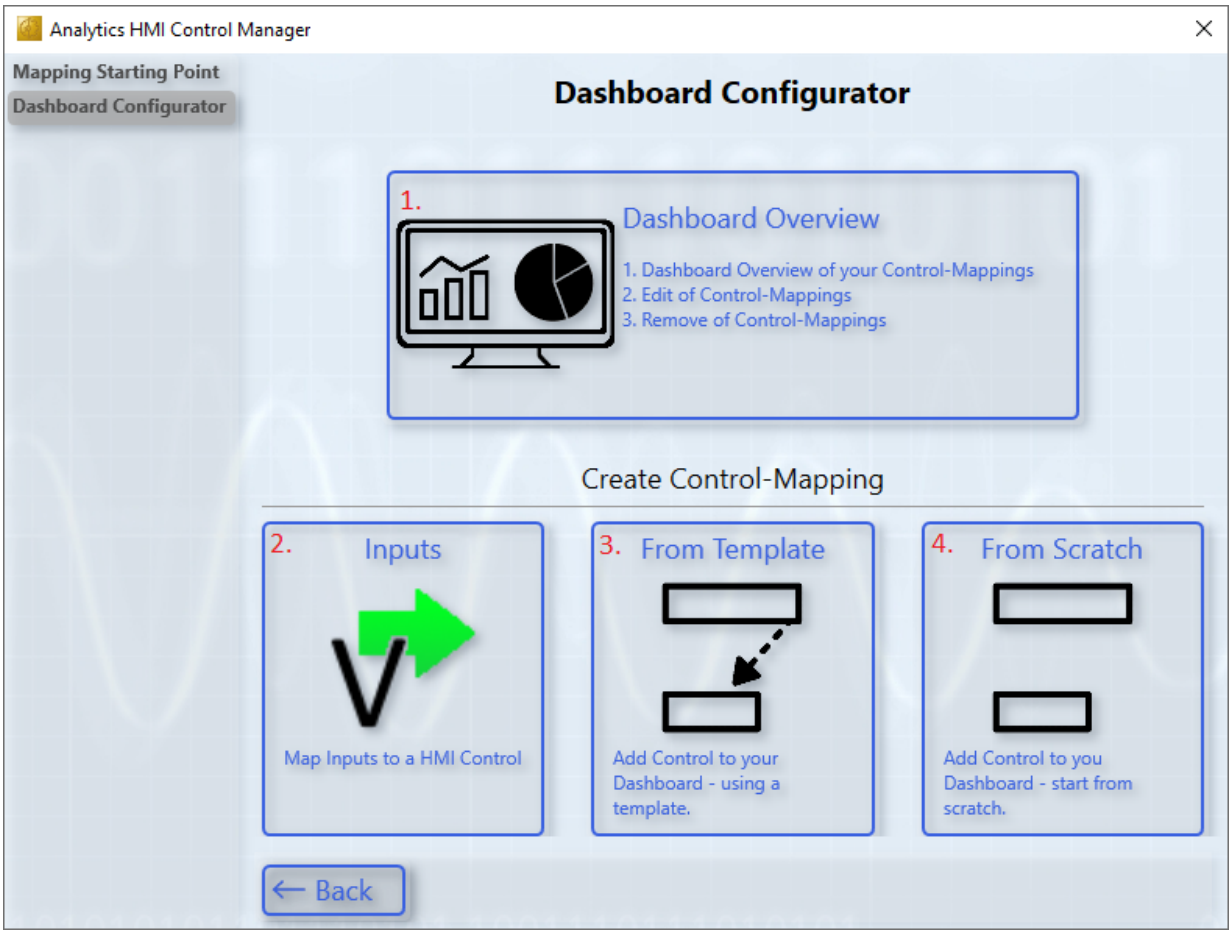
3.コントロールとモジュールインスタンスのマッピング

コントロールをモジュールクラスにマッピングするだけでなく、プロジェクトのモジュールインスタンスを直接リンクすることもできます。これはウィザード経由のみでなく、Analyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理 [▶ 428] からも可能です。

1. ウィザードの冒頭で、**HMI Dashboard Configurator** をクリックします。



2. いくつかの選択肢が表示されます。



1. 全てのコントロールの概要と編集オプション
   - また、この機能は、Solution Explorerでコントロールをダブルクリックすることで利用できます。
2. 入力データを表示するコントロールの追加
   - この機能はAnalyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理 [▶ 428] からも可能です。
3. モジュールデータを表示するためのコントロールの追加（テンプレートから選択）
   - この機能は、*Properties* ウィンドウで個々のモジュールごとに設定することもできます。これは、複数のモジュールのデータを表示する唯一の方法です。
   - テンプレートのどのデータがリンクされているかを直接確認することができ、調整できます。
4. モジュールのデータを表示するコントロールの追加（完全手動）
   - この機能はAnalyticsプロジェクトにおけるダッシュボードの構造とコンテンツ管理 [▶ 428] からも可能です。

各ポイントについては、ウィザードのステップに従ってください。手順は、項目2 「コントロールとモジュールのマッピング」に類似しています。

6.9.3 ユーザ管理とアクセス権の設定

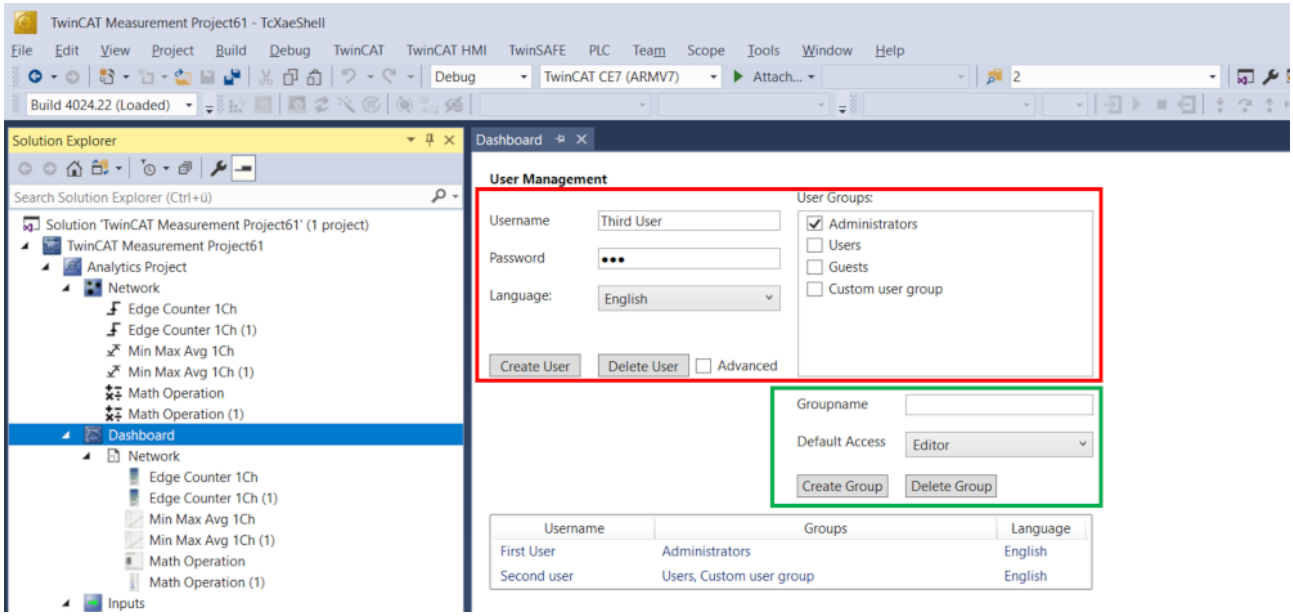
ユーザおよびユーザグループは、user managementで個別に作成可能です。コンテンツやコントロールへのアクセス権は、ユーザグループごとに定義できます。

ユーザ管理とアクセス権は、Dashboard バージョン 2.0 以降からのみ HMI プロジェクトに転送されることに注意してください。

6.9.3.1 Analyticsプロジェクトにおける定義

定義されたユーザ、ユーザグループ、アクセス権は、コード生成中にHMIプロジェクトに自動的に転送されます。ユーザやアクセス権は、生成後にウェブブラウザから変更できます。

6.9.3.1.1 ユーザ管理



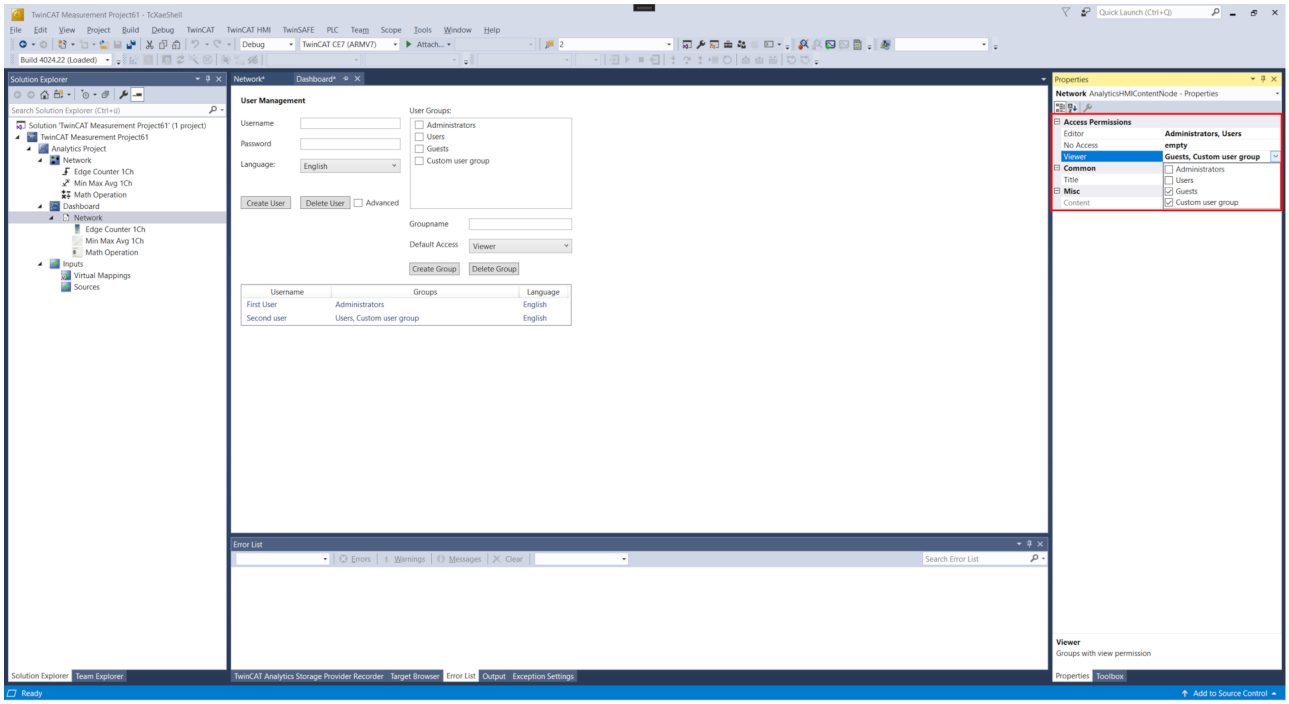
ユーザ（赤枠）は、Analytics プロジェクト・ダッシュボードノードの下に作成できます。新規ユーザには、ユーザ名とパスワード、言語を設定する必要があります。さらに、ユーザに1つまたは複数のユーザグループを割り当てる必要があります。

デフォルトでは、以下のユーザグループがあります：

- Administrators
- Users
- Guests

より細かく分類するため、追加のユーザグループを作成できます（緑枠）。そのためには、名前とアクセス権を入力する必要があります。

6.9.3.1.2 アクセス権の設定



アクセス権は、ダッシュボードノード以下のオブジェクト（コンテンツとコントロール）に対して設定できます。アクセス許可（赤枠）は、ユーザグループごとに設定できます。

以下のアクセス権が使用できます：

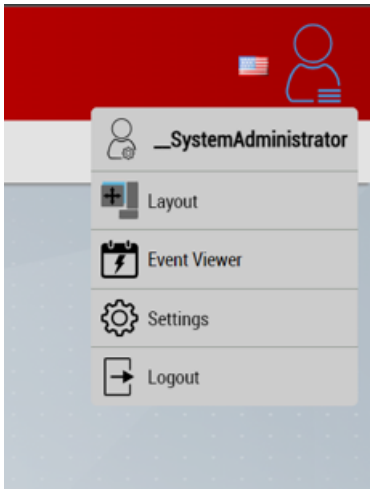
- Editor：オブジェクトへのアクセスを許可、変更（コンテンツオブジェクトのレイアウト調整など）も可能。
- Viewer：オブジェクトへのアクセスを許可、変更は不可。
- No Access：Solution Explorerのオブジェクトおよびそのオブジェクトに従属するオブジェクトへのアクセス不可。

コンテンツオブジェクトのアクセス権は、ウェブブラウザで調整できます。

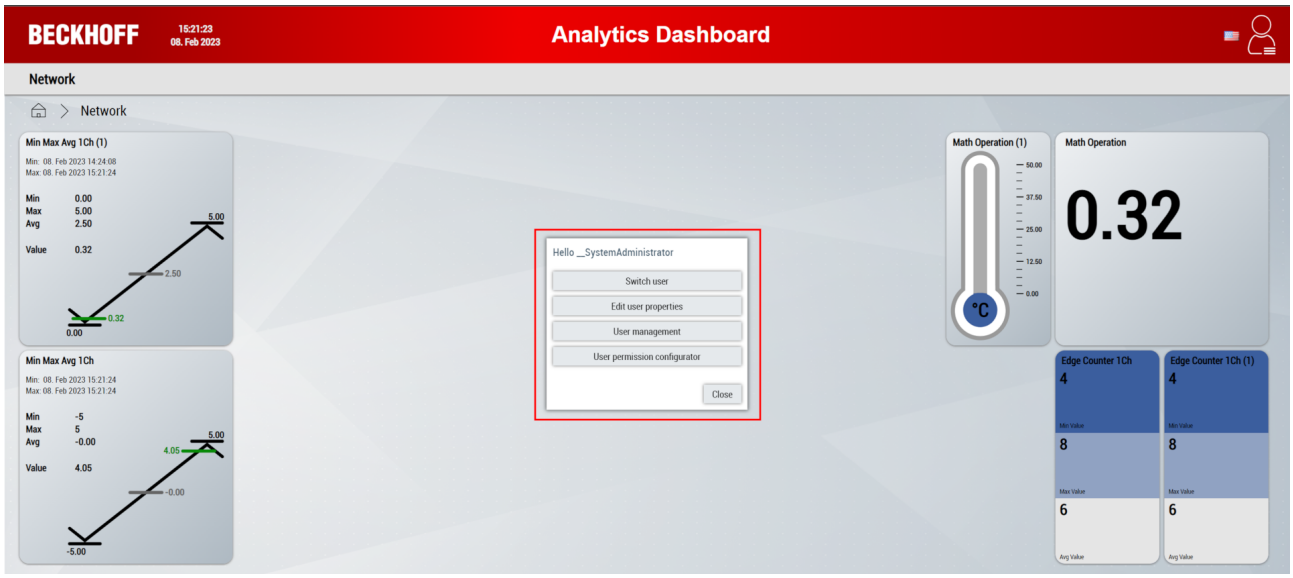
6.9.3.2 ウェブブラウザでユーザ管理とアクセス権をカスタマイズする

メニューコントロールとユーザ管理・アクセス権の設定は、Dashboard バージョン 2.0 以降でのみ利用可能です。

ユーザ設定



メニューコントロールを使用して、ユーザとアクセス権を変更できます。これを行うには、ユーザ名をクリックしてください（メニューの一番上）。



以下のオプションはポップアップ（赤枠）で表示されます：

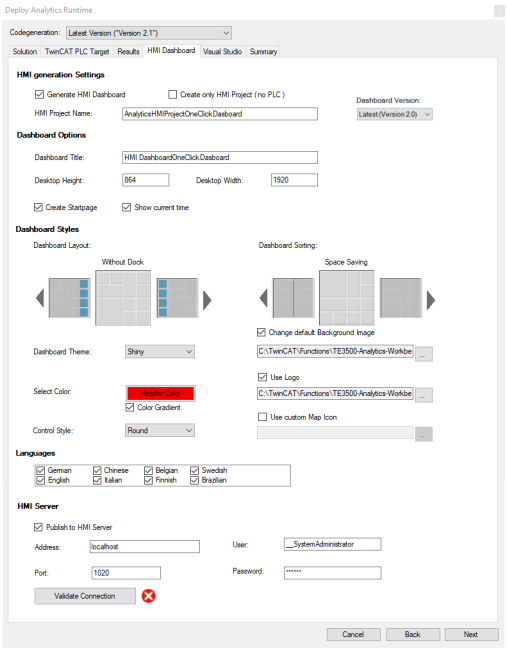
- **Switch user:**ログインユーザを変更します。
- **Edit user properties:**ユーザのプロパティを変更します。
- **User management:**新規ユーザの作成、既存ユーザの削除、ユーザプロパティの設定を行います。
- **User permission configurator:**コンテンツオブジェクトへのユーザグループのアクセス権を設定します。

新しいユーザを作成する場合、ユーザに個別に設定されたレイアウトが引き継がれることに注意してください。

6.9.4 ダッシュボードの設定

HMI Dashboard タブ

HMI Dashboardタブには、ダッシュボードの全ての設定が含まれています。



設定

HMI作成の設定

Generate HMI Dashboard	HMI ダッシュボードの作成を有効にします。この設定により、**TwinCAT PLC Target**タブの**Create Bootproject**と **Activate PLC Runtime**のチェックボックスが自動的に有効/無効になります。
Create only HMI Project (No PLC)	有効にすると、HMIプロジェクトが1つのみ作成され、PLCプロジェクトは作成されません。
HMI Project Name	TwinCAT HMIプロジェクトの名前。
Dashboard version	ダッシュボードバージョンを選択します。

ダッシュボードオプション

Dashboard Title	ダッシュボードのヘッダーに表示される HMIダッシュボードのタイトル。
Desktop Width	ターゲット画面の幅（単位：ピクセル）。
Desktop Height	ターゲット画面の高さ（単位：ピクセル）。
Create Start-Page	全機械の場所を示す地図を表示するダッシュボードのスタートページを作成します。位置情報は機械管理データから適用されます。
Show current time	ダッシュボードのヘッダーに現在の現地時間を表示する時計を作成します。

ダッシュボードのスタイル

Dashboard Layout	ダッシュボードのレイアウトを定義します。Dockは、モジュールの「Dock」プロパティがTRUEである必要があります。 Dock Left：左列に固定 Dock Right：右列に固定 Without Dock：列の固定なし
Dashboard Sorting	ダッシュボードのソート方法を定義します。 Space Saving:フリースペースなしでコントロールを配置。 Control Type: コントロールタイプ別に配置。 Control Size: 大から小までのサイズ別にコントロールを配置。 Filled: 画面全体が埋まるようにコントロールを配置。

	Network Groups: ネットワーク別にコントロールをグループ化し、まとめて配置。グループ化は定義したレベルから開始します（0 = 全て、1 = 1番目のレベル...）。
Control Distance	グリッド内のコントロール間の最小距離を決定します。
Dashboard Theme	HMI ダッシュボードのモードを定義します。コントロールと背景に影響します： Light: 明るいスキン（デイモード）。 Shiny: 明るいスキンに類似、コントロールにカラーグラデーションあり。 Dark: 暗いスキン（ナイトモード）。
Header Color	ダッシュボードのヘッダーの色を指定します。カラーグラデーションを有効にすると、グラデーションが含まれます。
Control Style	コントロールのスタイルを定義します： Round: コントロールの角が丸い（ラウンド）。 Flat:コントロールの角が直角（フラット）。
Change default Background Image	有効にすると、HMI ダッシュボードに顧客固有の背景画像を設定できます。何も定義されていない場合は、デフォルトの画像が使用されます。
Use Logo	有効にすると、ダッシュボードのヘッダーにロゴが追加されます。顧客固有のロゴ画像を設定できます。

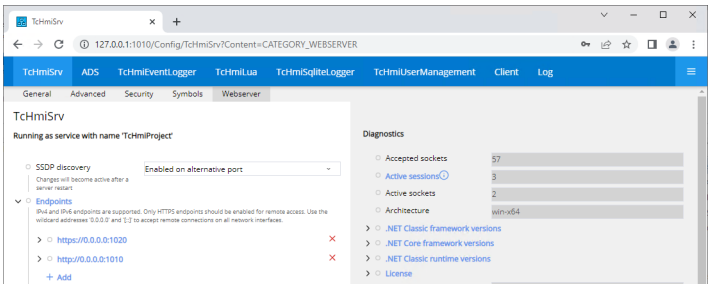
言語

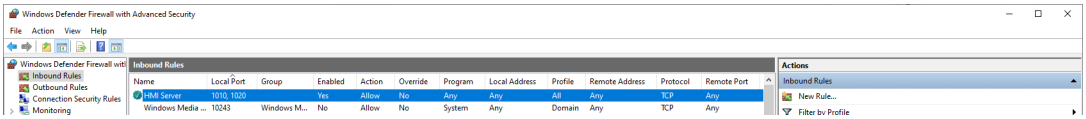
Languages	HMI ダッシュボードで使用可能な言語を選択します。
Select Default Language	ダッシュボードのデフォルト言語。

HMIサーバ

Publish to TwinCAT HMI Server	有効な場合、ダッシュボードをTwinCAT HMIサーバにパブリッシュします。
Address	TwinCAT HMIサーバのIPアドレスまたはホスト名を入力します。
Port	TwinCAT HMIサーバのポートを入力します。デフォルトは暗号化なしの1010。暗号化ありは1020。
User	ユーザ名を入力します。
Password	パスワードを入力します。(管理者パスワードは、HMI サーバの設定ページで一度設定する必要があります。)
Validate Connection	ボタンを押してサーバとの接続を確認します。

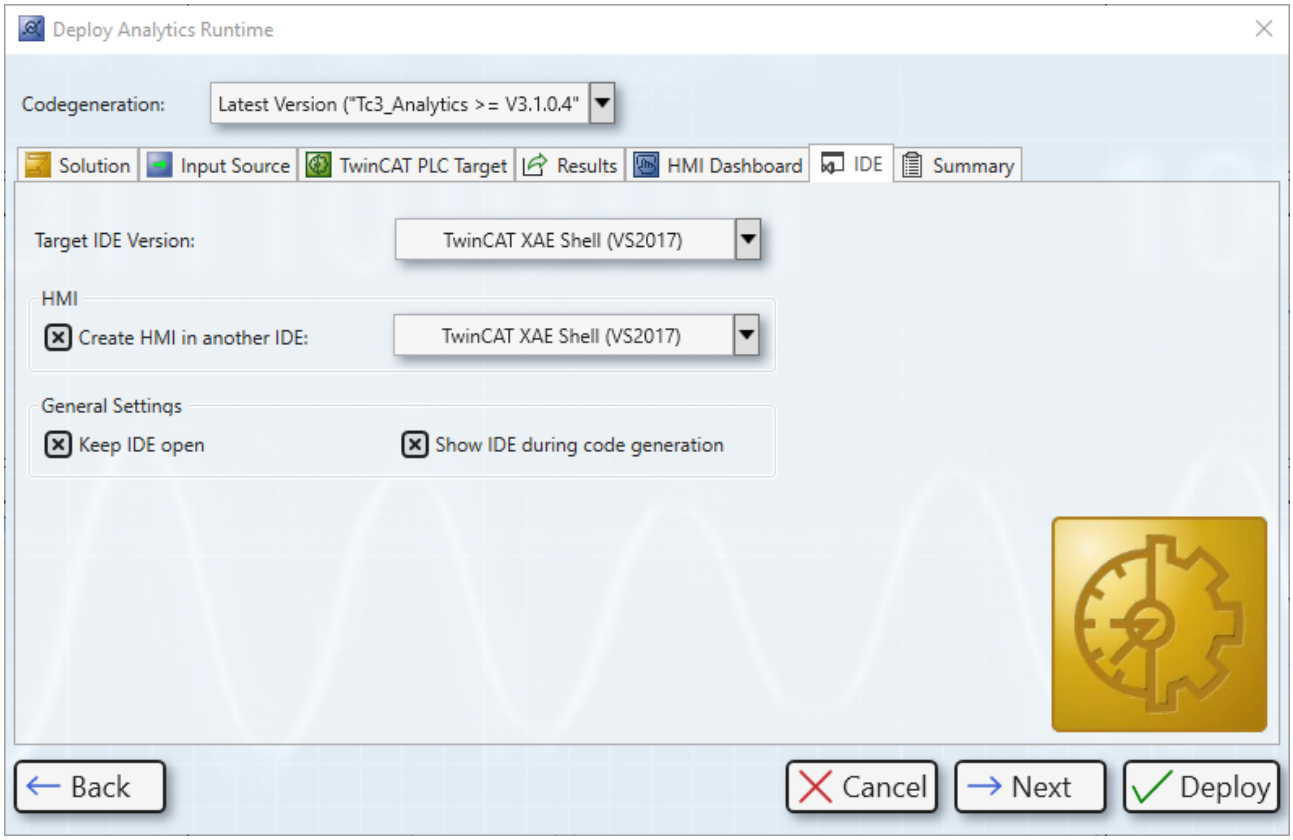
リモート HMI サーバにパブリッシュするには、開発用PCの HMI ポートと IP を HMI サーバに共有する必要があります。また、リモート PC から HMI ポートを含む受信ルールを Windows ファイアウォールに設定する必要があります。





IDEタブ

IDEタブで、PLCとHMIの生成に使用するIDE（Visual Studio®またはTwinCAT XAE Shell）を選択できます。2つの異なるソリューションでHMIとPLCを生成することも可能です。

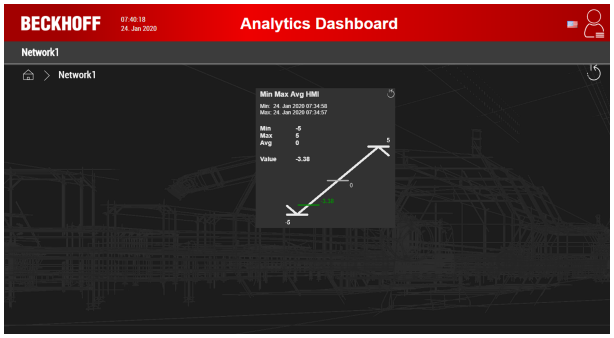


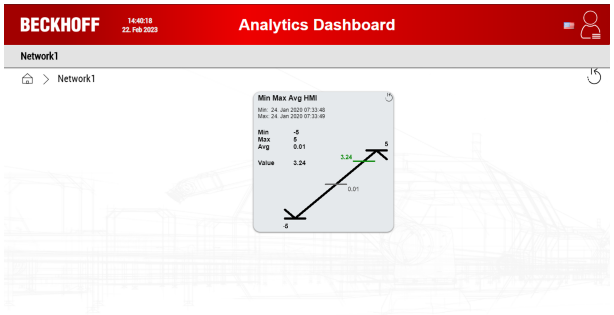
Keep IDE open	有効の場合、生成後もIDEが開いたままになります。
Target IDE Version	PLC および HMI 用の IDE ターゲットバージョン。
Create HMI in another IDE	有効の場合、HMIは2番目のソリューションに生成されます。別のIDEを選択することもできます。
要件：選択したIDEにTwinCATまたはTwinCAT HMIが、インストールされている必要があります。	

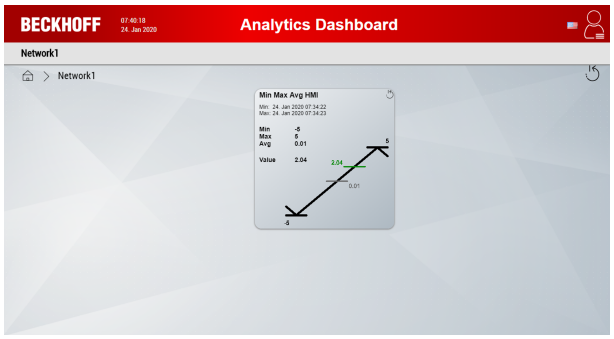
Nextをクリックして**Summary**タブを表示します、次に、**Deploy**をクリックしてダッシュボードの生成を開始します。

イメージ

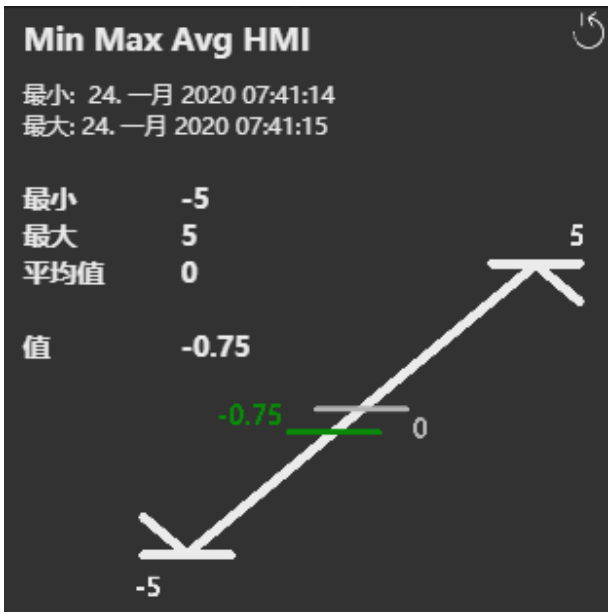
モード

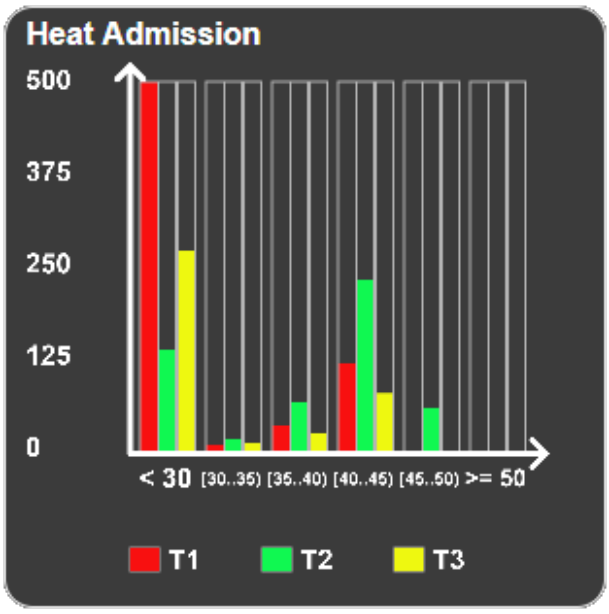




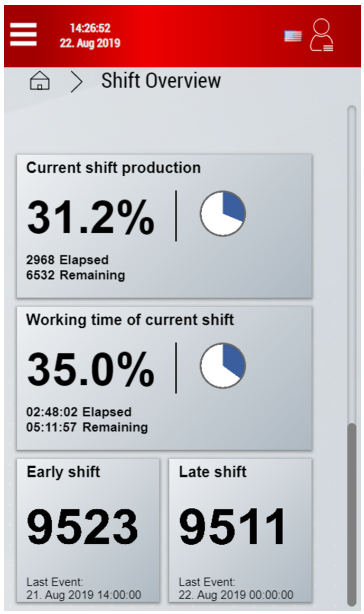


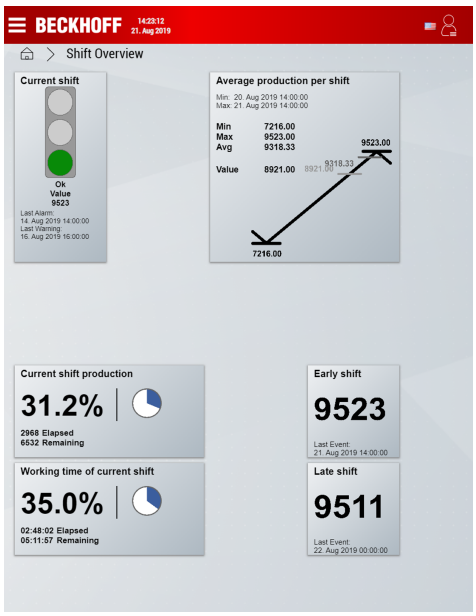
スタイル


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
平均値 0
値 -0.75
5
-0.75
0
-5

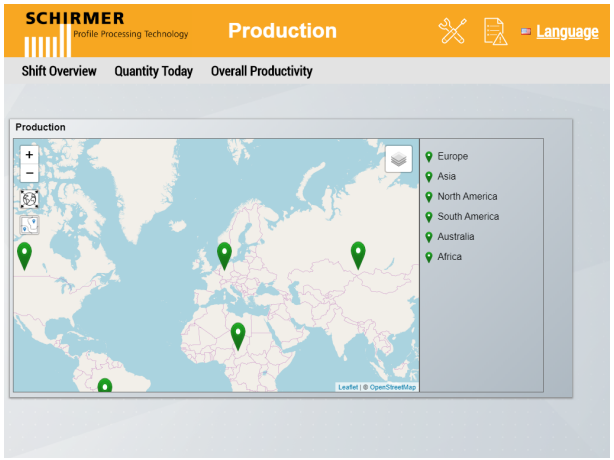

Heat Admission
500
375
250
125
0
< 30 [30..35) [35..40) [40..45) [45..50) >= 50
T1
T2
T3

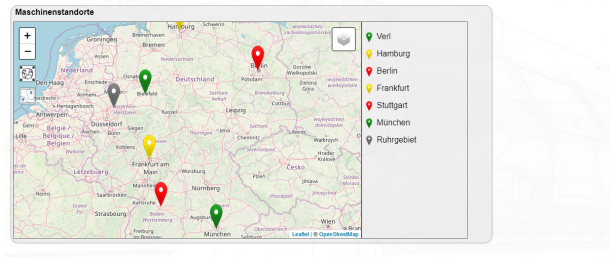
モバイルデバイス表示


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

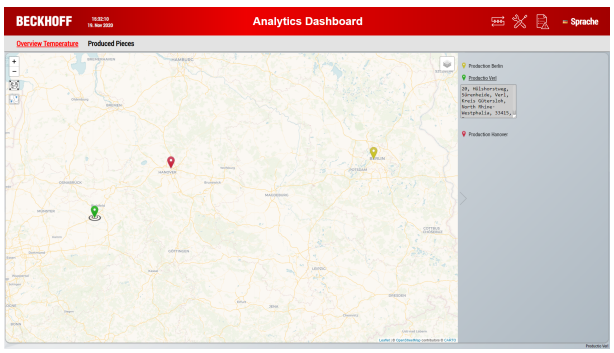


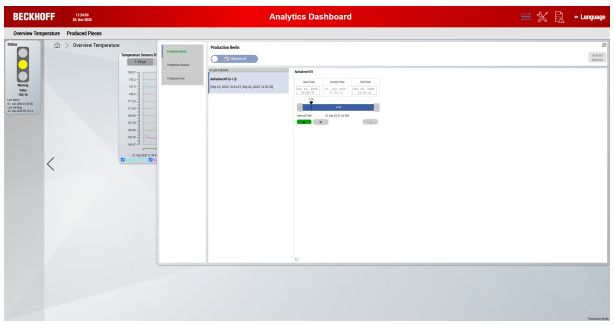
顧客固有ダッシュボード上での地図表示





履歴データと機械の切り替え





6.9.5 HMIエンジニアリングで生成したダッシュボードを修正する

TwinCAT 3 HMI プロジェクト

ダッシュボード生成の結果、完全なTwinCAT 3 HMIプロジェクトが作成されます。そのため、TwinCAT 3 HMI Engineeringが提供する全てのオプションを使用し、含めることができます。

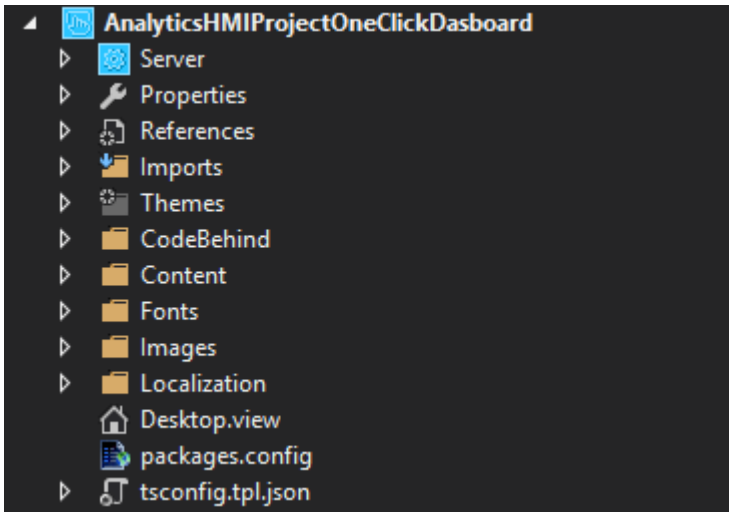


図 5:

ダッシュボードのカスタマイズの多くは、ランタイムでウェブブラウザ上で設定できます。

6.9.6 ランタイムでウェブブラウザ上でダッシュボードを設定

ランタイムでは、ウェブブラウザ上で様々な設定を行うことができます：

- ユーザごとのレイアウトを作成
- ユーザの管理（ユーザ管理とアクセス権の設定の項を参照）
- コンテンツへのアクセス権をカスタマイズ（ユーザ管理とアクセス権の設定の項を参照）
- ブラウザでコントロールのプロパティを変更

- パラメータのカスタマイズ
- グローバルダッシュボードのオプションを変更

Dashboardバージョン1.0では、一部の設定オプションが利用不可であることに注意してください。

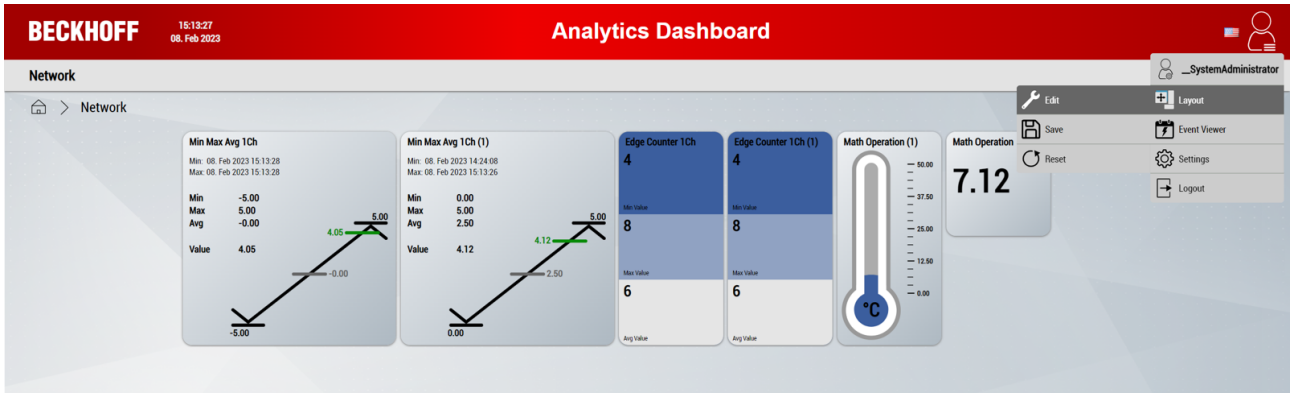
6.9.6.1 インタラクティブ機能でレイアウトをカスタマイズ

インタラクティブ機能では、ユーザ独自の方法でコンテンツ内にコントロールを配置できます。カスタマイズしたレイアウトは、HMIサーバの拡張機能「TcHmiAnalytics」に一元的に保存されます。

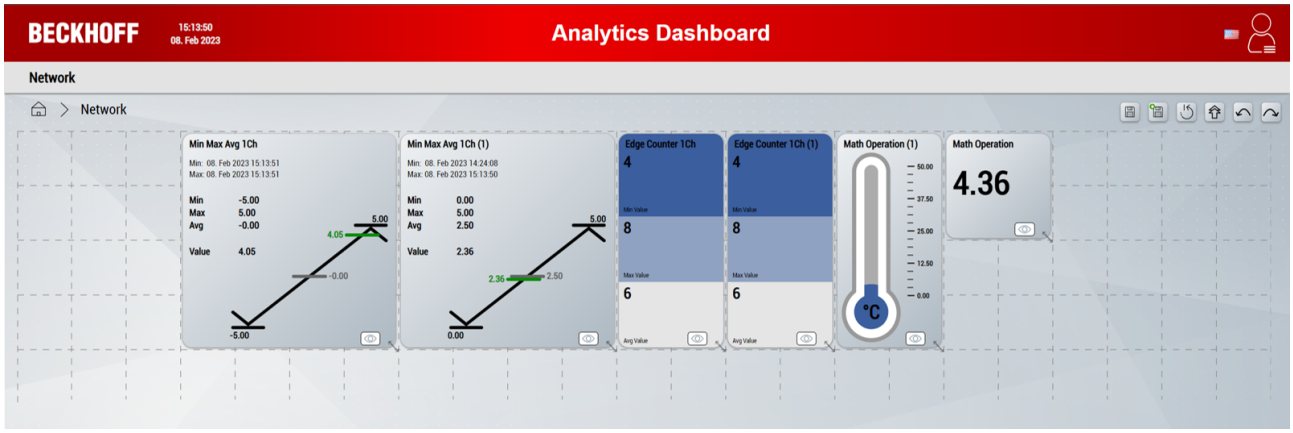
レイアウトの編集はDashboardバージョン2.0以降でのみ可能です。また、レイアウト編集は、ユーザが必要なアクセス権を持っている場合にのみ可能です。

編集モード

レイアウトの編集は特別な編集モードで行います。編集モードはメニューコントロールから呼び出すことができます。また、CTRL+Eのショートカットキーで編集モードを開始することもできます。



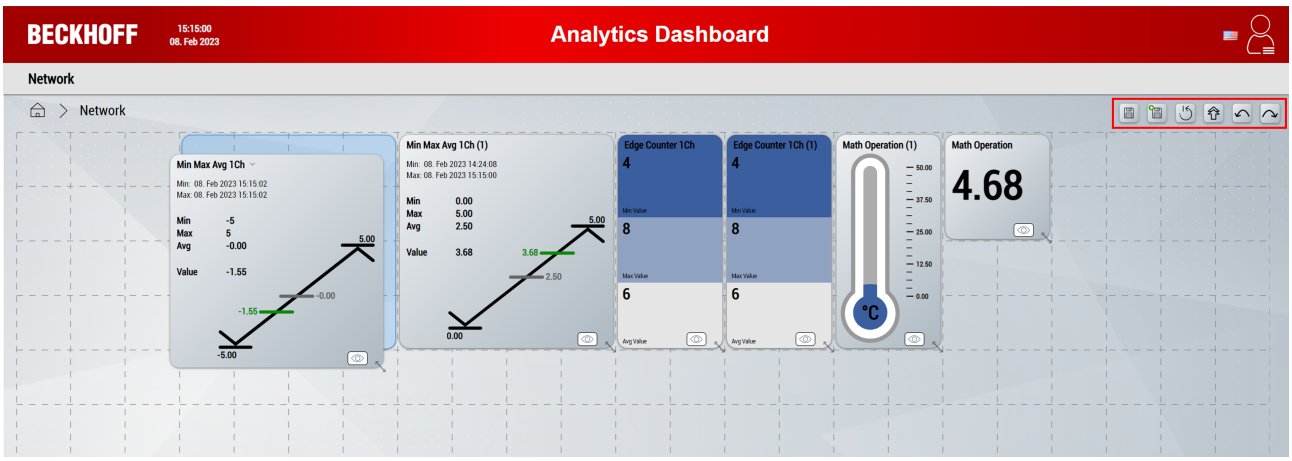
編集モードでは、背景にグリッドが表示され、その上にコントロールを自由に配置できます。



レイアウトの編集

提供可能なオプションは以下の通りです。

- **Positioning:**位置決めには、コントロールをクリックする必要があります。そうすることで固定が解除され、移動が可能になります。水色の枠はレイアウト内の新しい位置を示します。コントロールを離すと、新しい位置に固定されます。
- **Scale:**コントロールの右下にある矢印でサイズを変更できます。
- **Hide controls:**表示ボタン（コントロール右下）を使用して、表示/非表示を切り替えできます。非表示のコントロールは、編集モードでは半透明で表示されます。



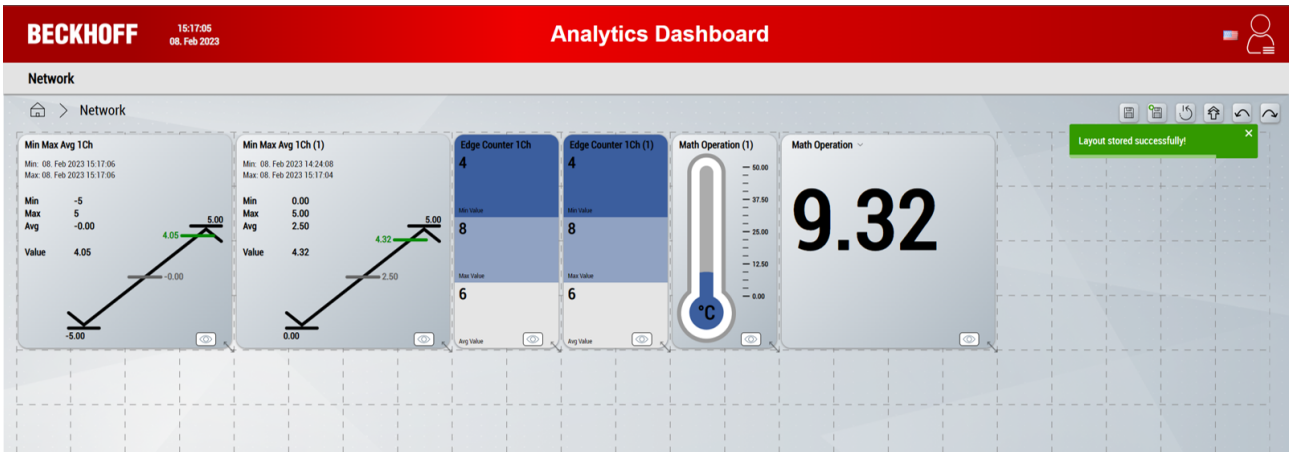
メニューの編集

編集モードでは、メニュー（赤枠）を追加して様々なアクションを実行できます：

- レイアウトを保存（ショートカットキー：CTRL + S）
- レイアウトを保存して編集モードを終了（ショートカットキー：CTRL + Q）
- 変更をリセット（ショートカットキー：CTRL + R）
- 空いている行が削除されるように、コントロールを上に移動（ショートカットキー：CTRL + U）
- 最後のアクションを元に戻す（ショートカットキー：CTRL + Z）
- 最後の操作を繰り返す（ショートカットキー：CTRL + Y）

レイアウトの保存

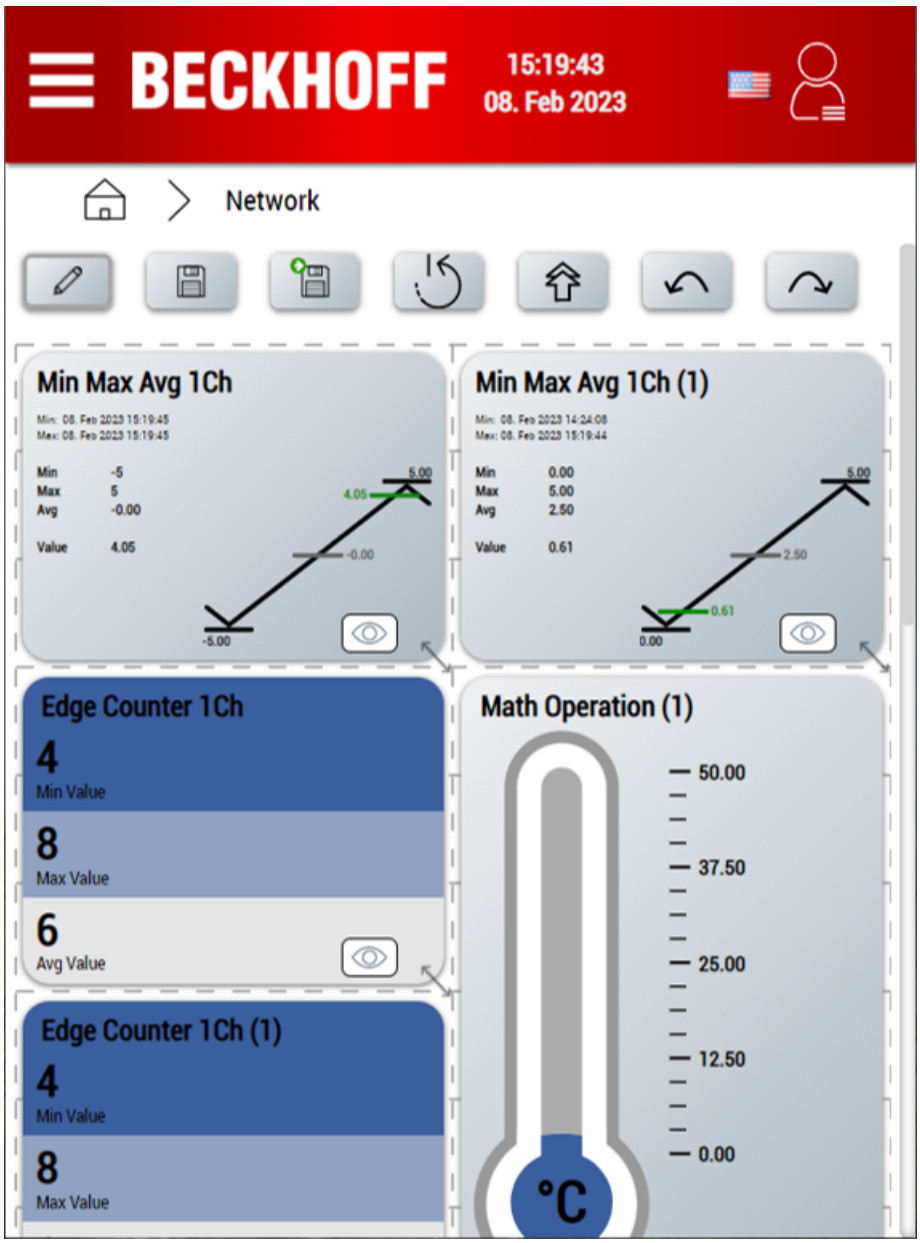
カスタマイズしたレイアウトは、編集メニューまたはメニューコントロールで保存できます。保存操作の結果がポップアップで表示されます。



保存中にエラーが発生した場合は、server configurationで「TcHmiAnalytics」拡張機能が有効になっているかどうかを確認してください。

モバイルデバイス用のレイアウト

ユーザ固有のモバイルデバイス用のレイアウトを追加で作成できます。これは標準レイアウトから独立しており、2列のみです。編集・保存操作は同じです。



6.9.6.2 ブラウザでコントロールのプロパティを変更

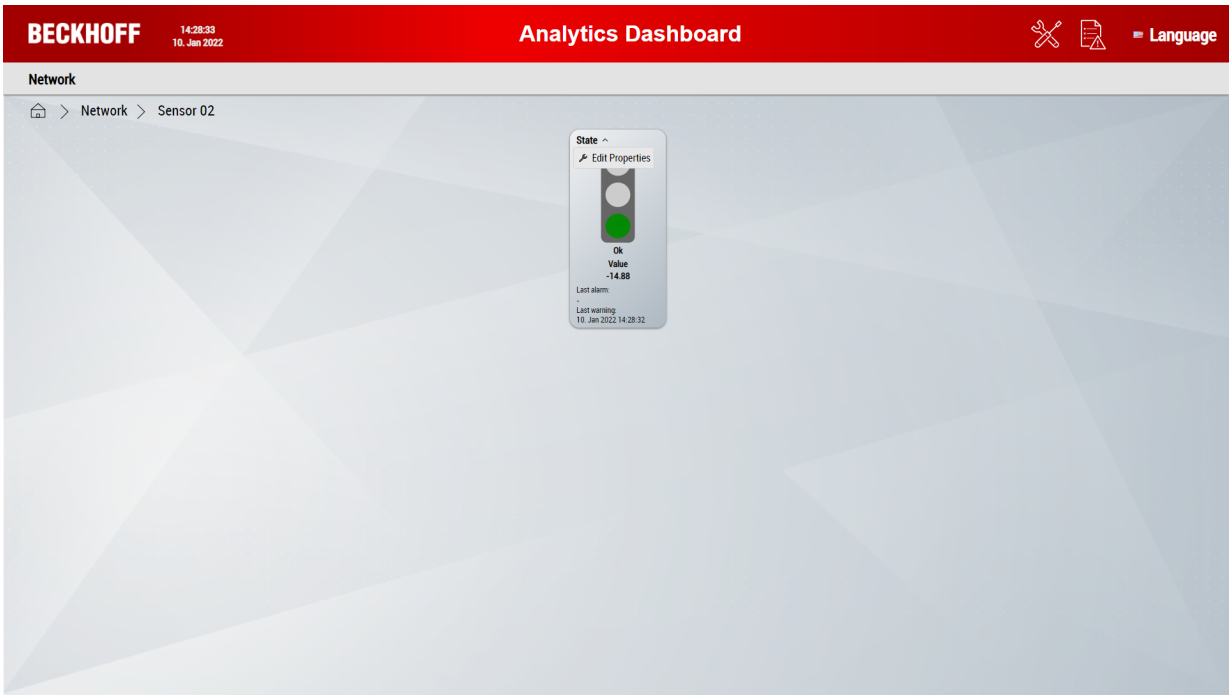
Analytics Workbenchプロジェクトでの変更を推奨

通常、ここで実施する変更は、ダッシュボードを生成する前にAnalytics Workbenchで既に変更できます（MappingウィザードまたはDashboardノードも参照してください）。これには、ダッシュボードが再生成されたときに変更が保存される利点があります。

各Analyticsコントロールには、ブラウザ上で動的に変更可能な特定のプロパティが選択されています。このオプションは、Analytics バージョン3.4.3145.0以降、またはControl Package バージョン1.1.31以降で使用できます。これを使用すると、古いHMIプロジェクトのNuGetパッケージを簡単に更新できます。

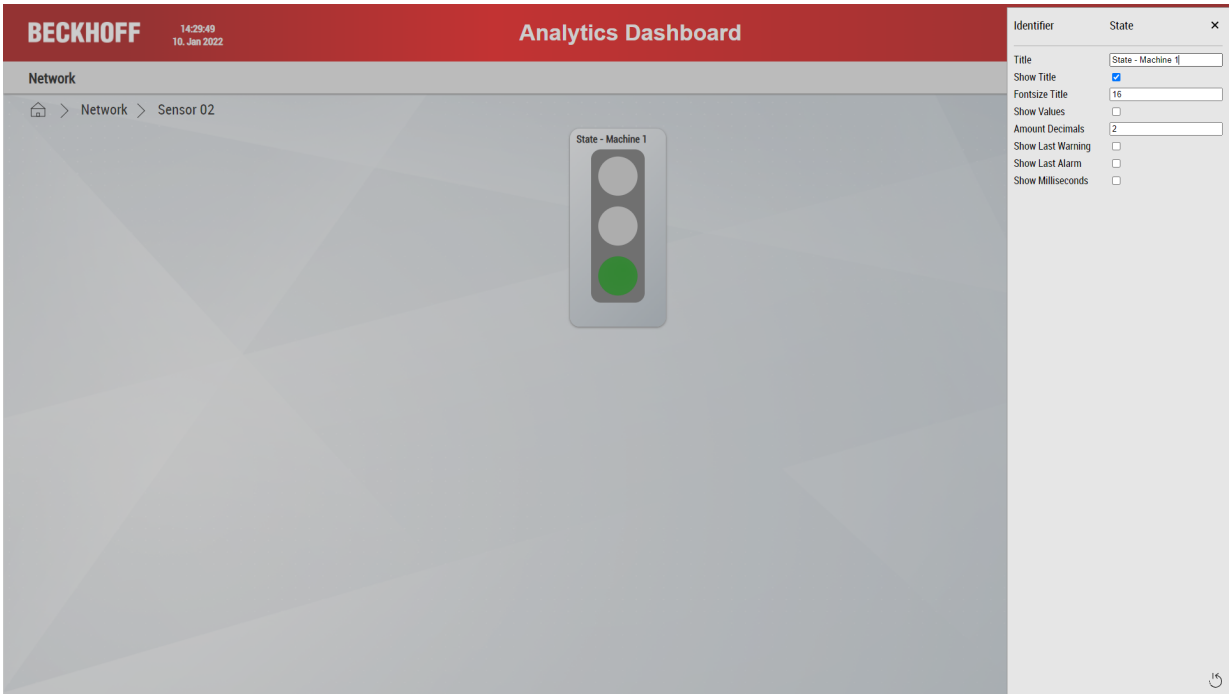
1. 生成された HMI プロジェクトを開き、プロパティを変更したいコントロールのタイトルをクリックします。

2. メニューが開くので、**Edit Properties**をクリックします。



⇨ 各コントロールは、カスタマイズ可能な異なるプロパティを持っています。ここでは、デモンストレーションのために、*Traffic Light*コントロールのプロパティを変更します。

3. タイトルを変更し、下の3つのボックスのチェックを外して、タイトルを変更した信号機のみを表示します。

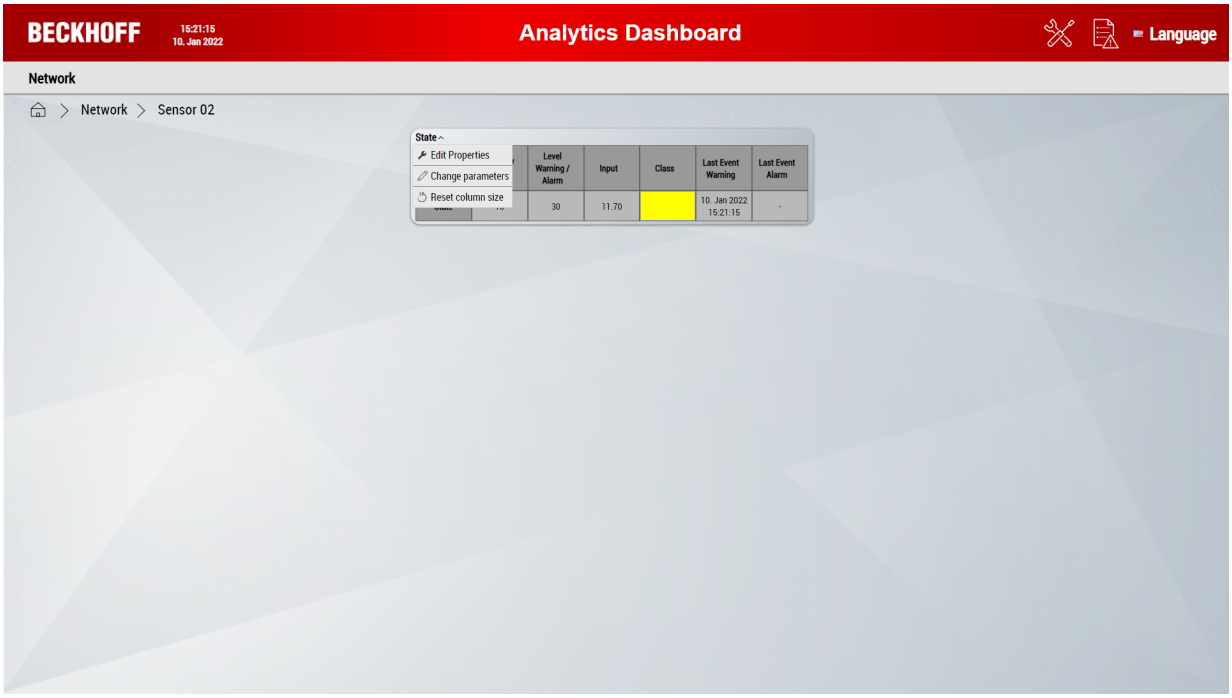


⇨ 変更内容はクライアントサイド（ダッシュボードを開いたデバイス上）に保存され、そこでしか表示されません。将来的には、HMIサーバ経由で保存され、変更はグローバルの各デバイス上で反映されることになります。

6.9.6.3 ブラウザでパラメータを変更する

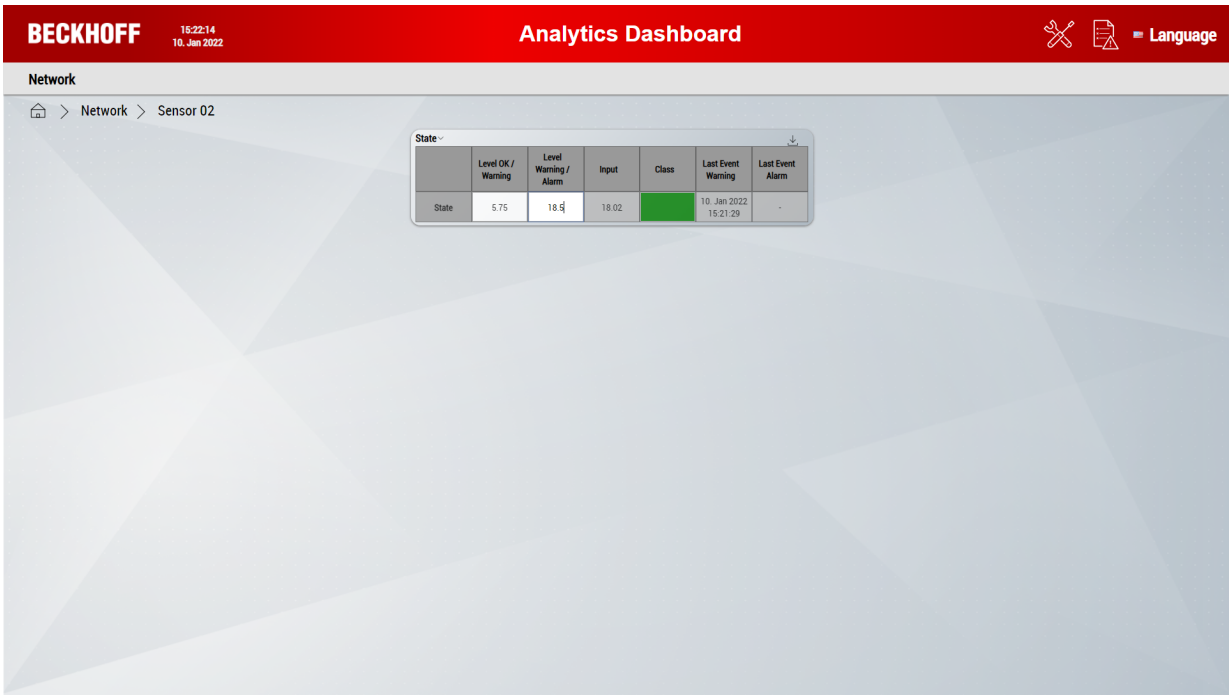
限界値などの数値パラメータは、ダッシュボードで変更可能です。これにはData Tableコントロールを使用します。値は永続的にPLCに書き戻されるため、再起動後も使用可能です。

1. 生成された HMI プロジェクトを開き、パラメータを変更したいTableコントロールのタイトルをクリックします。
2. メニューが開いたら、**Change parameters**をクリックします。パラメータを持たないモジュールには、このメニュー項目はありません。

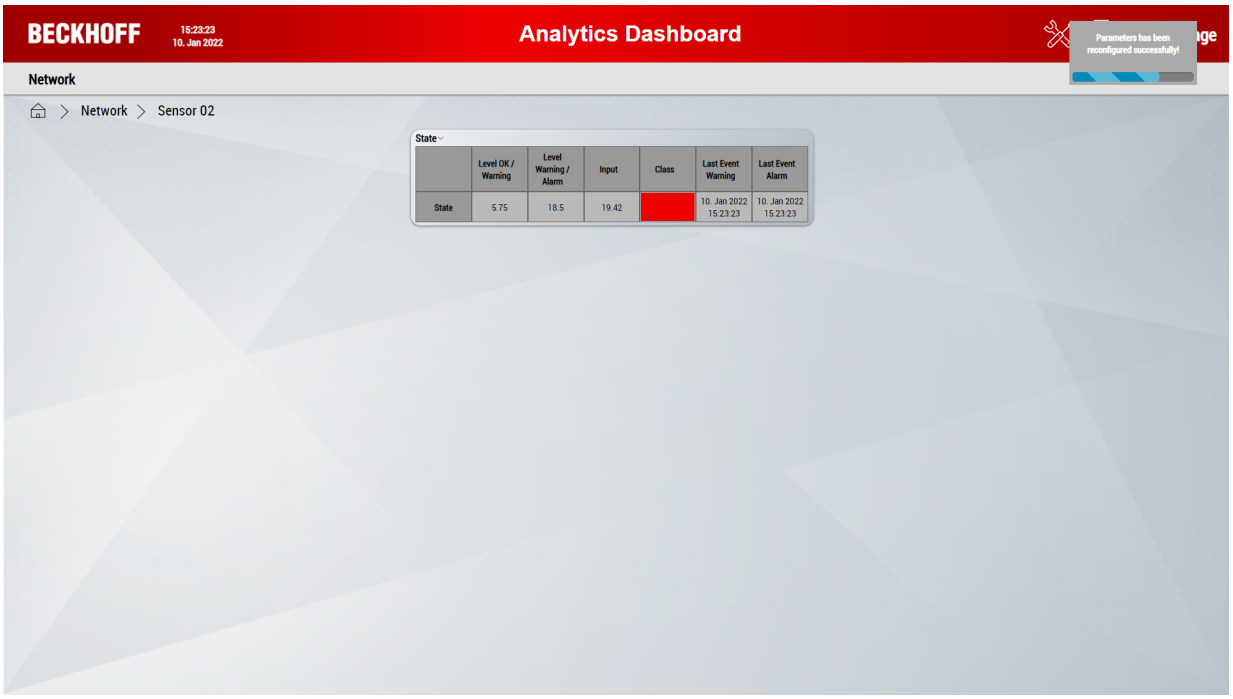


⇨ 変更可能なフィールドは白い背景の入力フィールドに変わります。

3. 値を変更し、右上の保存アイコンをクリックします。

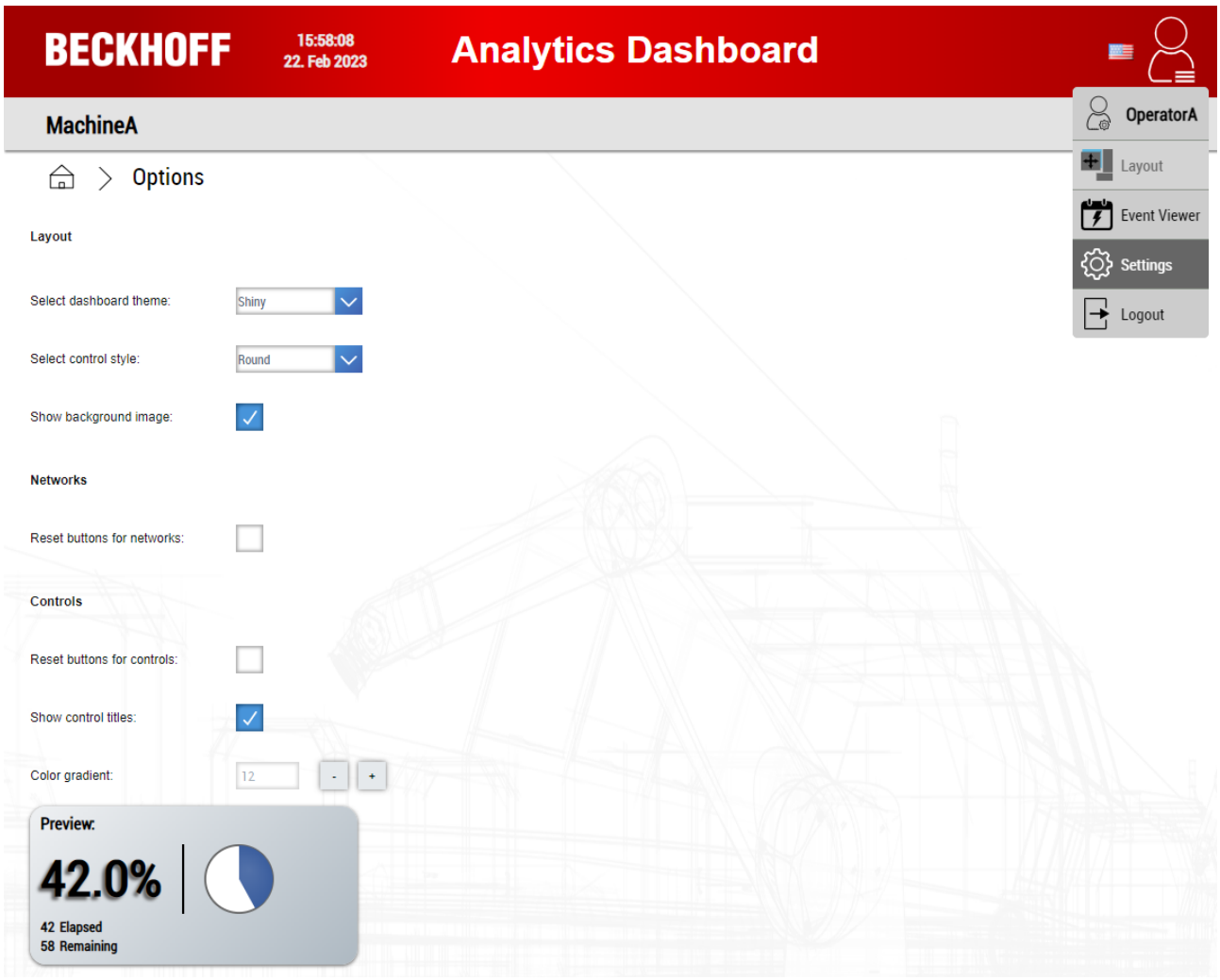


⇨ パラメータを保存すると、ダッシュボードの右上にメッセージが表示されます。再設定が完了したかどうか通知されます。



6.9.6.4 グローバルダッシュボードのオプションをカスタマイズ

オプションページでは、ダッシュボードの一般設定を編集できます。



レイアウト

Select dashboard theme	ダッシュボードのテーマをシャイニー、ダーク、ライトから選択して変更します。
Select control style	コントロールのスタイルをフラットとラウンドから選択して変更します。
Show background image	標準または顧客固有の背景画像を表示します。

ネットワーク

Reset buttons for networks	ネットワーク全体のリセットボタンを有効にします。選択したネットワーク内の全てのコントロールをリセットします。

コントロール

Reset buttons for controls	コントロールのリセットボタンを有効にします。Reset a single control.
Show control titles	全てのコントロールのタイトルを有効にします。

6.9.7 ウェブブラウザでチャートを作成

新しいインタラクティブチャート機能により、ニーズに合わせてダッシュボードにチャートを追加できます。チャートはウェブブラウザで追加・設定できます。エンジニアリング環境は必要ありません。Analytics Storage Providerで履歴化されたデータは、チャートのデータソースとして使用できます。設定時に、表示する変数、時間範囲、チャートタイプを選択できます。

Analytics Storage Providerの更新が必要

データを正しくロードするには、**TwinCAT Analytics Storage Provider バージョン1.0.12.3以降**が必要です。

インタラクティブチャート機能を使用するには、以下のNuGetパッケージが必要です：

Beckhoff.TwinCAT.Analytics.Controls 1.35.0以降

Beckhoff.TwinCAT.Analytics.ControlsExtension 1.35.0以降

インタラクティブチャート機能は、NuGetパッケージを更新することにより、既に生成されたHMIプロジェクトに統合できます。

6.9.7.1 データソースの管理

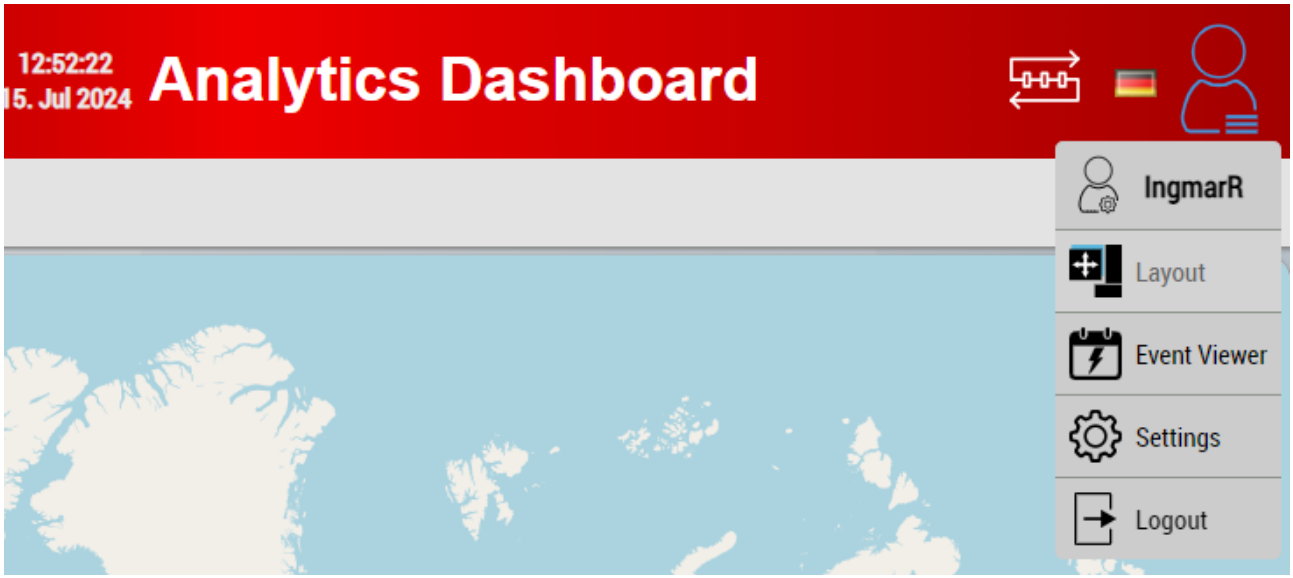
Analytics Storage Providerで履歴化されたデータは、チャートのデータソースとして使用できます。データソースはダッシュボードで直接管理できます。

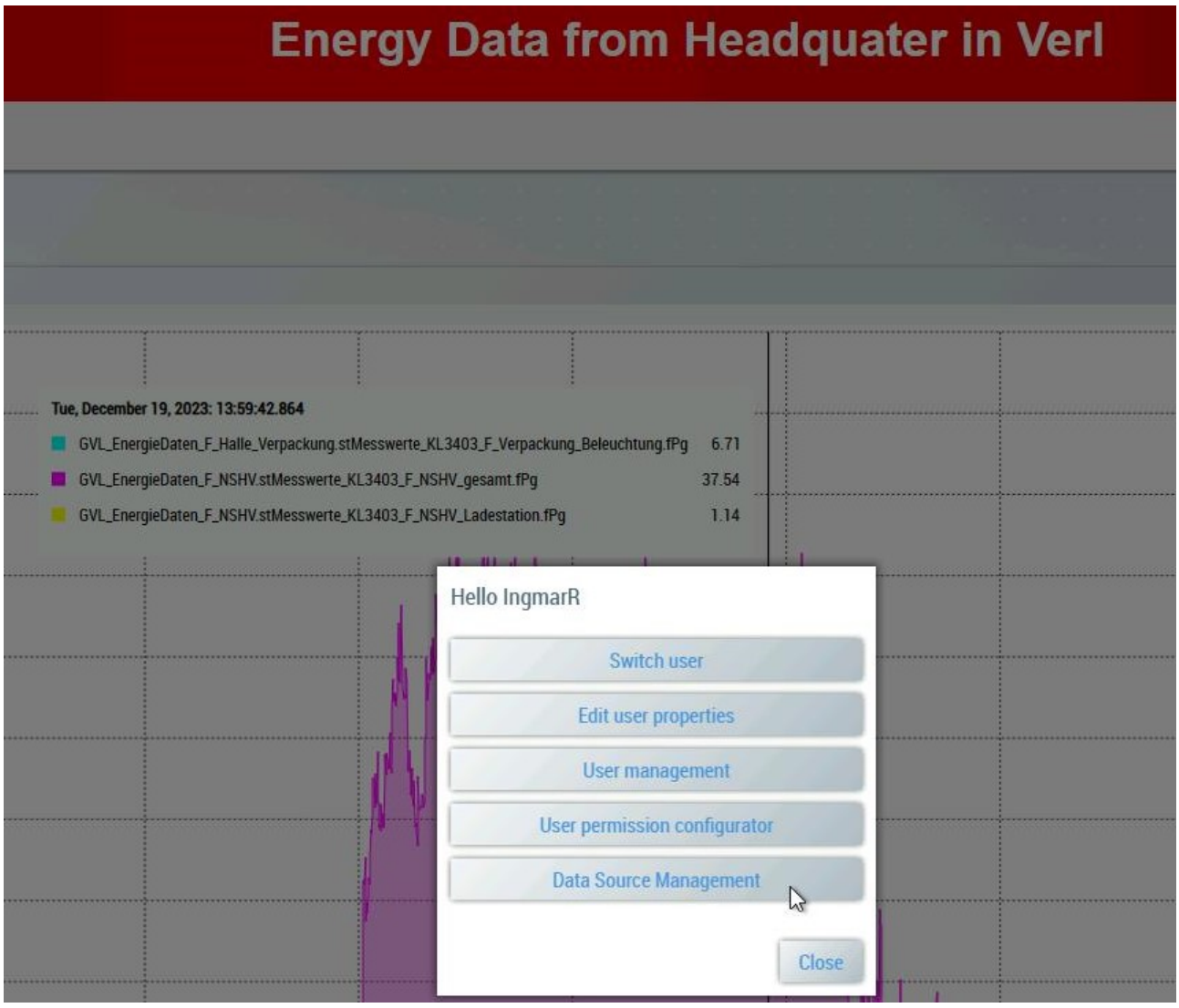
コード生成時に履歴データソースを自動追加

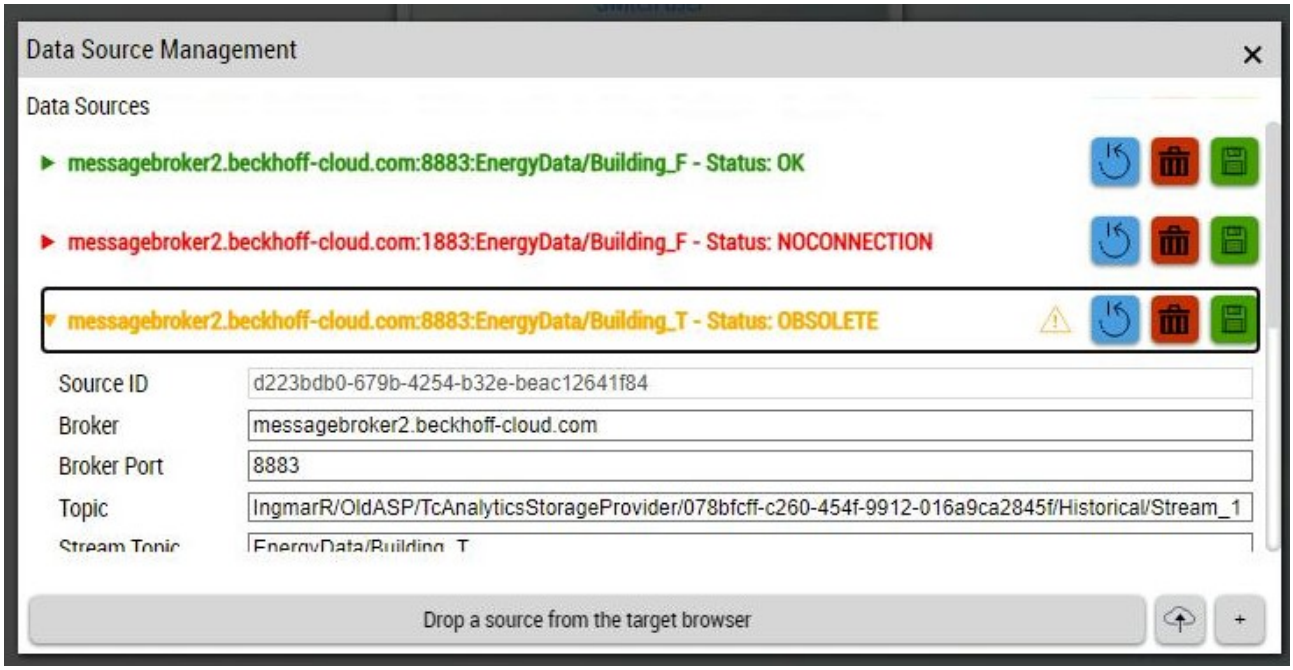
TwinCAT Analytics Workbenchバージョン49.0以降では、コード生成時にプロジェクトの全ての履歴データソースがダッシュボードに自動的に転送されます。これらがプロジェクトで積極的に使用されているか、単にソースとして追加されているかに関わらず転送されます。ライブデータソースは使用できませんのでご注意ください。

データソース管理のオープン化

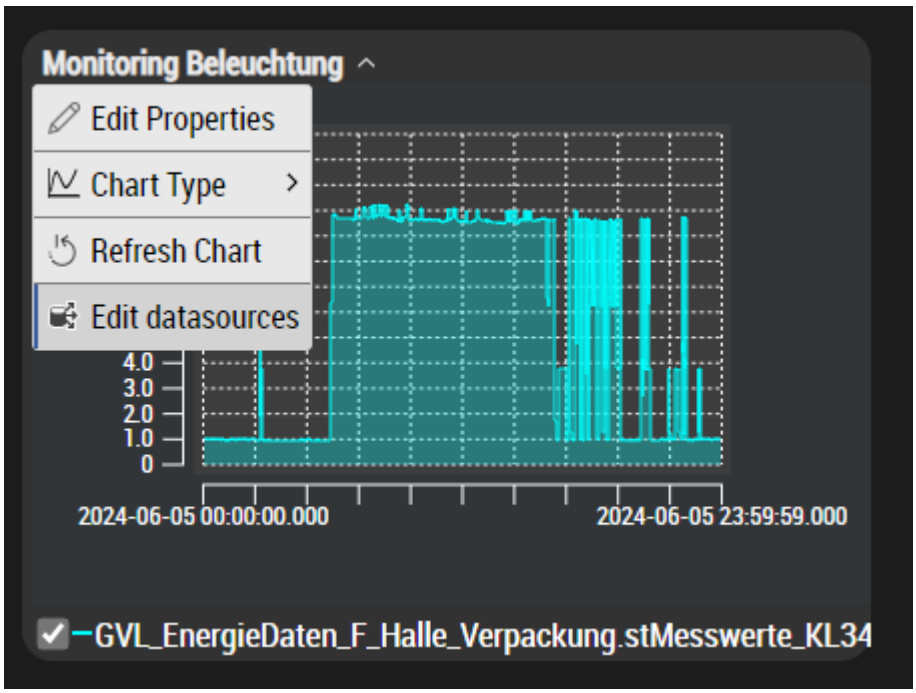
データソースは、「__SystemAdministrators」 または 「Administrators」 グループのユーザのみが管理できます。右上のメニューアイコンからデータソース管理を開くことができます：**Menu - <username> - Data Source Management**。







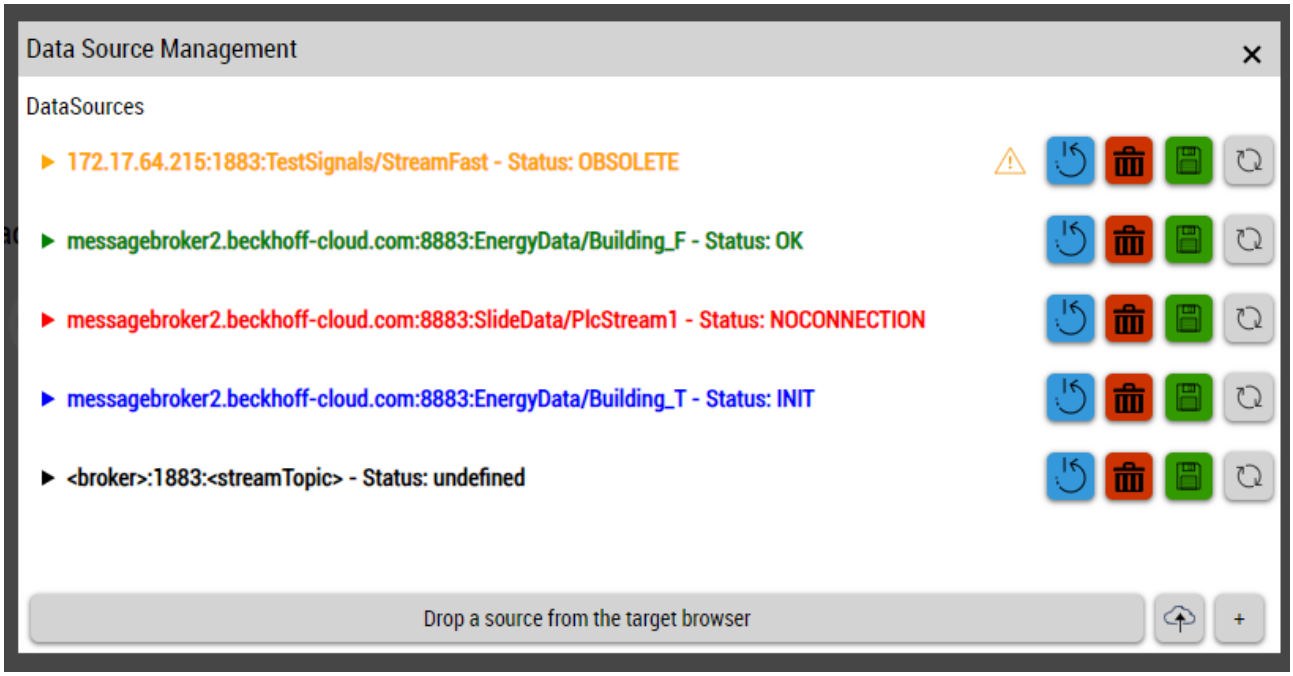
また、既存の**Interactive Chart**からデータソース管理を開くこともできます。



データソースの概要

既存の全てのデータソースがデータソース管理に表示されます。各データソースのステータスも表示されます。これには次のような意味があります：

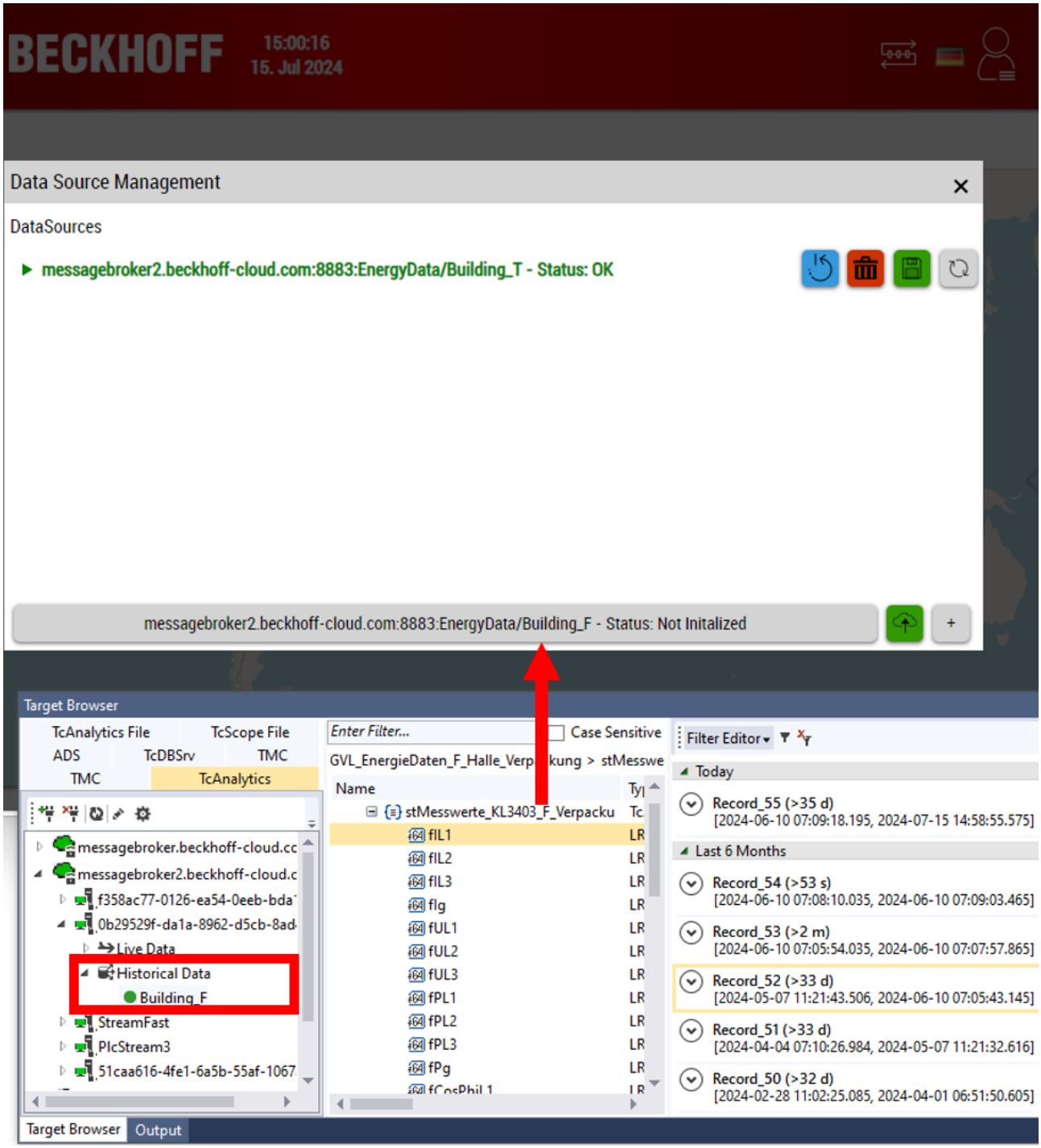
- OK：データソースが正しく設定されました。
- NOCONNECTION: データソースが正しく設定されていません。この場合、接続パラメータをチェックし、修正する必要があります。
- OBSOLETE: データのストレージプロバイダが古いため、少なくともバージョン 1.0.12.3以降 に更新する必要があります。更新しない場合、データを正しく照会できません。
- INIT：新しい設定がサーバエクステンションに送信されました。これはデータソースへの接続を試みます。
- UNDEFINED：設定はローカルにのみ作成され、サーバエクステンションにはまだ保存されていません。
- NOSOURCEHANDLER: サーバエクステンションで設定が正しくロードされませんでした。



データソースの作成

新しいデータソースは、TargetBrowserからドラッグ＆ドロップするだけで追加できます。**TcAnalytics** タブで、Historical Data sources を選択します。
全てのシンボルが自動的にデータソースに追加されます。

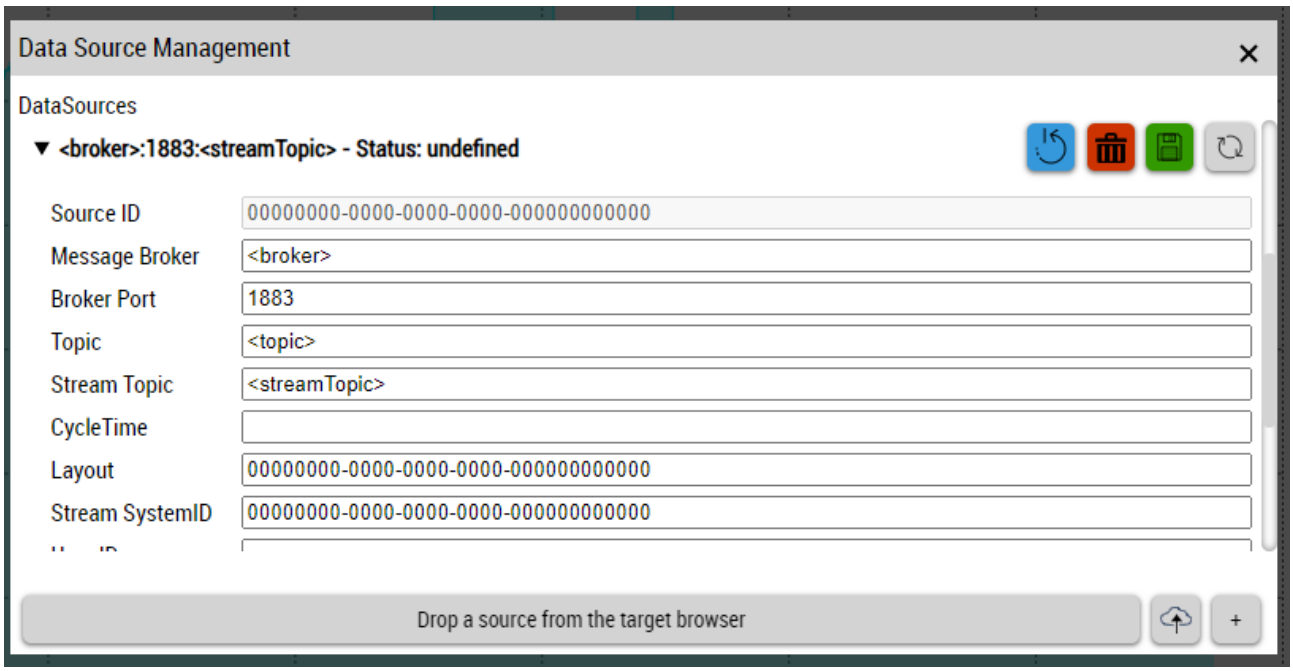
パラメータの読み込みに成功すると、アップロードアイコンが緑色に点灯します。アップロードアイコンをクリックして、サーバエクステンションに設定を送信します。



Target Browserがデバイス上で利用できない場合は、右下のプラスアイコンを使用して空のデータソースを作成できます。これは手動で設定する必要があります。

データソースの設定と変更

データソースの左側にある矢印を使用して、データソースを展開し、編集できます。



以下のパラメータを設定できます。

パラメータ	説明
Source ID	データソース固有のID。これは一意であり、サーバエクステンションによって割り当てられます。
Message Broker	メッセージブローカーのアドレス
Broker Port	メッセージブローカーのポート
Topic	履歴データのMQTTトピック ".../TcAnalyticsStorageProvider/.../Historical/Stream..."
StreamTopic	記録したストリームのオリジナルトピック。
Cycle Time	データソースのサイクルタイム（100ナノ秒単位）。
Layout	ストリームのレイアウト
Stream SystemID	記録したストリームのSystemID。
UserID	メッセージブローカーにアクセスするためのユーザ名。
Password	メッセージブローカーにアクセスするためのパスワード。
（証明書付き）	値がチェックされている場合、選択した証明書ファイルが使用されます。
CA	CAファイル（サーバエクステンションにアップロード可能）。
CERT	CERT ファイル（サーバエクステンションにアップロード可能）。
KEY	KEYファイル（サーバエクステンションにアップロード可能）。
Key Password	KEYファイルのパスワード。

変更内容は、保存アイコン（緑）を使ってサーバエクステンションに送信できます。また、ローカルで行った変更は、リセットアイコン（青）を使って元に戻すこともできます。データソースのステータスは、リフレッシュアイコン（グレー）を使って再度照会できます。

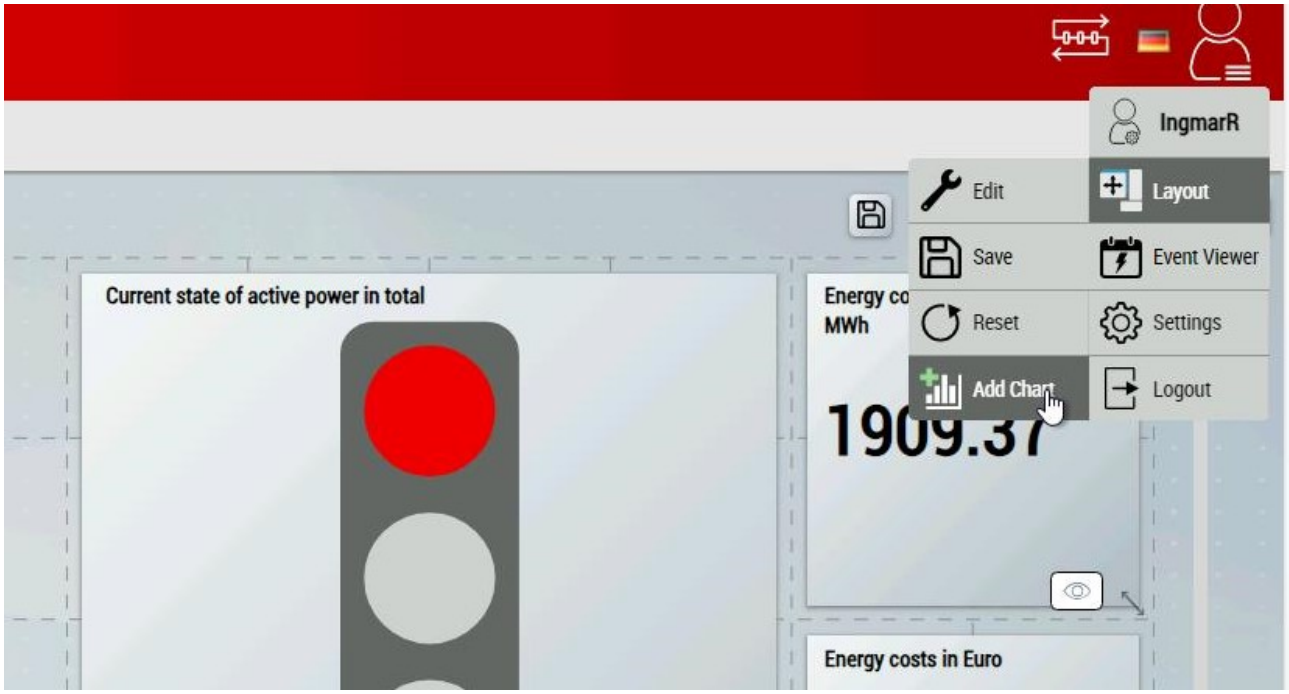
データソースの削除

データソースは、削除アイコン（赤）を使用して削除できます。このデータソースを使用した全てのチャートは機能しなくなりますのでご注意ください。ブローカーアドレスや証明書の更新など、パラメータが変更された場合は、既存のデータソースを適応させ、削除しないことをお勧めします。

6.9.7.2 チャートの追加/削除

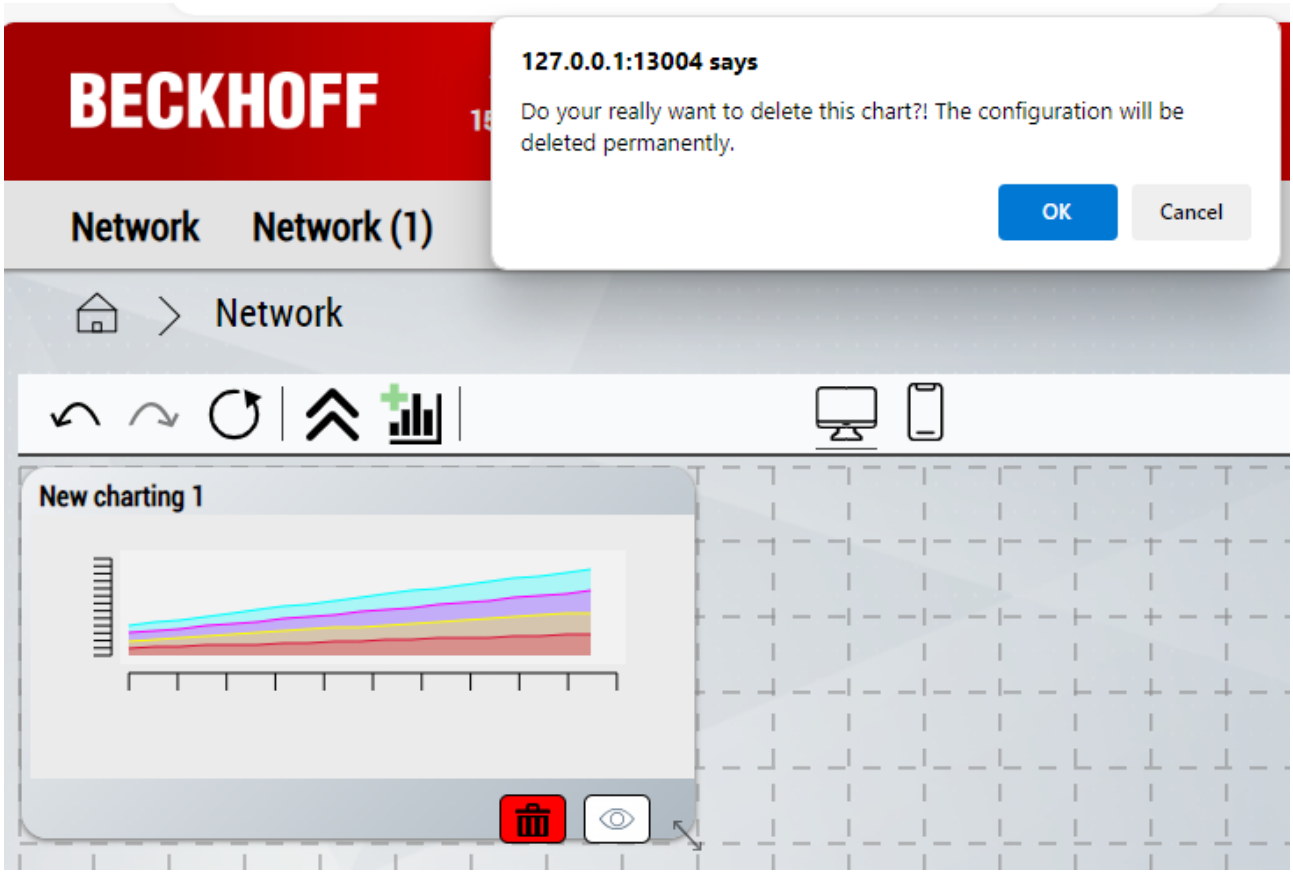
ウェブブラウザ

メニューアイコンから新しいチャートを追加します。Layoutカテゴリの下にある**Add Chart**をクリックします。新しいチャートはレイアウトの最初の空き位置に挿入されます。



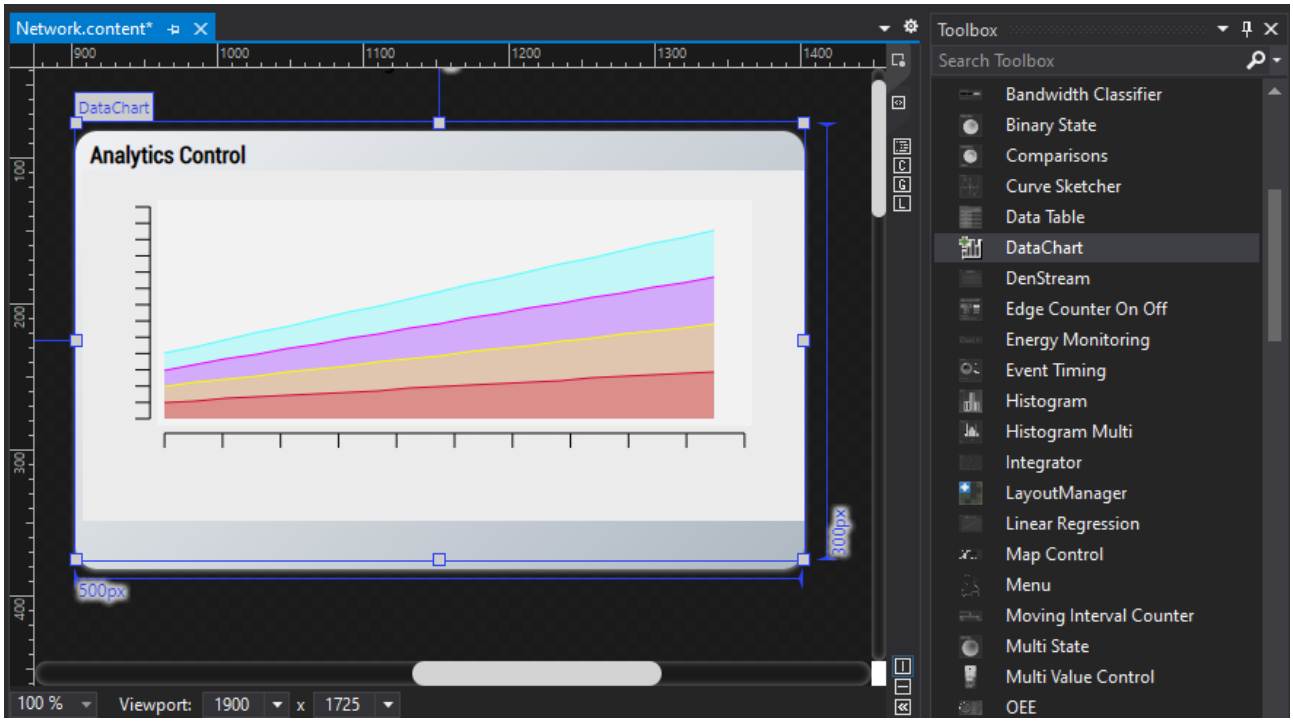
チャートは、ユーザがレイアウトを編集する権限を持っている場合にのみ追加できます。

追加したチャートはユーザ固有のものであり、現在のユーザのみが見ることができます。位置や大きさは、必要に応じてレイアウト内で変更できます。編集モードでは、右下のごみ箱アイコンを使ってチャートを削除できます。



HMIエンジニアリング

チャートは、HMIエンジニアリング経由で追加することもできます。これらのチャートはデフォルトで全てのユーザが利用でき、削除できません。エンジニアリングで追加したチャートのconfiguration [▶ 482]は、ウェブブラウザ上で設定します。

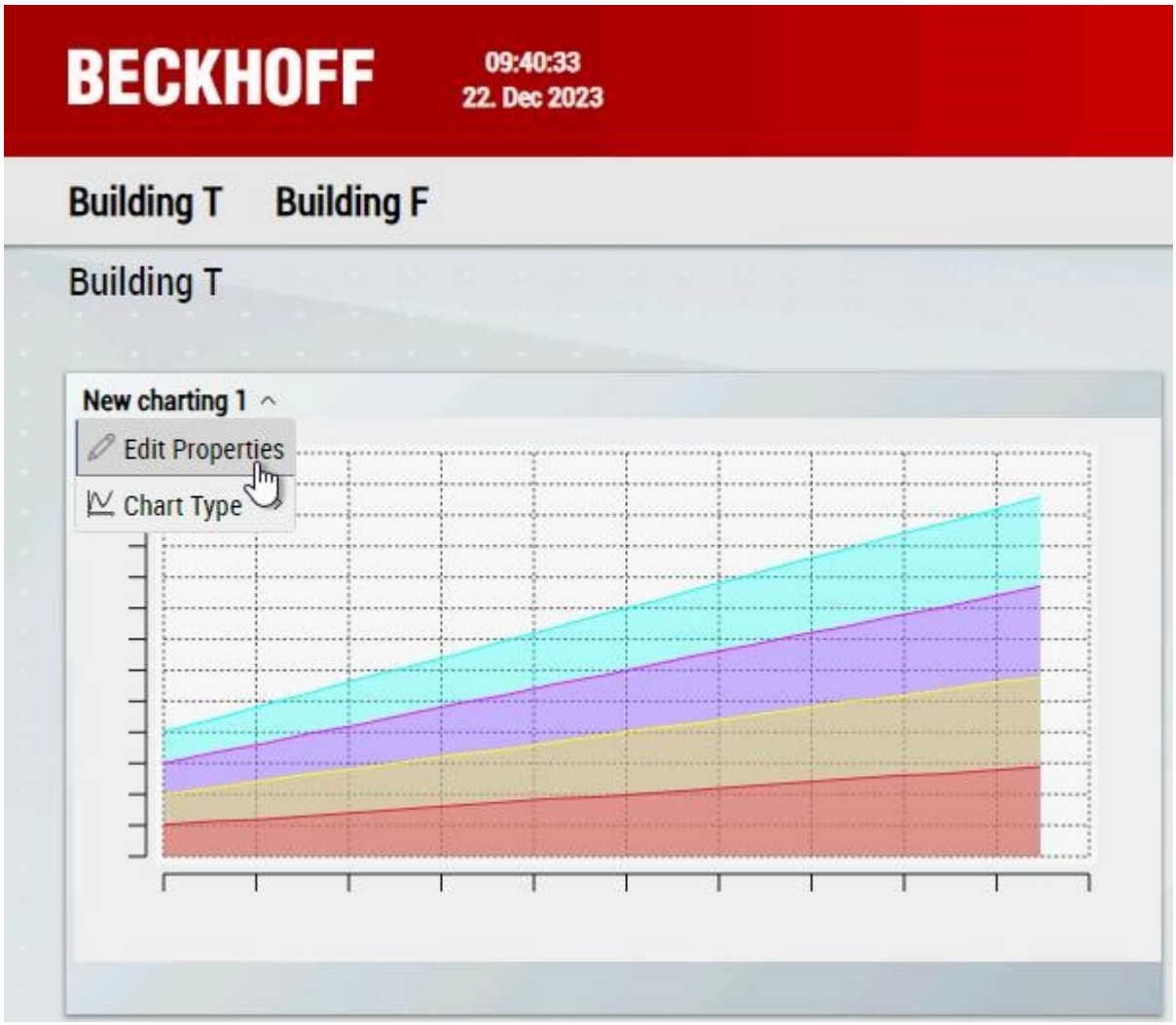


6.9.7.3 チャートの設定

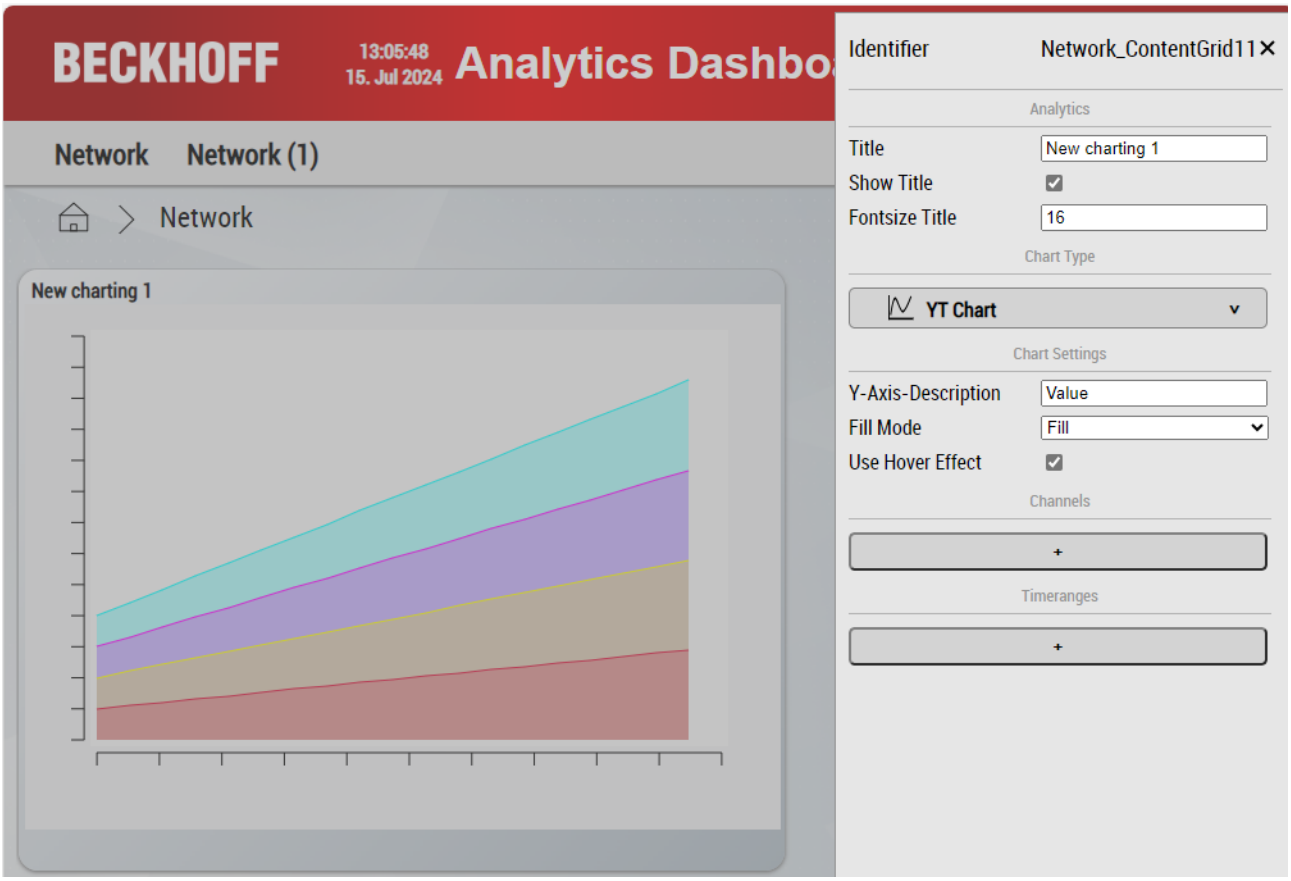
チャートはウェブブラウザで設定します。全ての設定（チャートの種類、変数、期間など）は、チャートのプロパティウィンドウで編集できます。

チャートのプロパティを開く

Chartコントロールのタイトルをクリックして、チャートのプロパティを開きます。



チャートのプロパティを開くには、**Edit Properties**をクリックします。

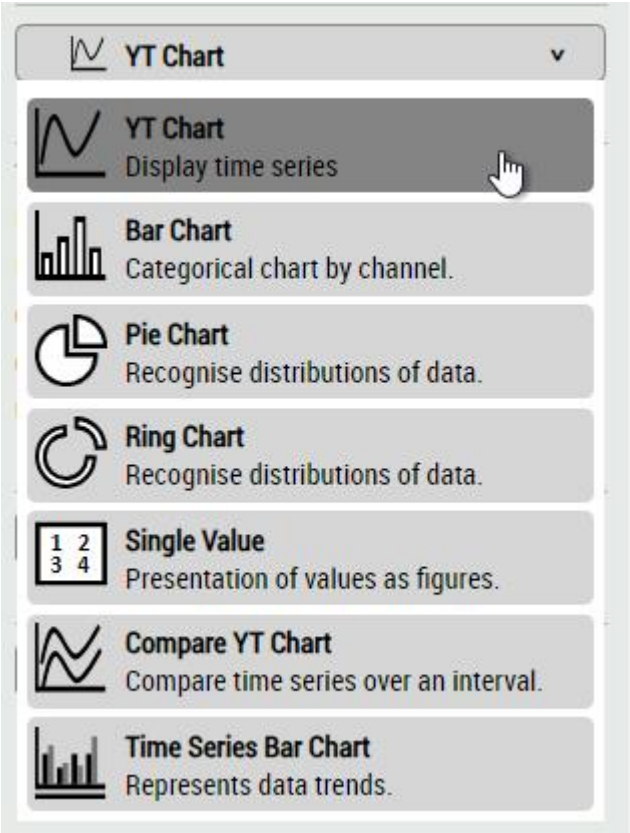


チャートのプロパティは4つのカテゴリに分類されます。

- Chart Type:チャートタイプの選択
- Chart Settings:チャートの設定
- Channels:変数の選択と表示
- Timeranges：表示する時間範囲の選択

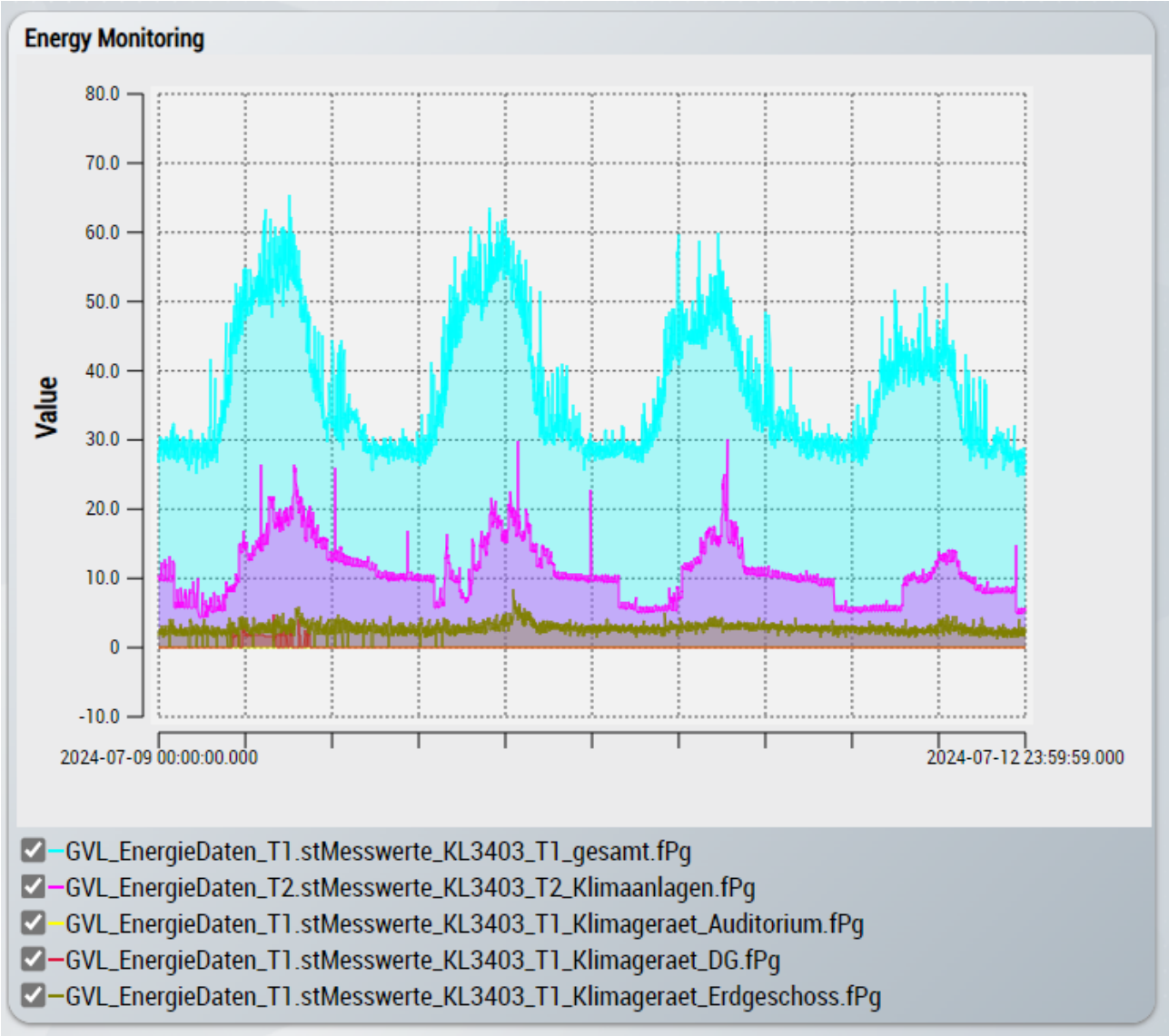
6.9.7.3.1 チャートタイプ

チャートタイプは、チャートのプロパティで設定できます。



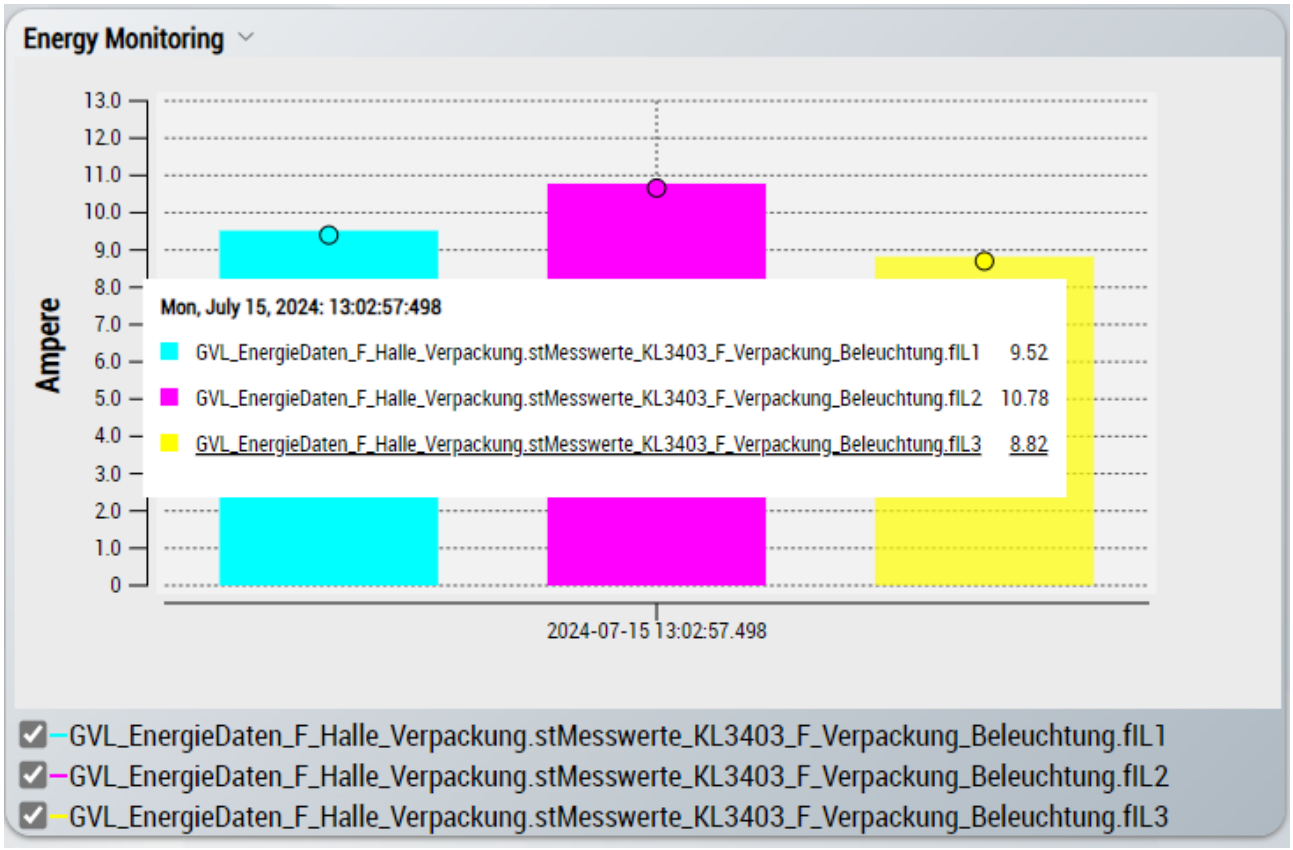
以下のチャートタイプを使用できます：

YTチャート



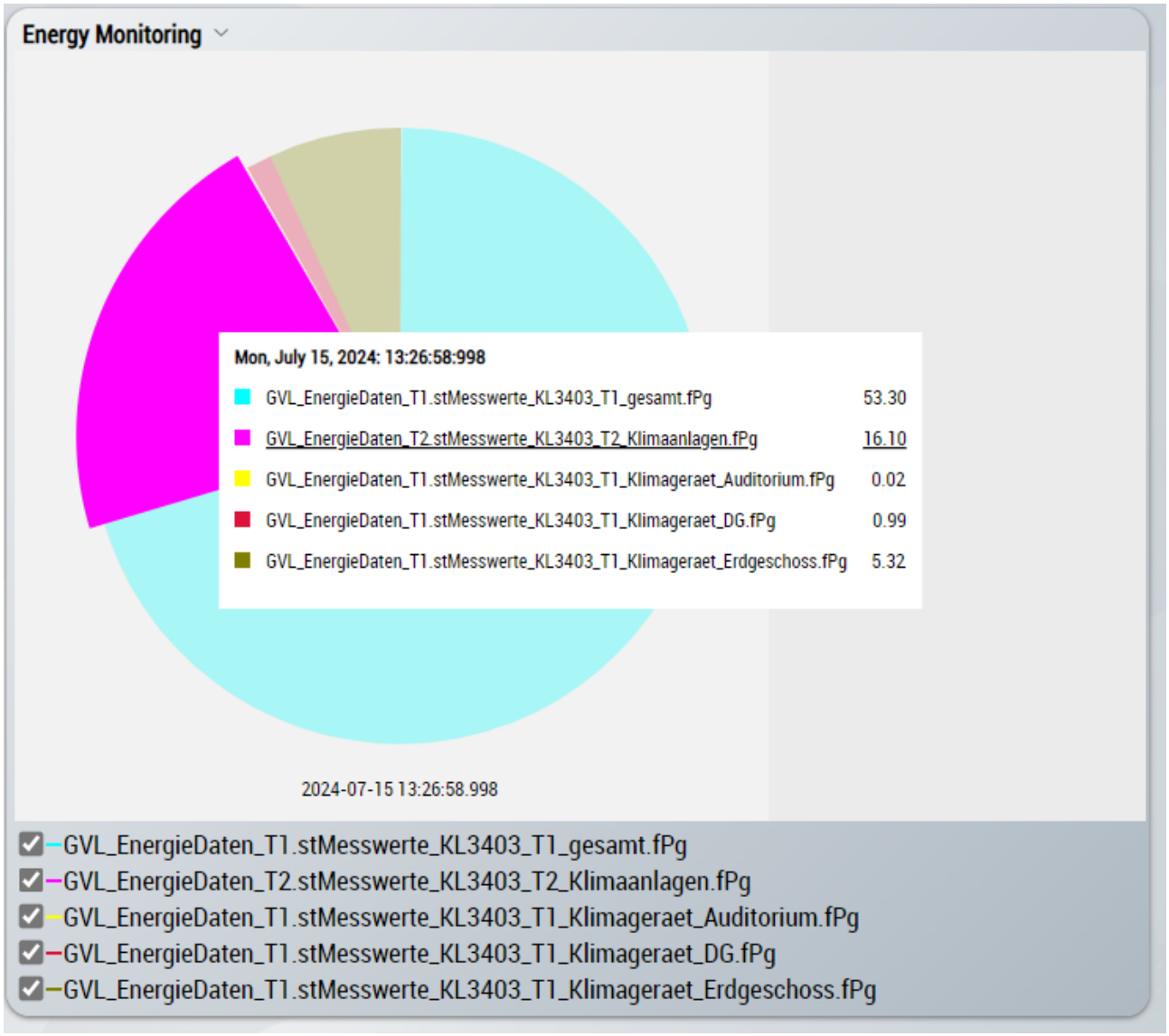
X軸に時間、Y軸に対応する値を表示します。

棒グラフ



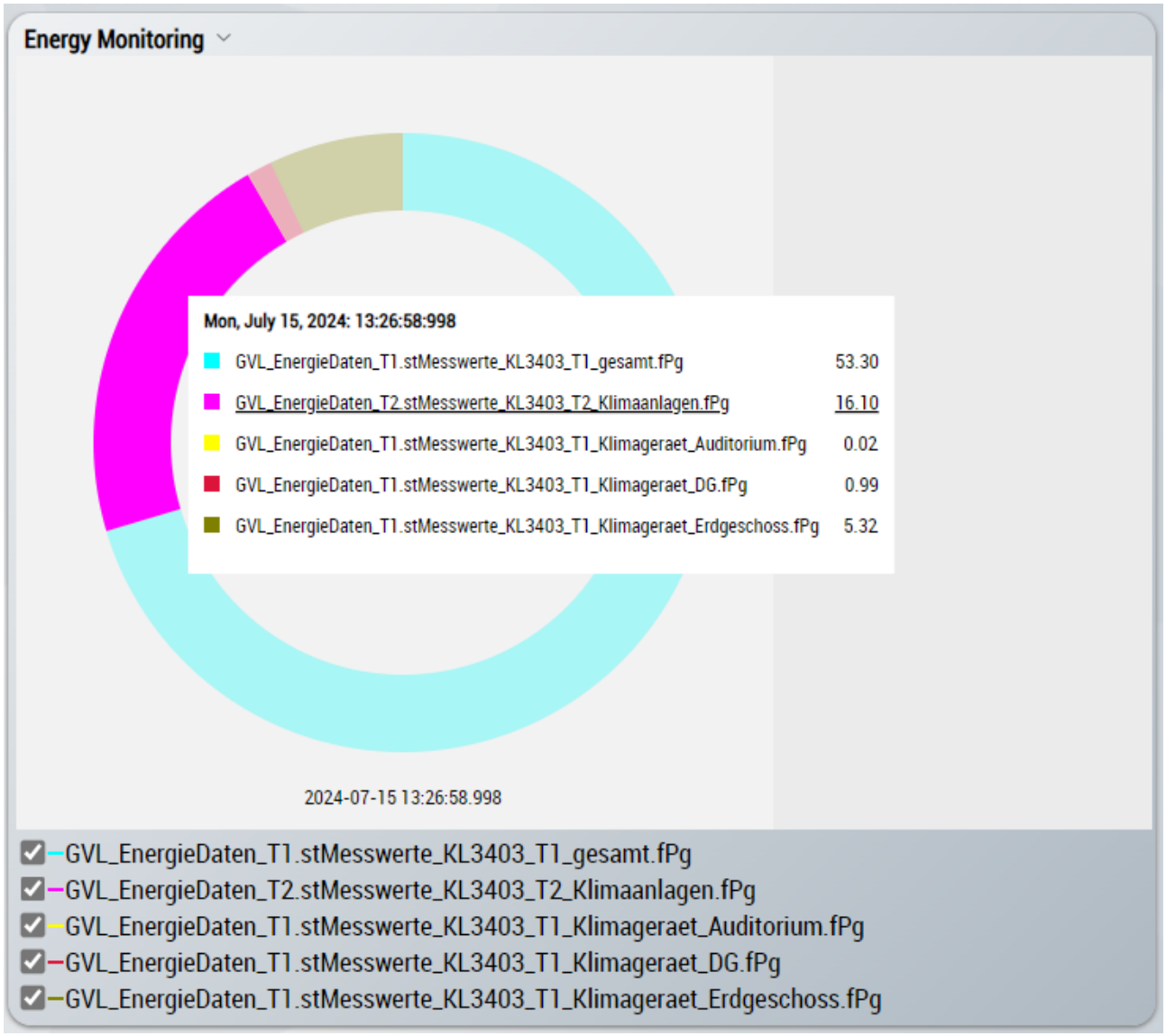
選択した時間範囲と変数ごとに棒グラフが表示されます。棒グラフは時間範囲の最後の値を示します。

円グラフ



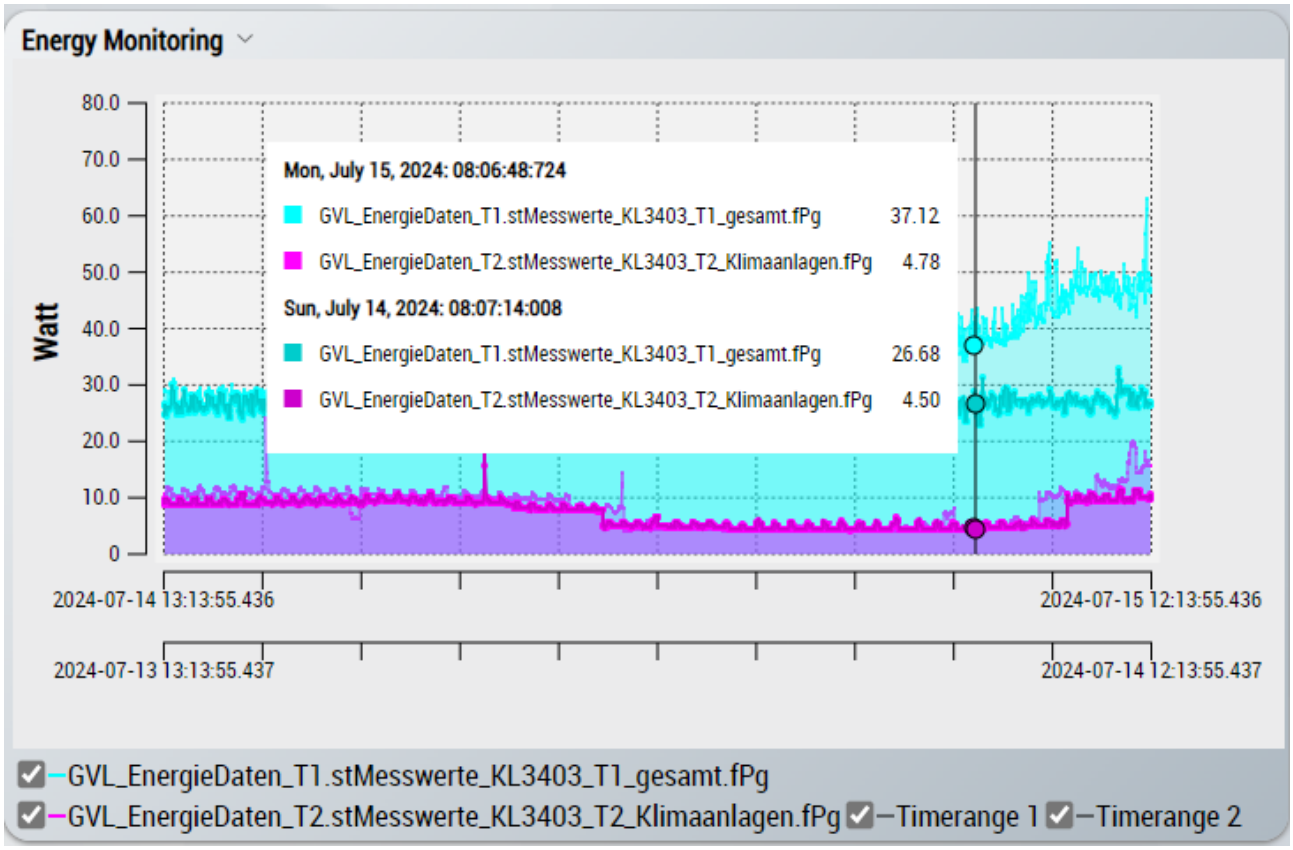
選択した時間範囲ごとに最後のサンプル値を示す円グラフが表示されます。負の変数値は表示されません。

リングチャート



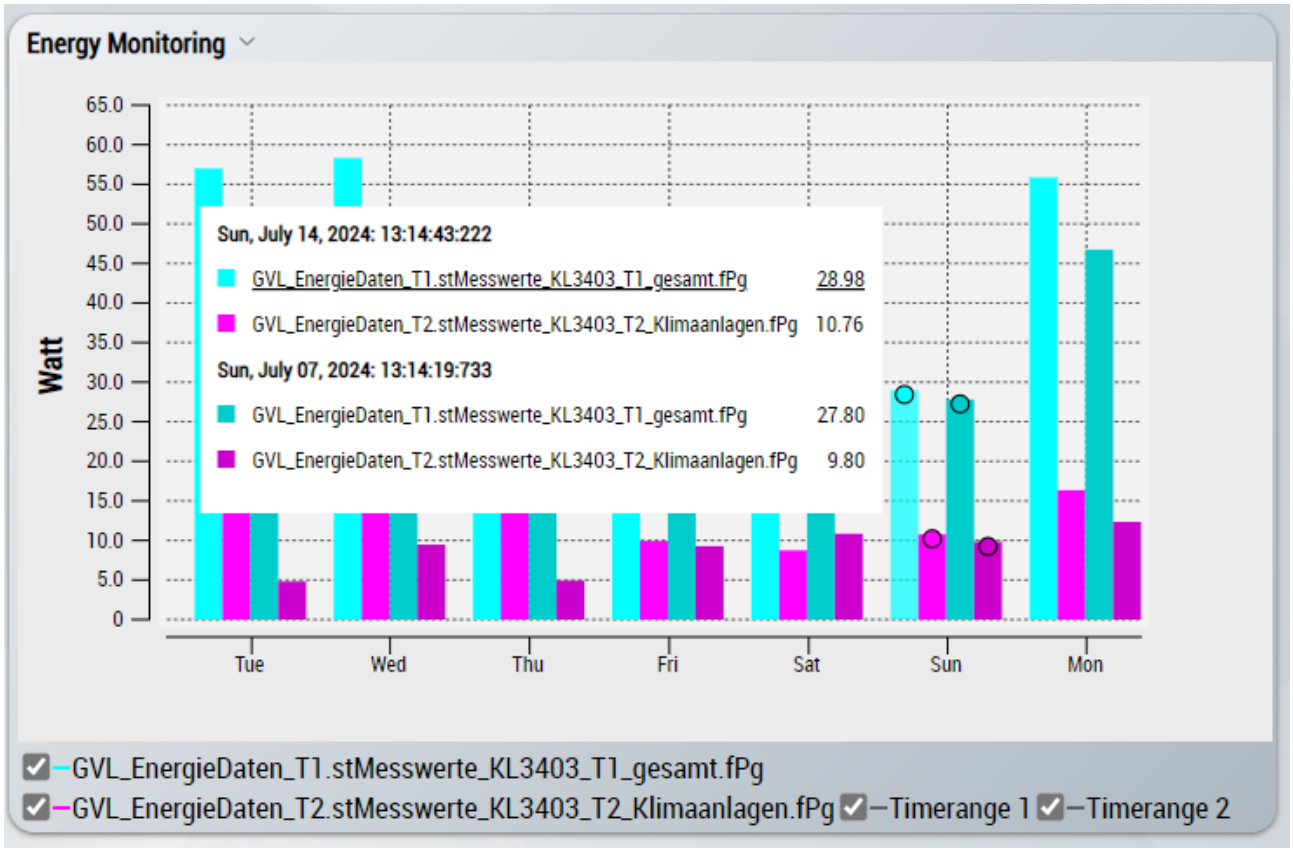
選択した時間範囲ごとにリングが表示され、時間範囲の最後のサンプル値を示します。負の変数値は表示されません。

比較YTチャート



複数の時系列データを比較するのに適しています。選択した比較区間にわたってデータを比較します。比較には複数のモードが用意されています。X軸に比較期間を示します。

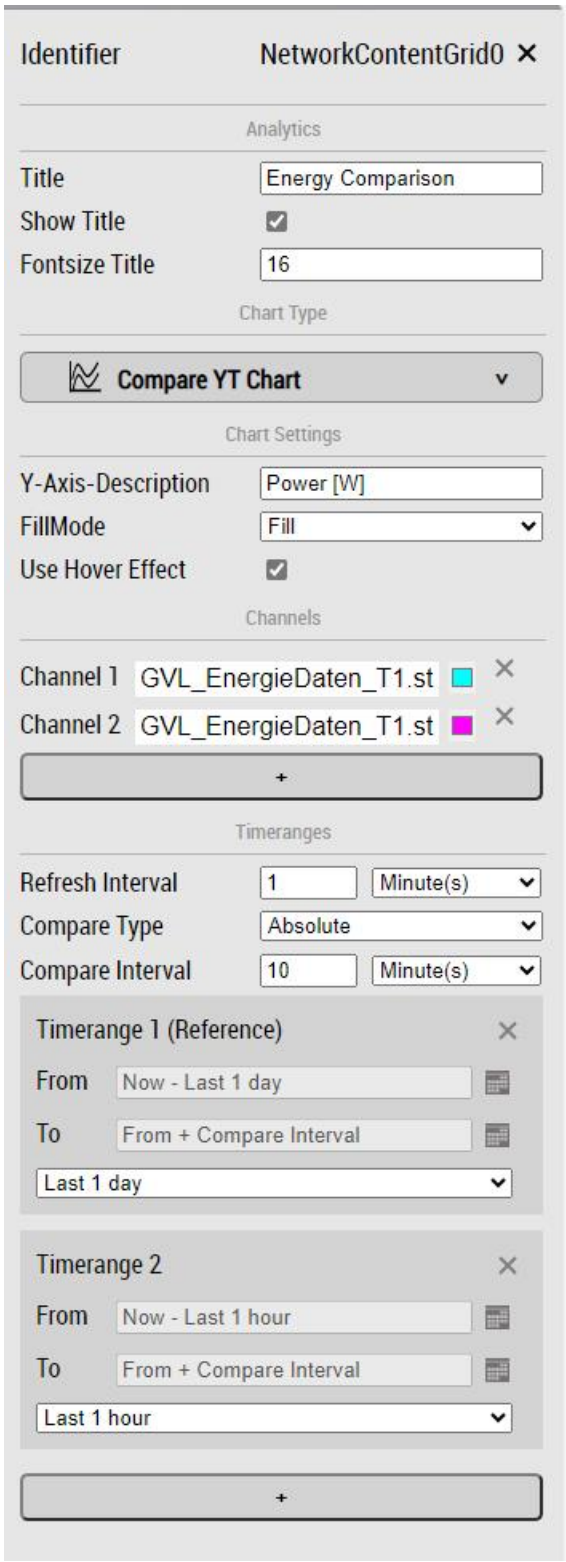
時系列棒グラフ



複数の時系列データを比較するのに適しています。選択した比較区間にわたってデータを比較します。比較には複数のモードが用意されています。棒グラフは、選択した時間区間ごと、および比較する時間区間に基づく変数ごとに生成されます。

6.9.7.3.2 一般的なチャート設定

Chart Settingsエリアでは、一般的なチャート設定ができます。

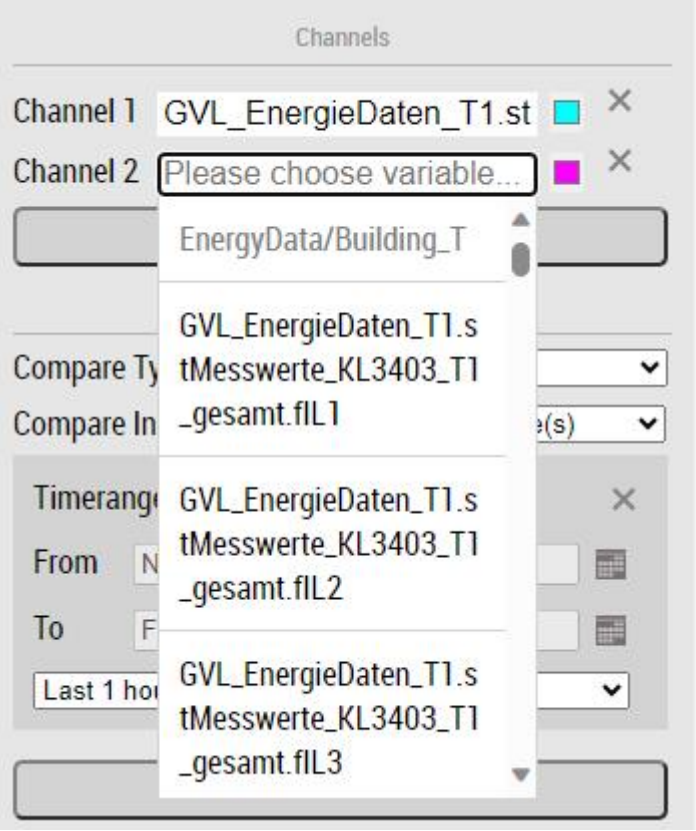


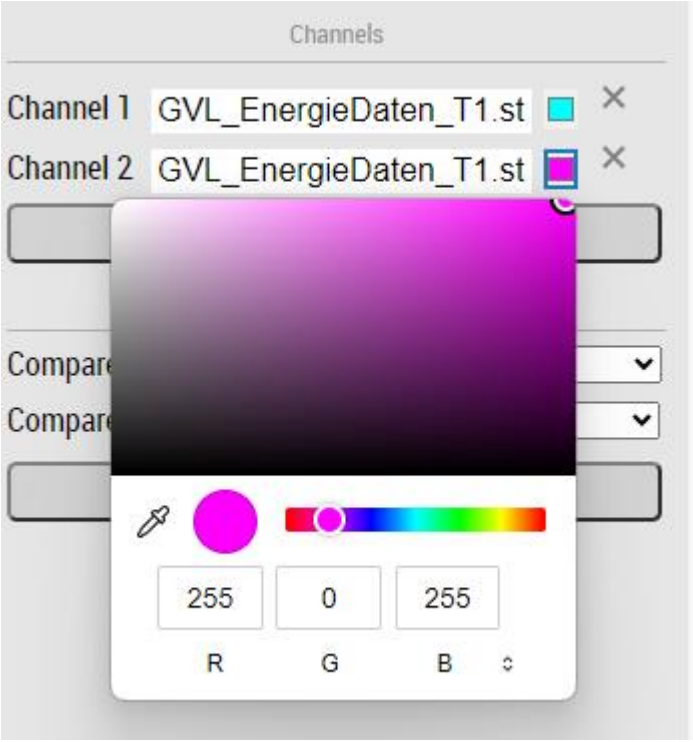
ここで設定できるオプションは以下の通りです：

プロパティ	説明
Y-Axis-Description	Y軸のラベリング。
FillMode	グラフを塗りつぶすかどうかを選択します。
Use Hover Effect	このオプションが有効で、マウスポインタがチャート内にある場合、正確な時点と変数値がウィンドウに表示されます。

6.9.7.3.3 変数の追加

Channelsエリアでは、表示する変数を選択できます。チャートに新しいチャンネルを追加するには、プラスアイコンを使用します。ドロップダウンウィンドウでチャンネルの変数を定義します。カラーアイコンを使用して、チャンネルに個別のカラーを割り当てます。X アイコンを使ってチャンネルを削除します。





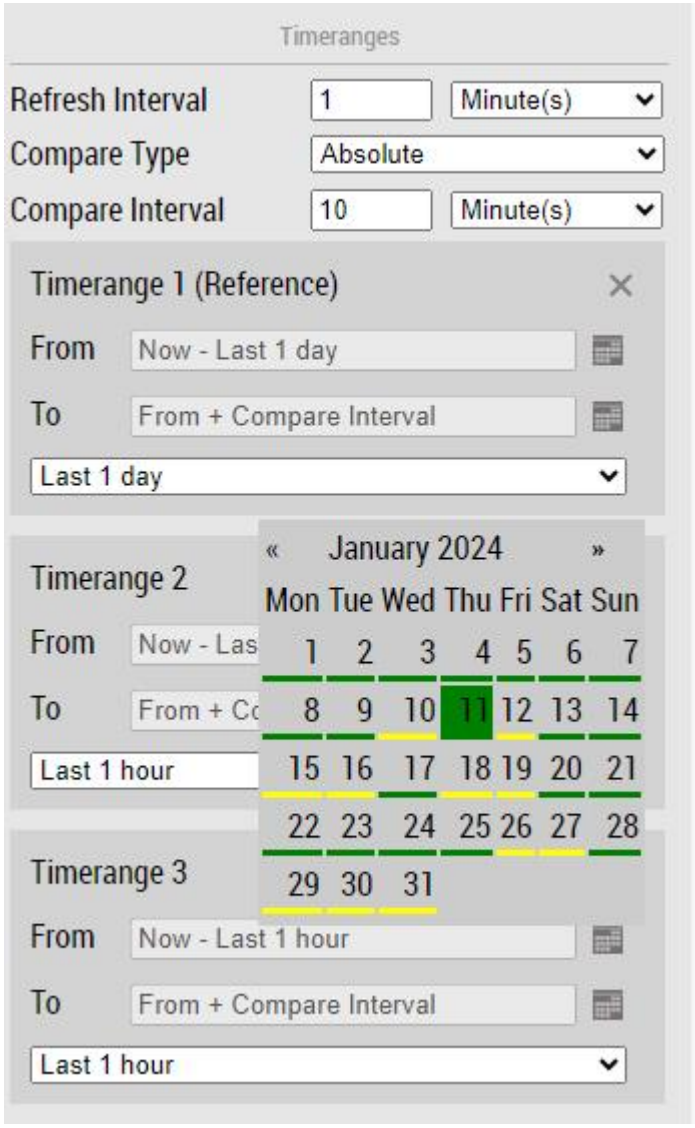
6.9.7.3.4 時間範囲の追加

表示する時間範囲は「Timerranges」エリアで選択できます。プラスアイコンを使って新しい時間範囲をチャートに追加できます。

絶対時間範囲

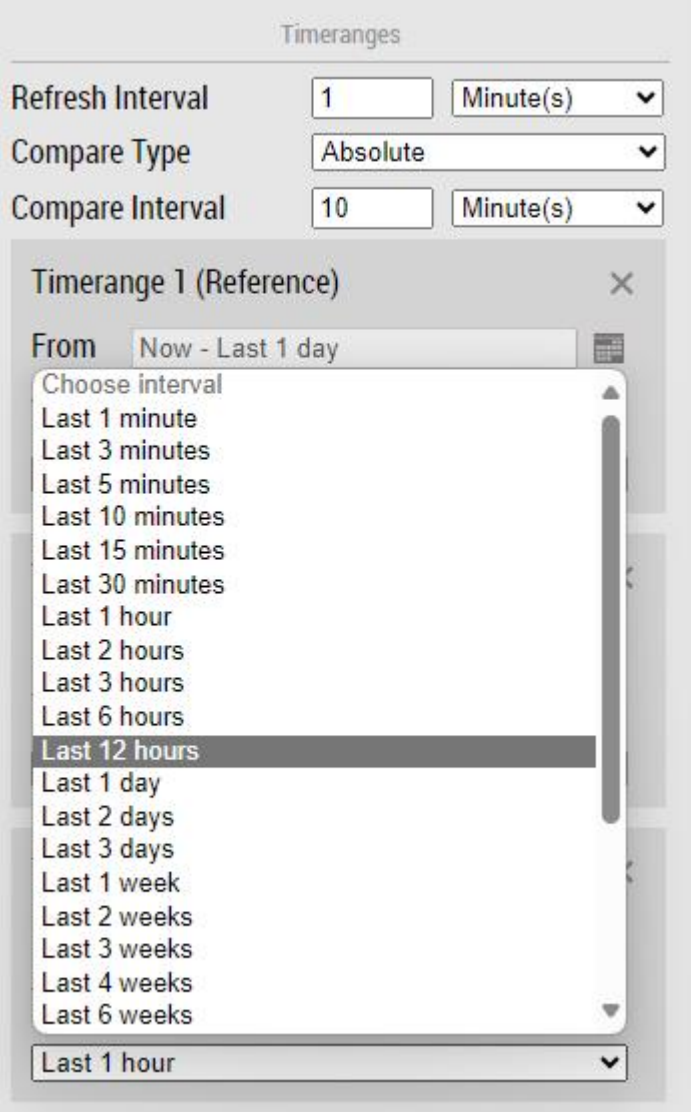
時間範囲については、「From」と「To」パラメータを使用して開始時間と終了時間を選択できます。日付はカレンダー（カレンダーのアイコンをクリックすると開きます）から選択できます。カレンダーは日ごとに色別で示されます：

- 緑: この日は全て記録済みです。
- 黄色： この日は一部記録済みです。
- グレー：この日のデータはありません。



相対時間範囲

絶対時間範囲の代わりに、相対時間範囲を選択することもできます。相対的な時間範囲はドロップダウンウィンドウで選択できます。少なくとも1つの相対時間範囲が選択されるとすぐに、「Refresh Interval」プロパティを使用して、チャートが更新される時間区間を指定できます。



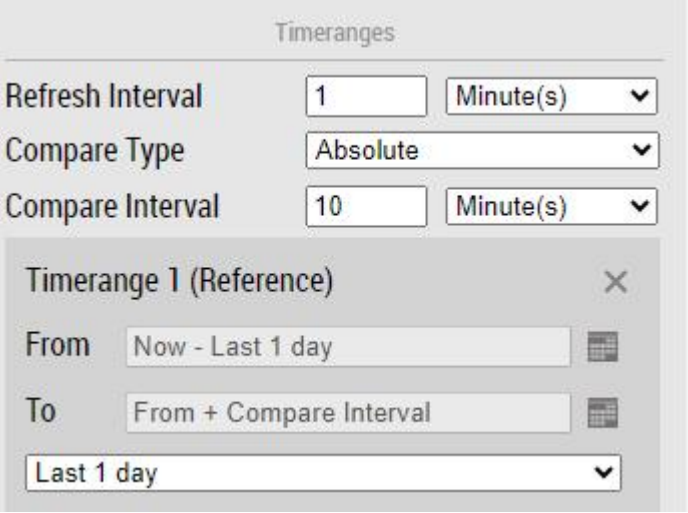
終了時間の自動設定

Compare-YT-ChartとTime-Series-Bar-Chartでは、設定された比較区間に基づいて終了時間が自動設定されます。これらのチャートタイプでは、終了時間を手動で選択できません。

6.9.7.4 データの比較

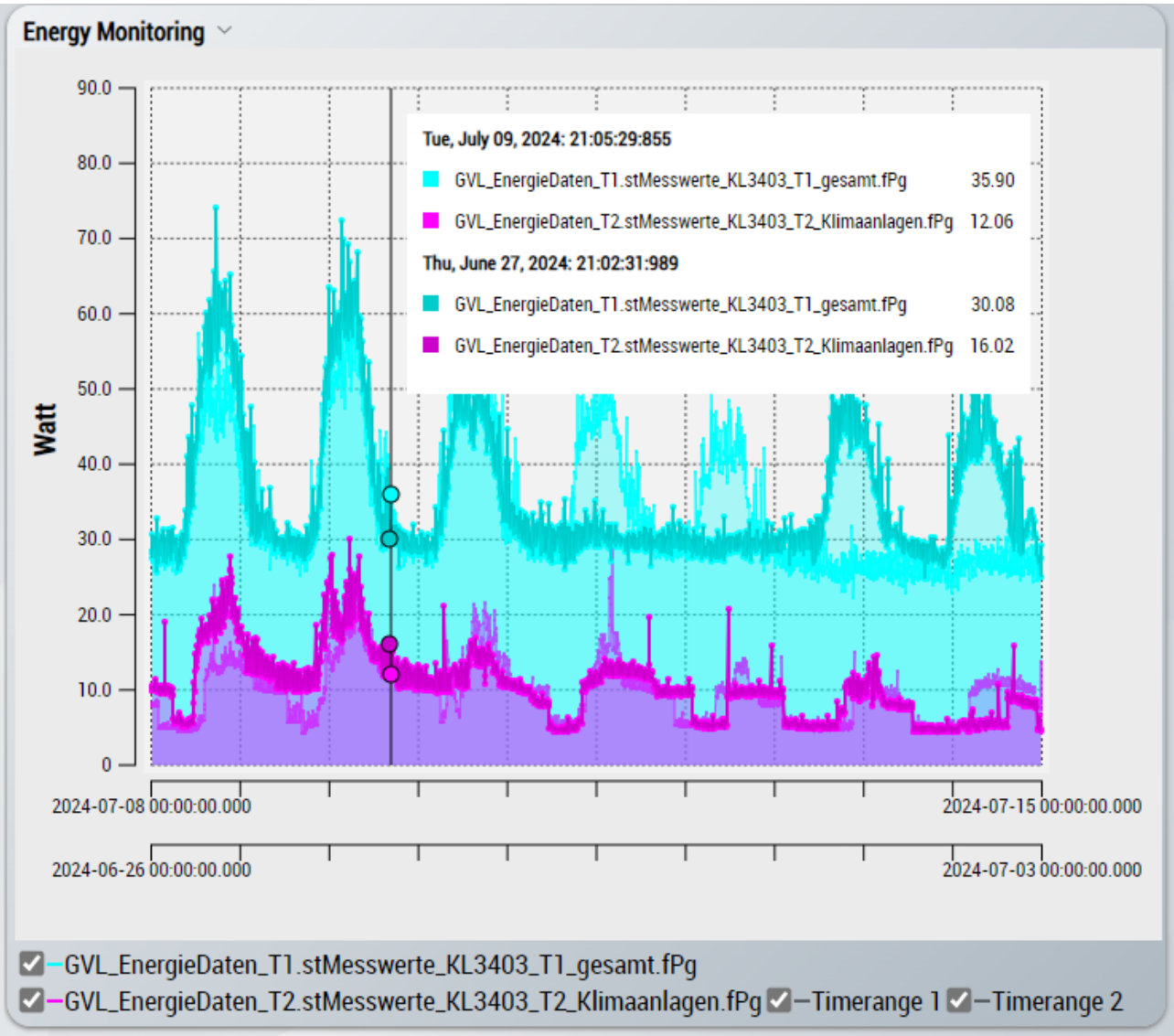
CompareCompare-YT-ChartとTimeSeries-Bar-Chartは、異なる時間範囲のデータを比較するために使用します。比較のため、設定から最大3つの時間範囲を追加できます。1つ目の時間範囲は、表示基準となります。

データは定義されたタイムスパン（例：3時間、1週間、等）にわたって比較されます。チャートのタイプとして Compare-YT-Chart または TimeSeries-Bar-Chart が選択されている場合、比較期間の長さは「Timeranges」 エリアで設定できます。



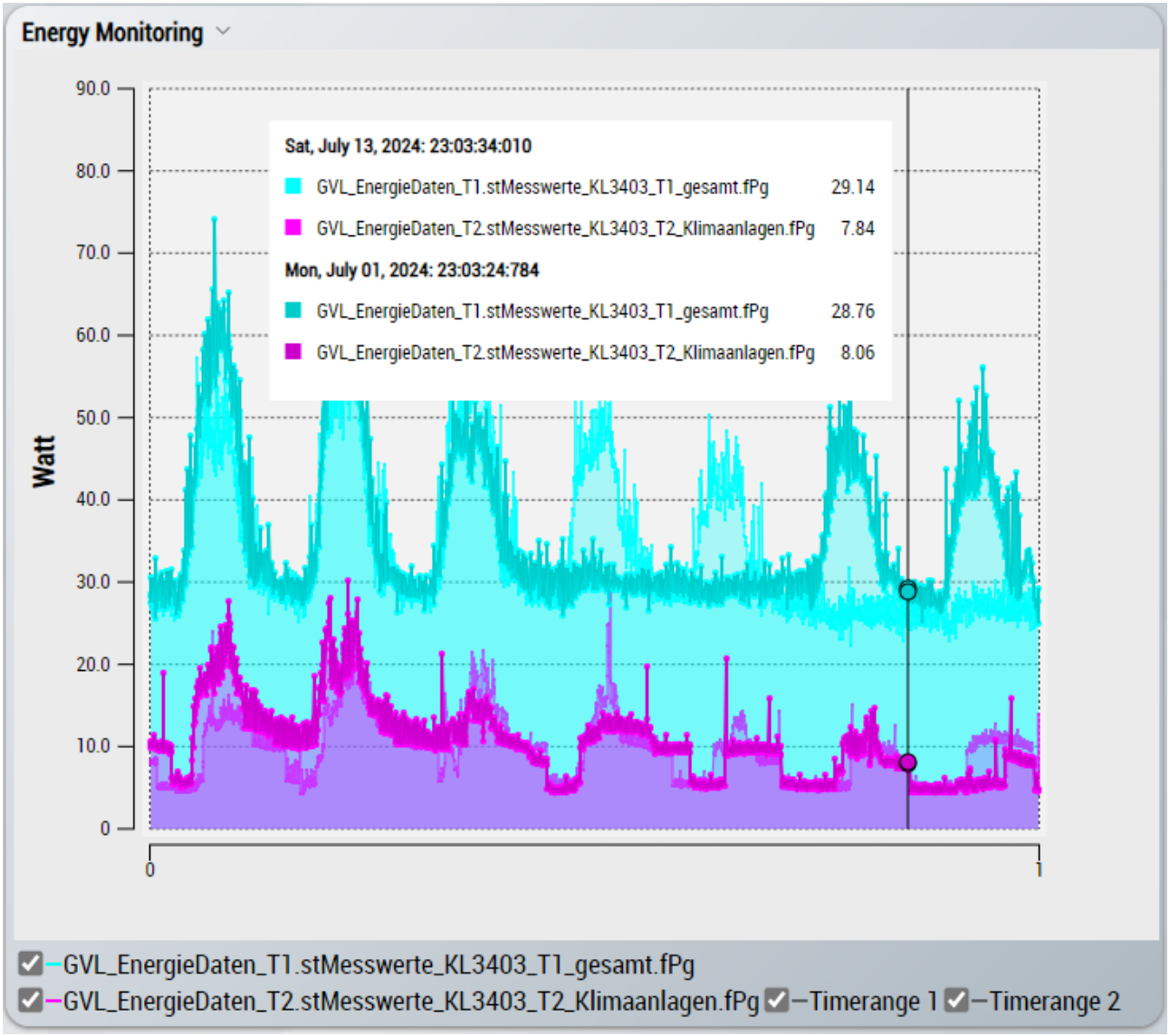
タイムスパンだけでなく、比較モードも設定できます。これは、データを単に上下に並べて表示するか、互いに同期して表示するかを設定できます。これは図中の時間軸のシフトに相当します。利用可能な比較モードは以下の通りです：

Absolute



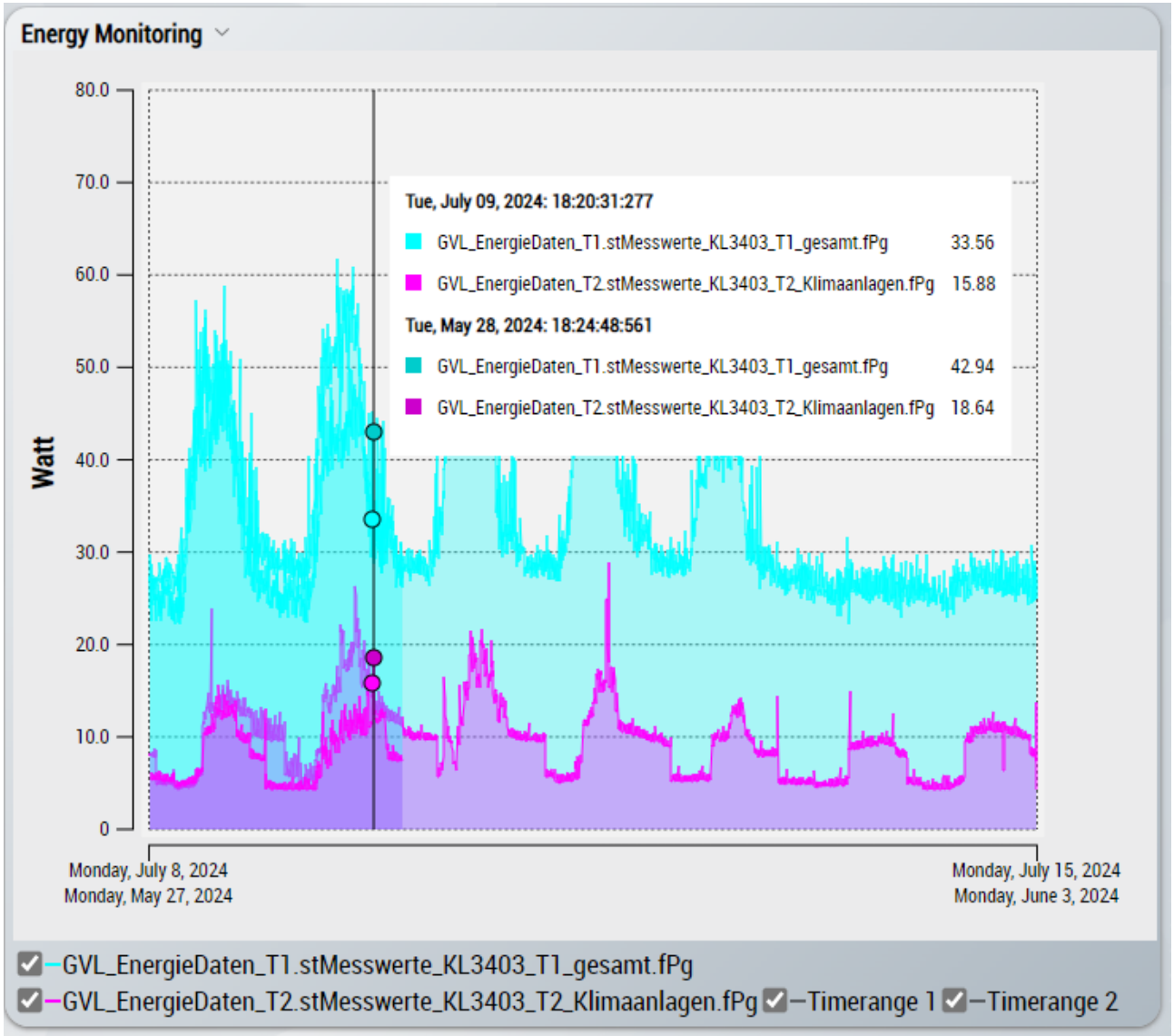
このモードでは、選択した全ての時間範囲が上下に並んだ状態で表示されます。X軸には絶対時間が表示されます。時間軸のシフトはありません。原点は常にタイムスパンの開始時間です。

Relative



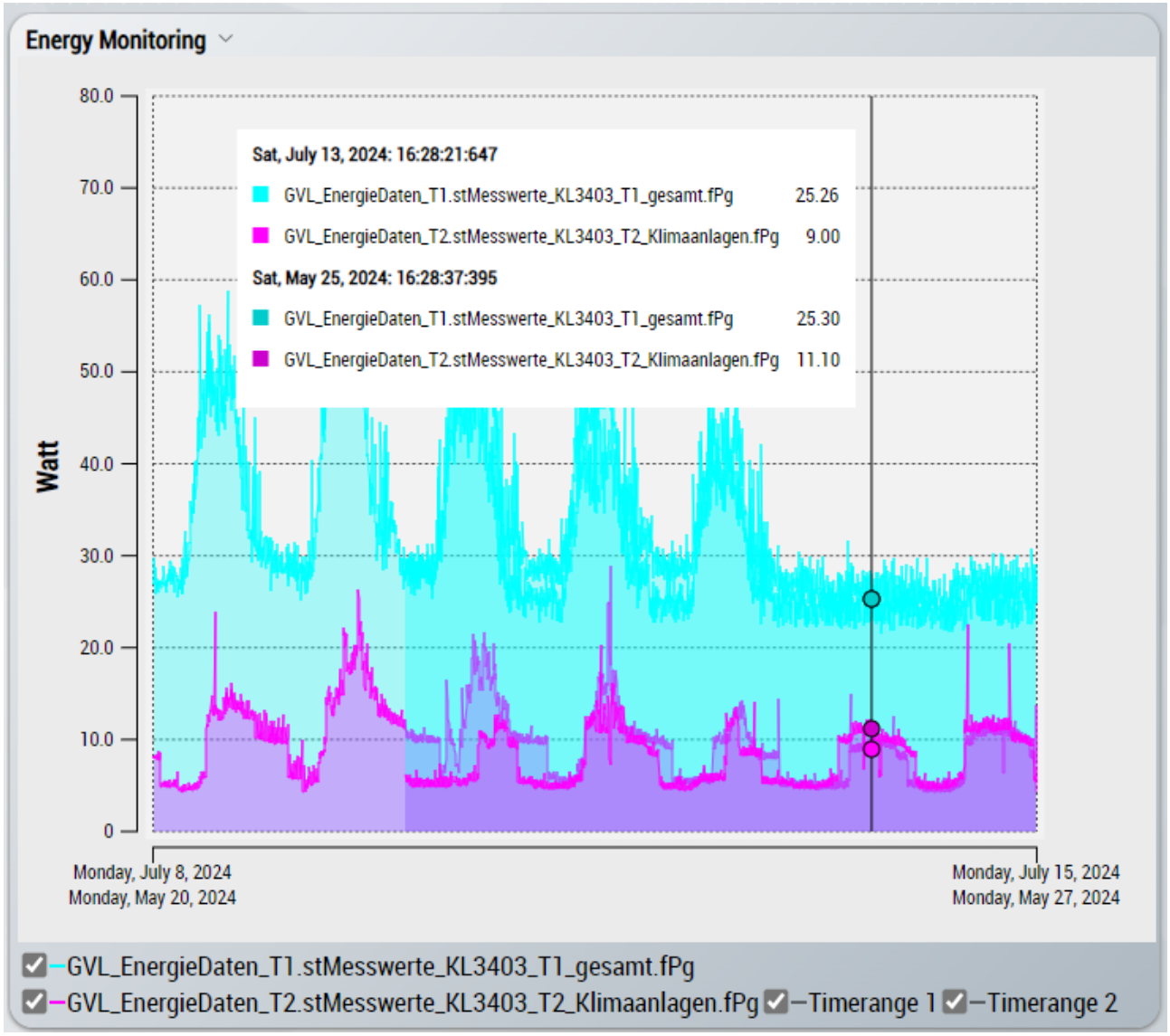
このモードでは、選択した全ての時間範囲が上下に並んだ状態で表示されます。相対的な経過時間がX軸に示されています。時間軸のシフトはありません。

Match Start



このモードでは、データが正しい時間に合わせて重ねて表示されます。1つ目の時間範囲の開始時間が基準となります。他の時点のデータは、チャートの先頭のが揃うようにシフトされます。その基準となるのが、選択した比較期間です（表を参照）。

Match End



このモードでは、データが正しい時間に合わせて重ねて表示されます。1つ目の時間範囲の終了時間が基準となります。他の時点のデータは、チャートの最後に時間が合うようにシフトされています。その基準となるのが、選択した比較期間です（表を参照）。

比較期間	手順
Second(s)	ミリ秒単位のデータの比較。同じミリ秒に達すると、データは重ねて表示されます。
Minute(s)	秒単位のデータの比較。同じ秒に達すると、データは重ねて表示されます。
Hour(s)	分単位のデータの比較。同じ分に達すると、データは重ねて表示されます。
Day(s)	時間単位のデータの比較。同じ時間に達すると、データは重ねて表示されます。
Week(s)	曜日単位のデータの比較。同じ曜日に達すると、データは重ねて表示されます。
Month(s)	日付単位のデータの比較。同じ日（日付）に達すると、データは重ねて表示されます。 1カ月は常に31日に相当します。
Year(s)	月単位のデータの比較。同じ月に達すると、データは重ねて表示されます。

データのシフト

データが同じサイクルタイムで記録されていない場合や、同じサンプリングレートでスキャンされていない場合、値が必ずしも正確に重ねて表示されない可能性があります。

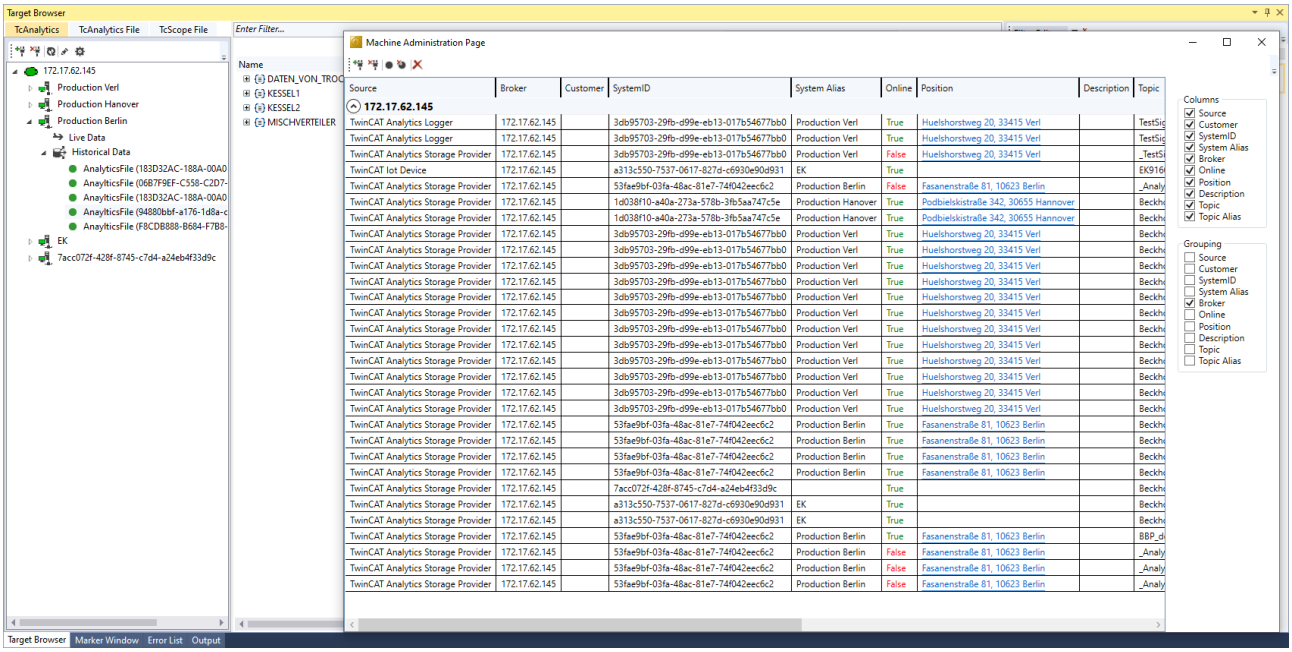
6.9.8 HMIダッシュボードで複数の機械を切り替える

TwinCAT Analyticsでは、複数の機械から異なるデータストリームを使用し、解析の中でデータを切り替えることができます。これは、Virtual Input Sourceを介した Analytics Workbench でも、生成された PLC およびHMIでも可能です。ライブデータと履歴データの両方を使用できます。各データストリームについて、簡単な説明と場所をAnalytics LoggerまたはMachine Administration [▶ 29]で追加できます。

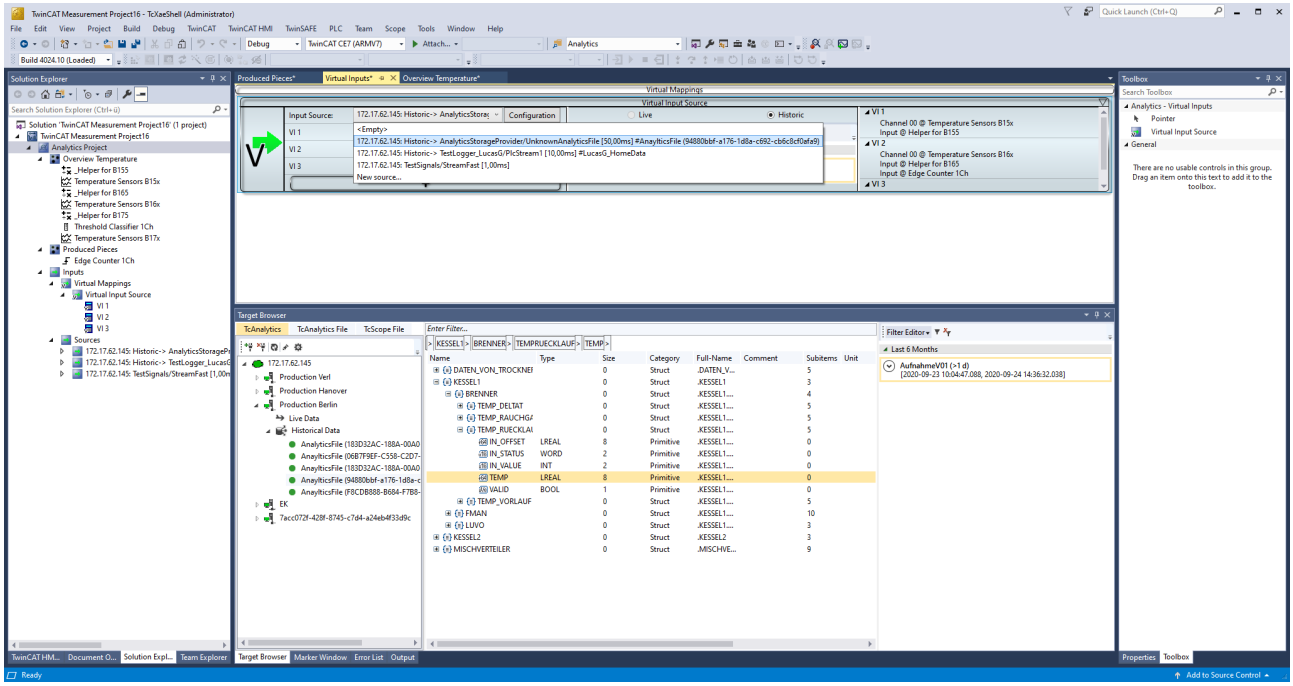
Machine Administration page（機械管理ページ）

で設定されていない場合は、データの解析 [▶ 29] のページで機械のメタデータを設定します。（**TwinCAT > Target Browser > Target Browser**）を開き、歯車のアイコンをクリックします。機械の場所、簡単な説明、機械名を入力します。Analyticsプロジェクト内の既存のデータには、このメタ情報は割り当てられていません。Target Browserからドラッグ＆ドロップでのみ情報が転送されます。この場合、データストリームを削除して再作成できます。

Analytics設定で使用するデータストリームごとに、ランタイムへのデプロイ [▶ 357] でマップエントリが作成されます。これらのマップエントリは、HMI ダッシュボードのスタートページにある一般マップの入力変数として使用されます。

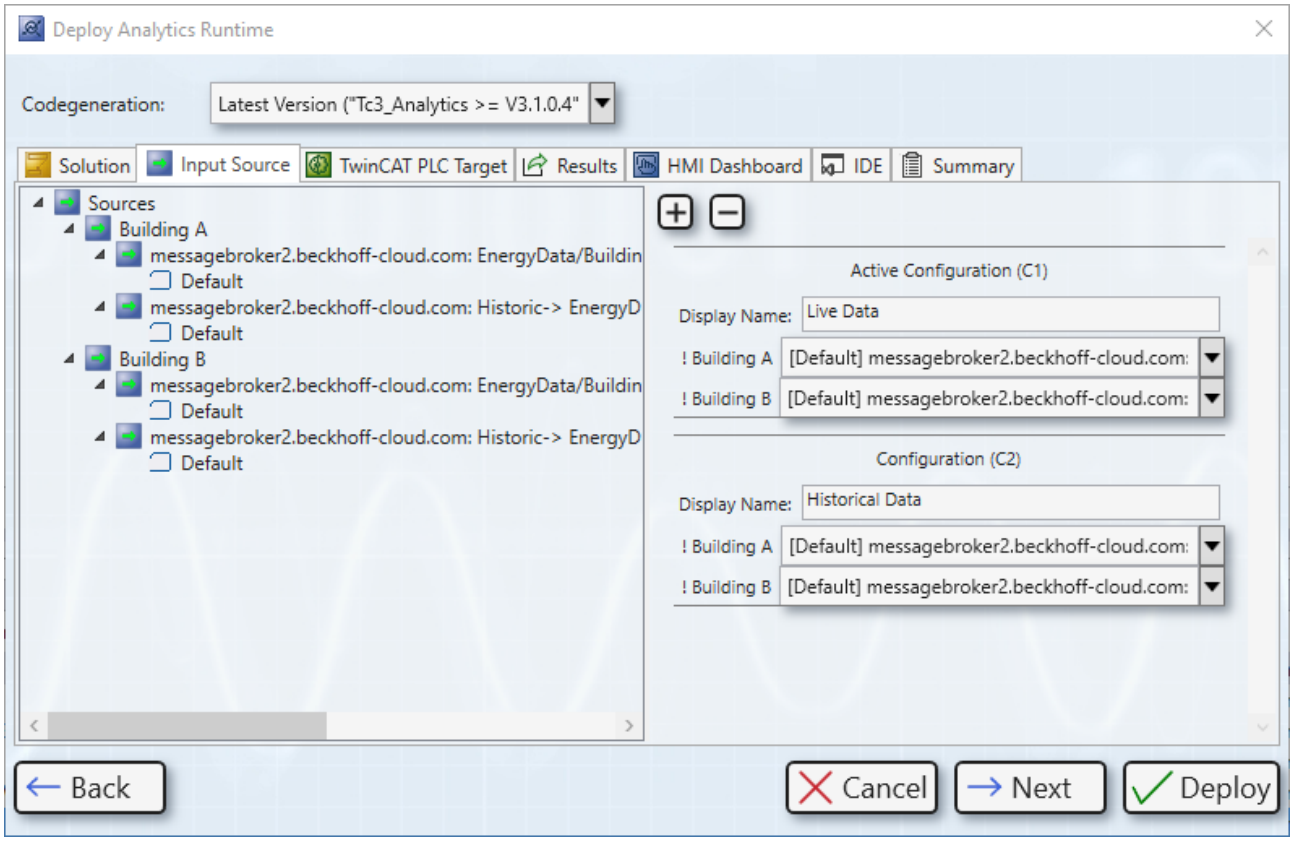


データの解析 [▶ 29] で、データストリームは**Sources**の下にリストされています。これらは、 の様々なソースデータを使用して作成されています。これらのソースはVirtual Input Sourceにリストされています。ソースの切り替えは可能です。（Workbenchのスイッチに相当する）各ソースを1回ずつクリックして、全ての入力がリンクされていること、いずれの入力も**Empty**に設定されていないことを確認します。その後、PLCでHMIを生成できます。

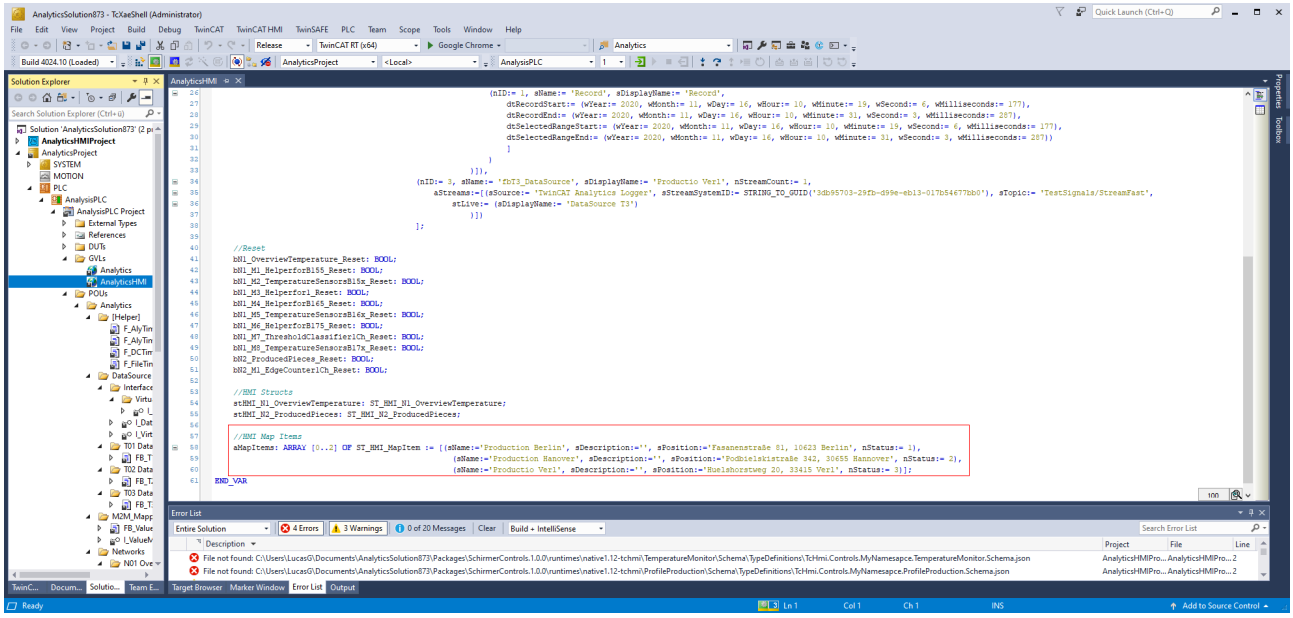


デプロイウィザード

ランタイムへのデプロイ [▶ 357] では、**Input Source**に任意の数の設定を作成でき、HMIで切り替えることができます。これにより、解析HMIでは、異なる機械からのライブデータと履歴データを解析し、切り替えることが可能になります。リストされた各設定は、並行して解析できます。常に、Analytics Workbenchで設定されている数と同じ数のVirtual Input Sourceを持ちます。

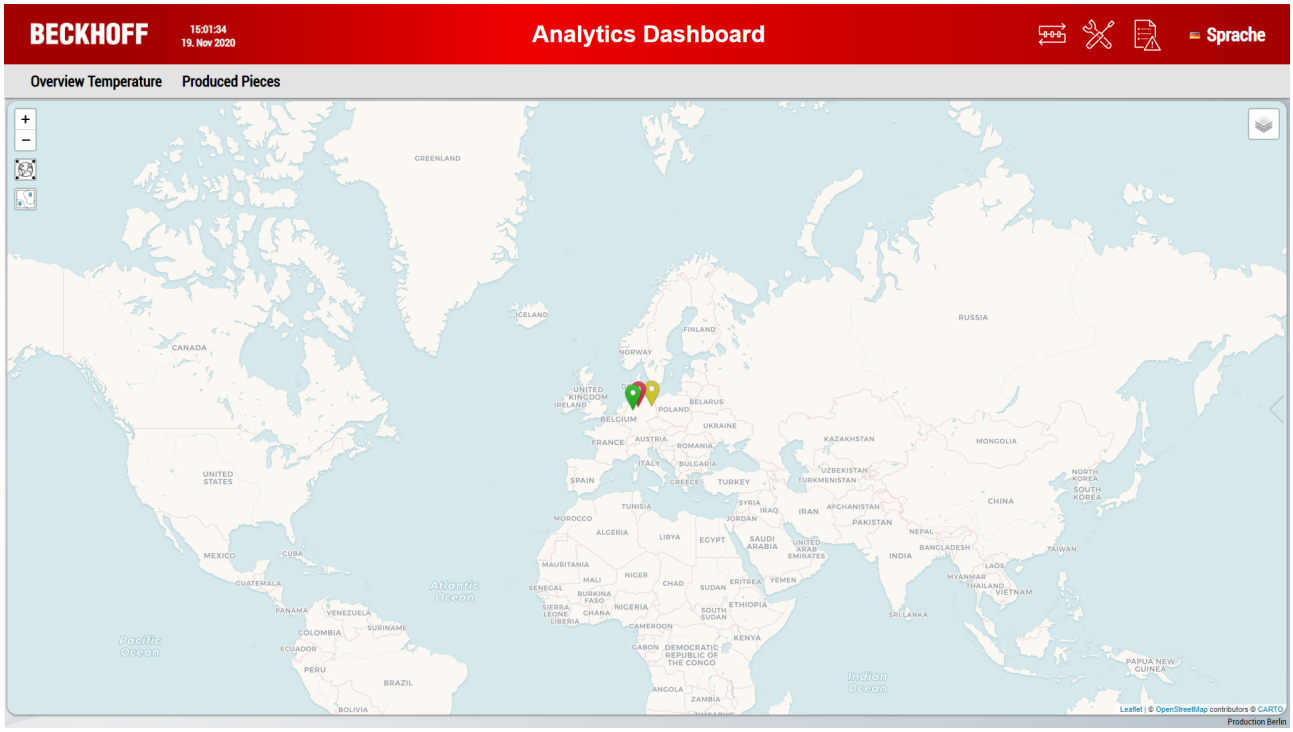


HMI と PLC の生成に成功すると、入力ソースの名前、位置、説明が PLC のグローバル変数リストに入力されます。設定に誤りがある場合、または後から値を変更したい場合は、PLCで直接変更できます。

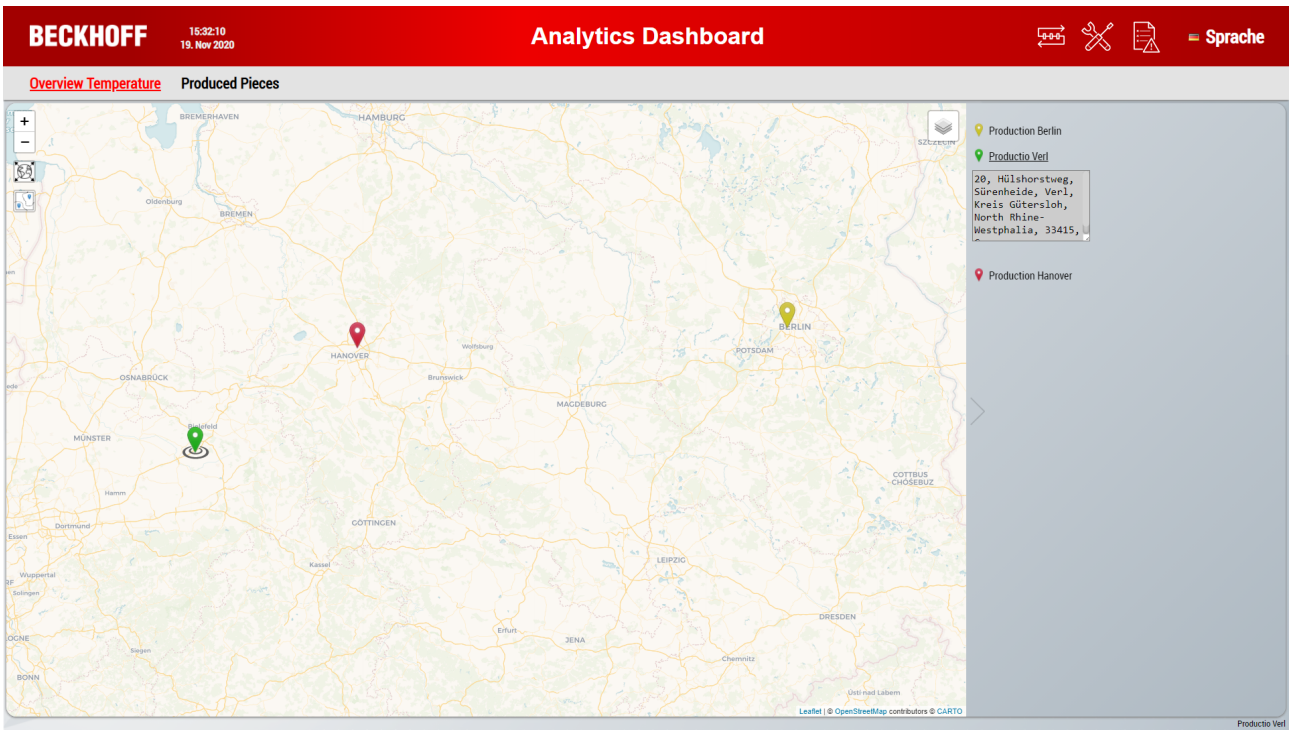


機械のマップ

スタートページの作成を選択した場合、HMI ダッシュボードのホームページに*Map Control*が作成されます。マップには全ての機械の場所が表示され、凡例に機械名（システムエイリアス）が記載されます。凡例は右の矢印で開閉できます。アイコンの色は、現在の機械の状態を示します：緑＝OK、黄＝警告、赤＝アラーム。

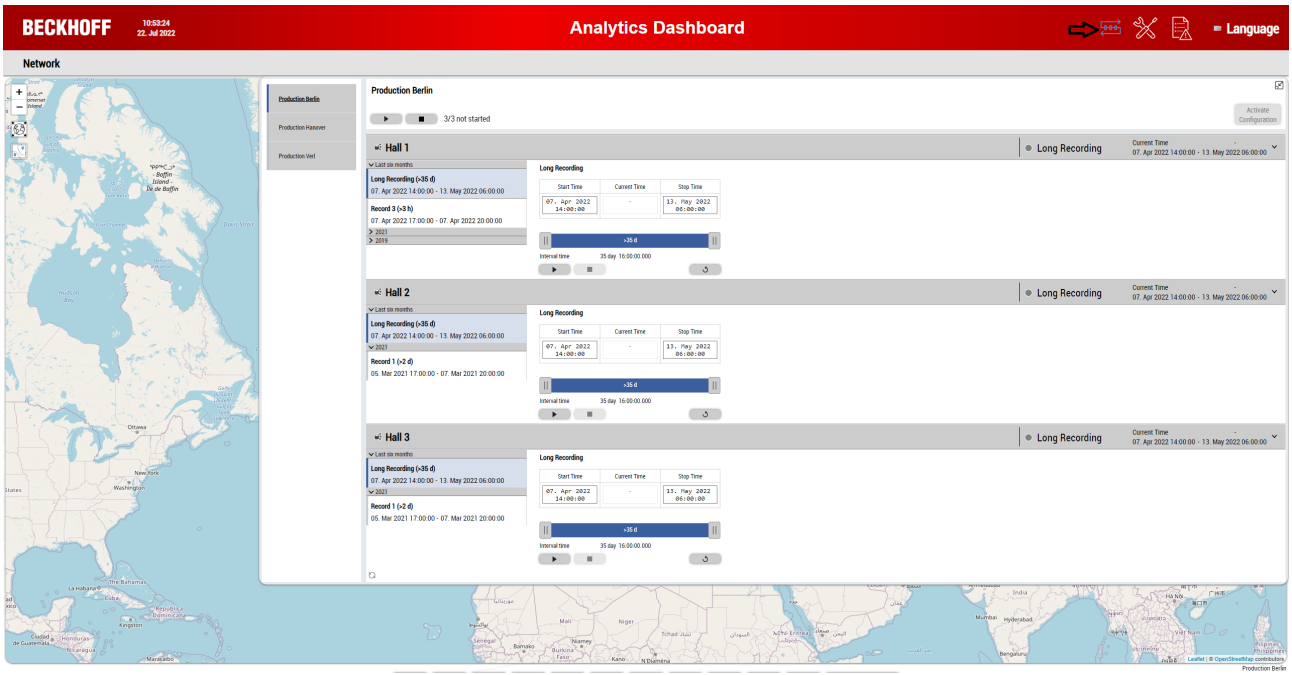


アイコンをダブルクリックすると拡大表示され、左側のアイコンを使うとズームアウトし、全ての機械の概要が表示されます。地図上のアイコンをクリックすることもできます。対応する凡例項目は自動的にハイライトされます。凡例項目をクリックすると、対応するアイコンが自動的にハイライトされます。



機械データの切り替え

各機械の設定マップに加え、各設定の解析結果を切り替えることができます。PLC では全ての解析が別々のタスクで実行されるため、全ての解析データが同時に利用できます。生成されたHMIでは、特別に開発されたコントロールで様々な機械を切り替えることができます。コントロールの開閉は、下の画像でハイライトされているアイコン（黒い矢印）で行います。

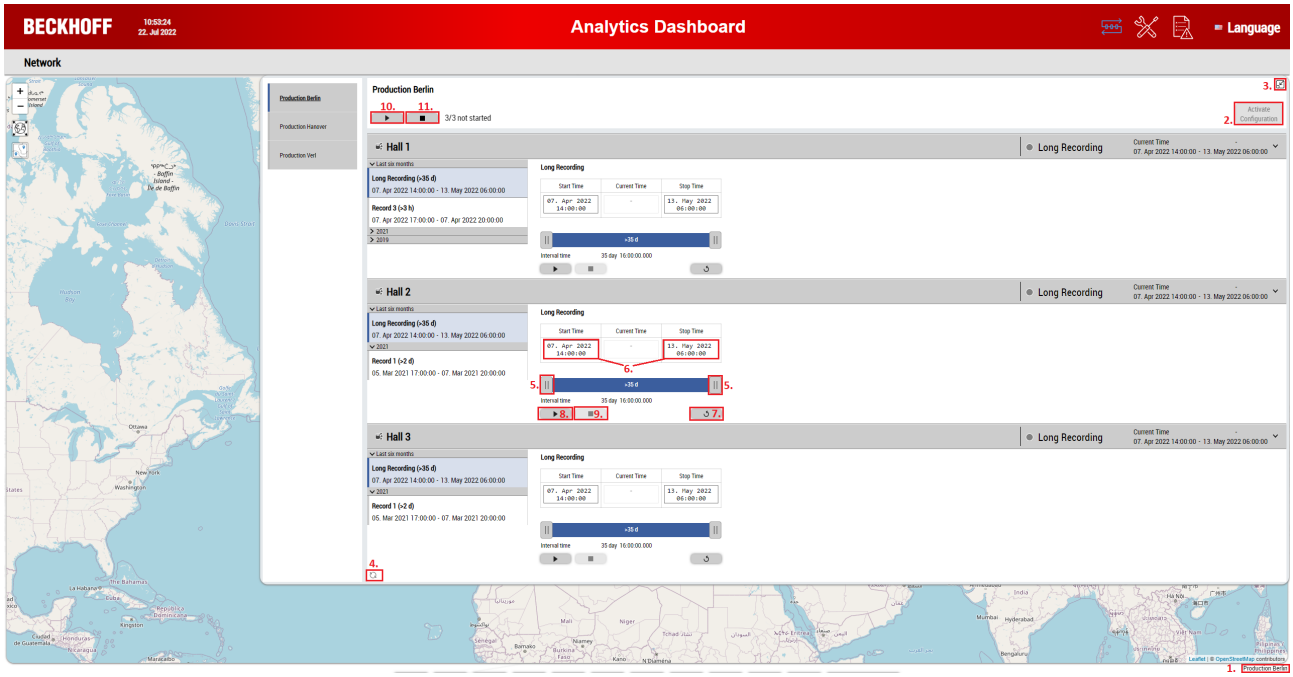


これまで地図上に表示されていた設定は、コントロールから選択できるようになっています。設定はライブデータと履歴データから成り、履歴データのみが**Production Berlin**に設定されます。履歴データは、HMIの履歴データの取り扱い [▶ 355] と同様に解析できます。

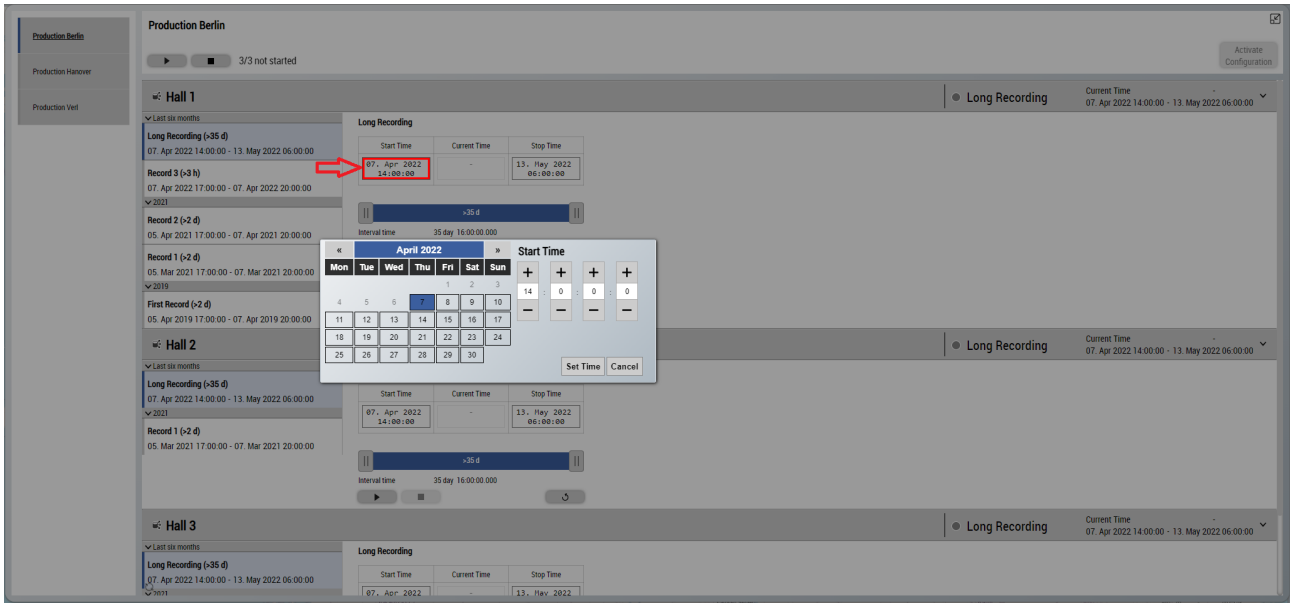
他の機械をクリックすると、設定はコントロールにのみ表示されます。アクティベートするには、**Activate Configuration**をクリックします。
コントロールについては以下の通りです。

1. 現在アクティブな設定。

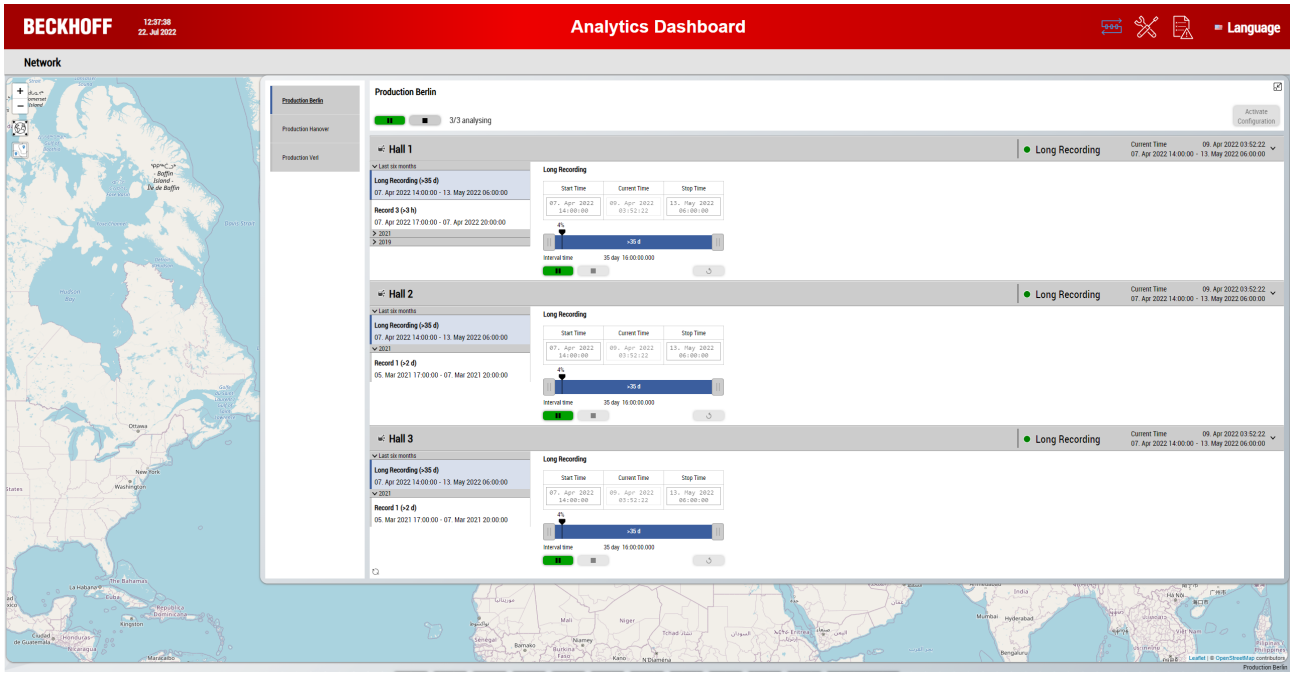
2. 選択した設定のアクティベートボタン。
3. 選択ウィンドウを最小化（選択した設定を自動的にアクティベート）。
4. 記録リストの再読み込み。これにより、後から記録されたデータをダッシュボードに取り込み、柔軟に解析できます。
5. スライダーで開始時間と終了時間を選択。
6. カレンダー付き選択ウィンドウで開始日と終了日を選択。
7. 開始時間と終了時間をリセット。
8. 選択した**Input Source**の解析を開始。
9. 選択した**Input Source**の解析をキャンセル。
10. 解析全体を開始。
11. 解析全体をキャンセル。



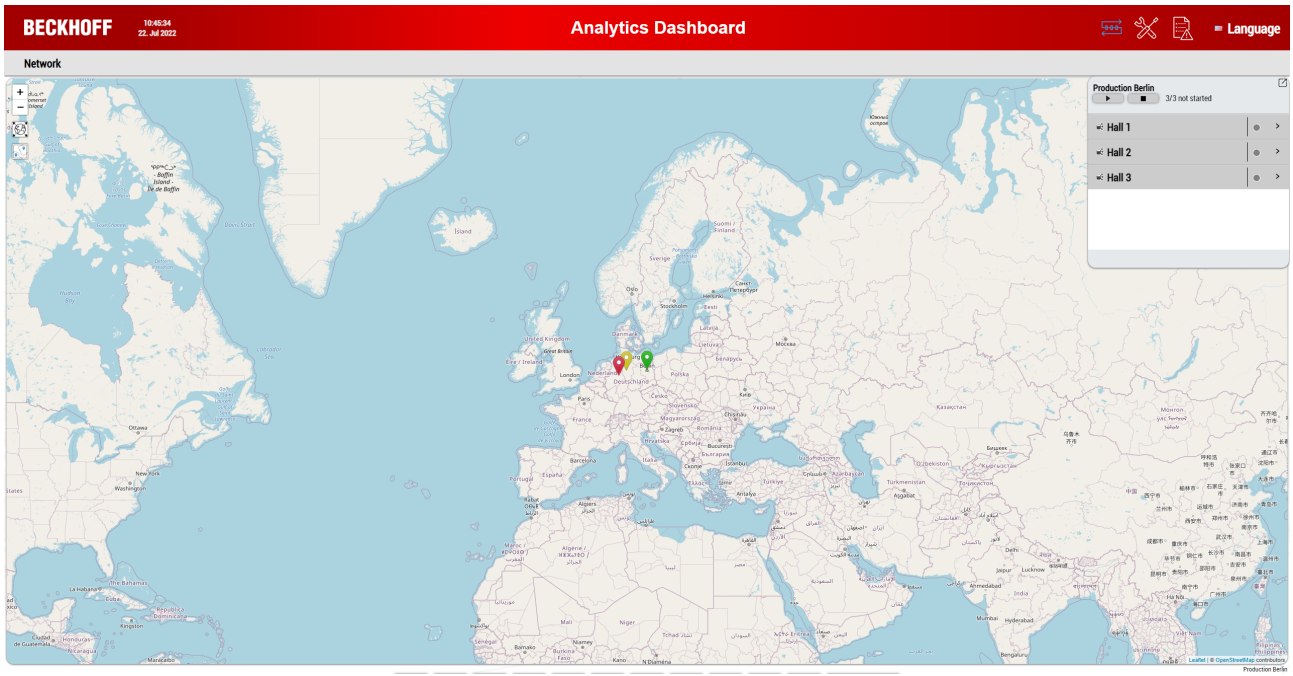
スライダーはクイックテストや開始・終了時間のおおよその設定に最適です。ミリ秒単位で正確に設定したい場合は、**text field**をクリックしてください。カレンダー付きの選択ウィンドウが開き、日付の選択と正確な時間の設定ができます。

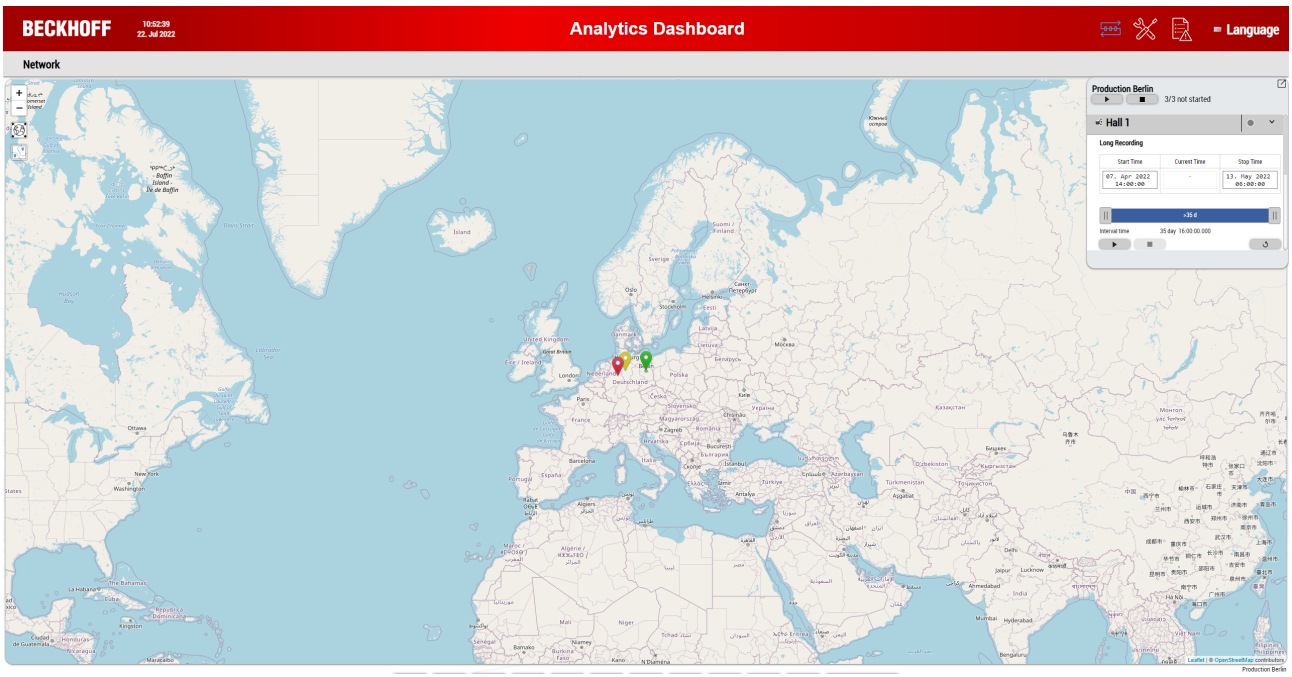


Startボタンをクリックすると、PLC 内の全ての入力ソースで解析がトリガされます。解析が開始されるまでローディングシンボルが表示されます。最初のデータが表示されると緑色になります。同様に、スライダー上のマーカーは、履歴解析における現在の時刻を示します。処理速度は、帯域幅、データ数、記録データのサイクルタイムに依存します。機械を切り替えるだけで、複数の履歴データの記録を同時に解析できます。内部的には、全ての解析は並行して継続します。これにより、ダッシュボードで解析を簡単に切り替えることができます。



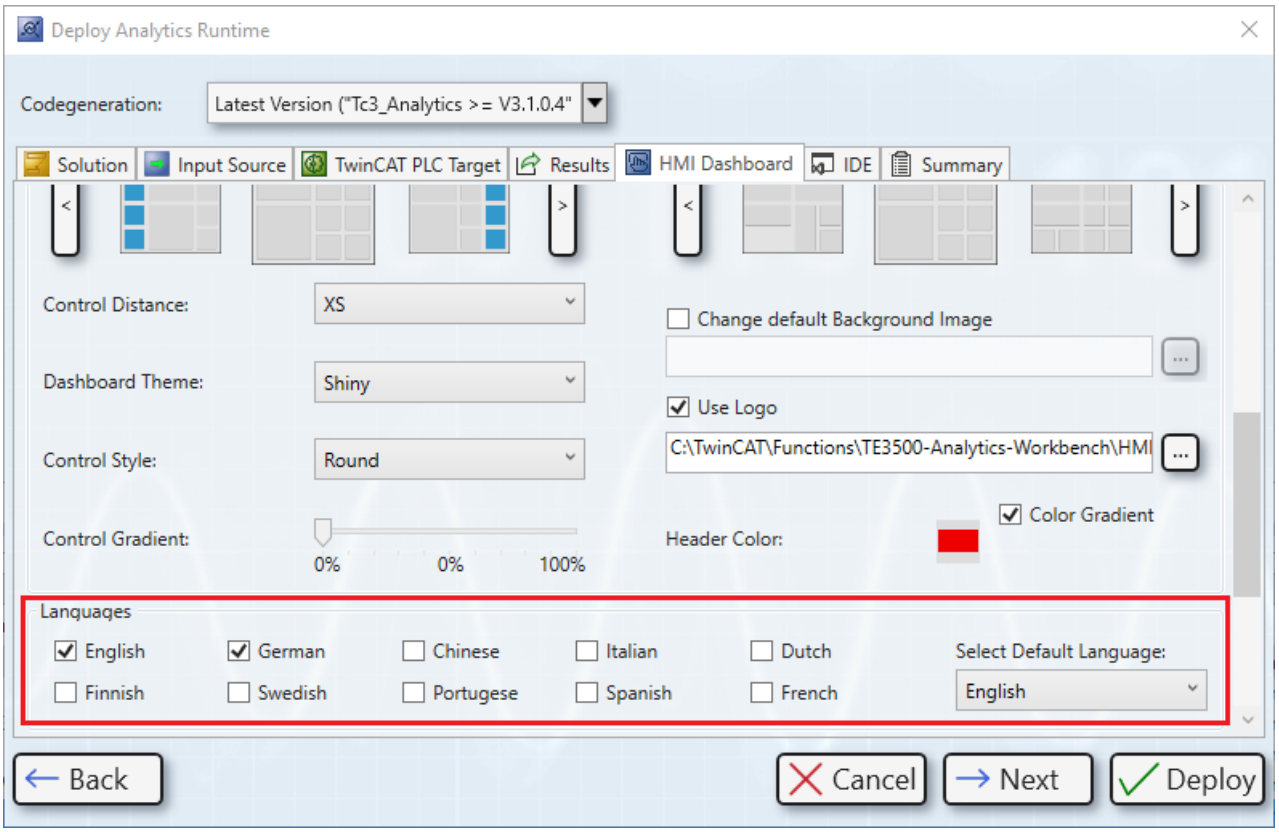
右上の**Minimize**ボタンは、設定と記録の選択を非表示にします。これにより、入力ソースから記録データを取り扱い、同時にデータ表示できます。上部の青いアイコンをクリックすると、ウィンドウを完全に非表示にできます。ダッシュボードの右下には常に設定名だけが表示されています。





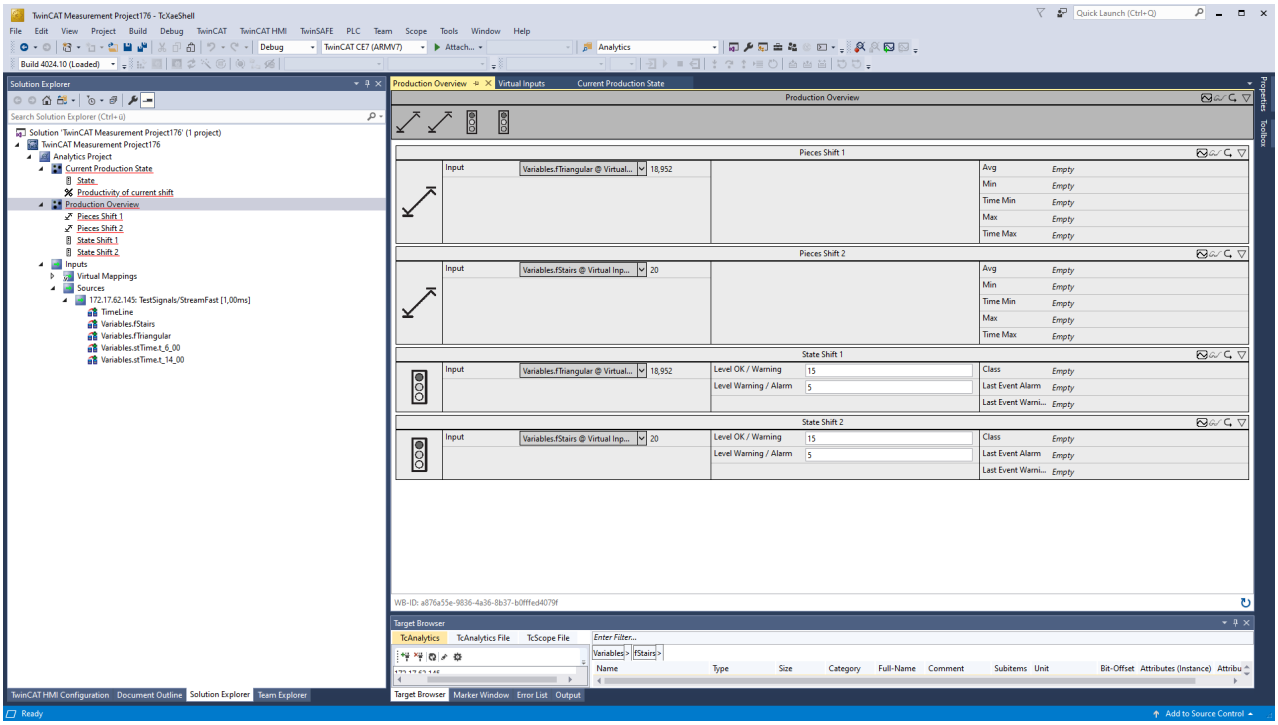
6.9.9 言語切り替え機能の統合

Deploy ウィザードの設定ウィンドウでは、HMI エリアの言語切り替えに最大 8言語を選択できます。

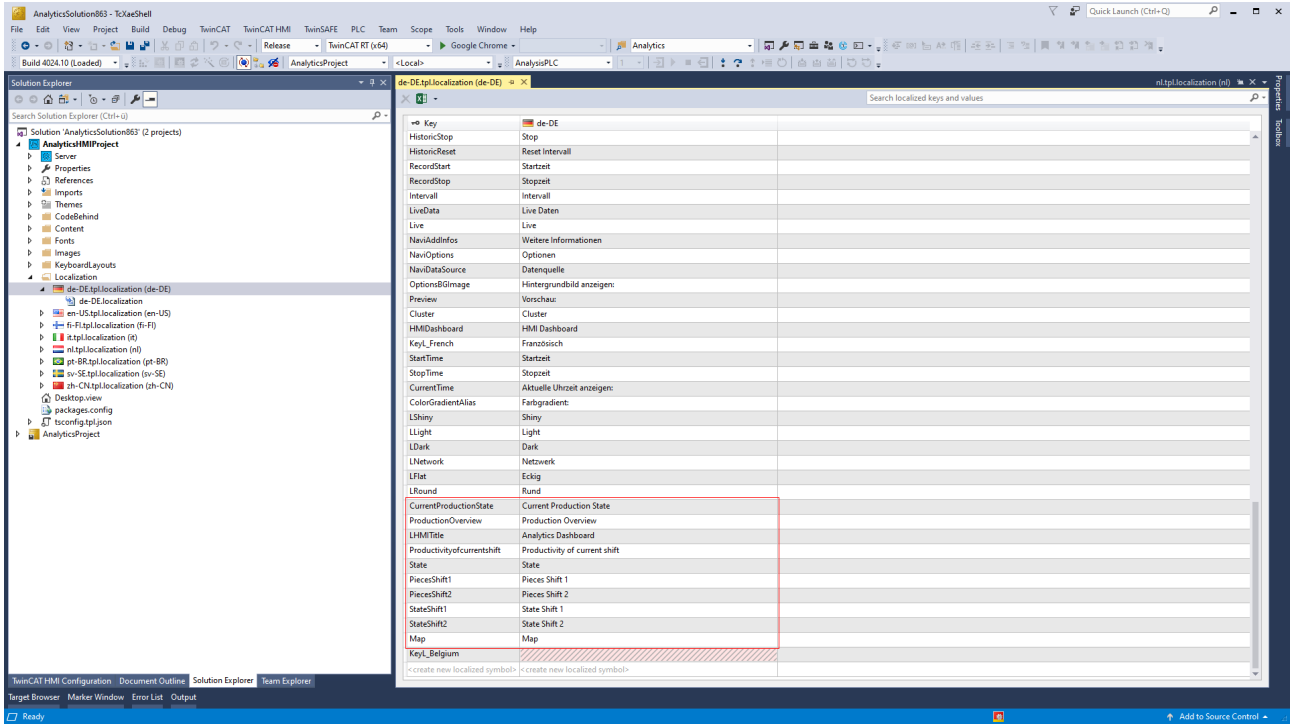


ユーザ管理を使用している場合、作成されたユーザの言語は既に選択されています。これらは選択解除できません。

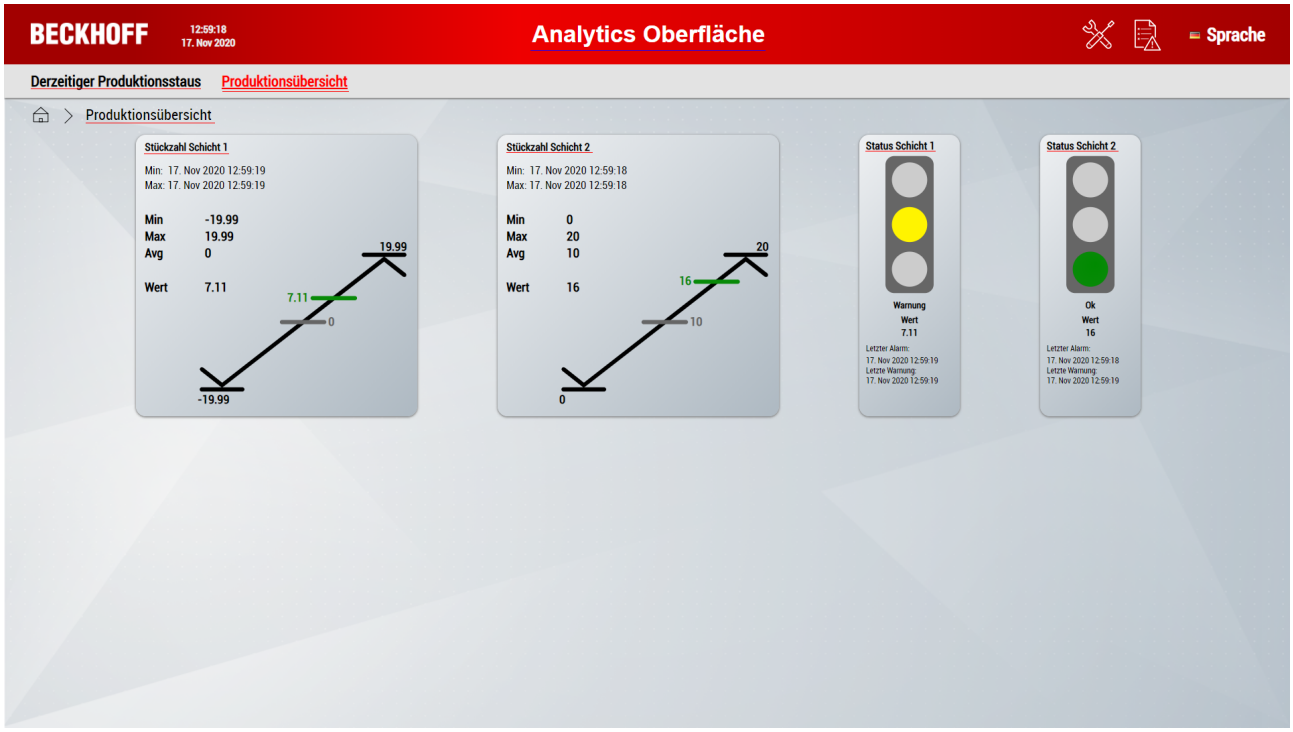
提供されているAnalyticsコントロール内のテキストや、その他すべてのテキストは、自動的に切り替えることが可能です。切り替え言語に含める場合は、ネットワーク名とモジュール名のみ翻訳する必要があります。以下のスクリーンショットの通り、影響を受ける主な名前が**Solution Explorer**でマークされています。



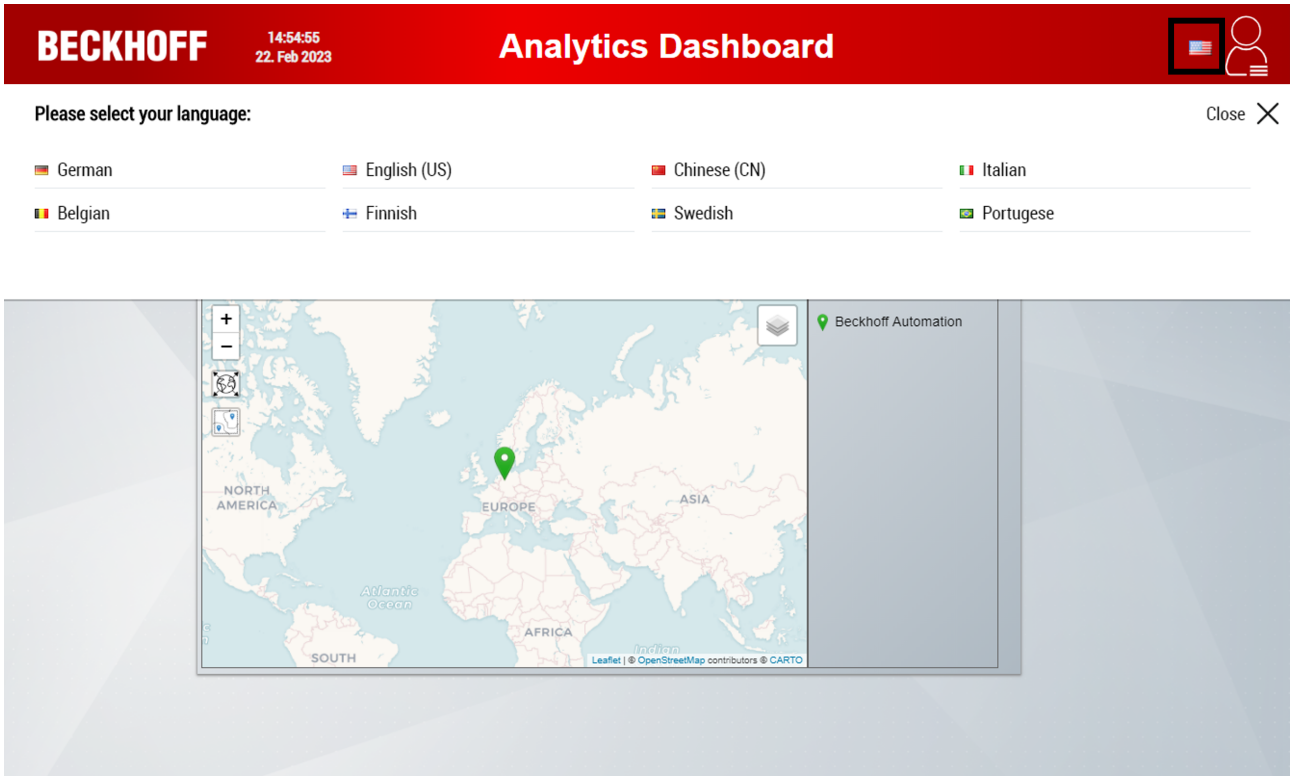
生成されたTwinCAT HMIエンジニアリングプロジェクトでは、翻訳エントリが既に準備されているため、この作業は簡単に行うことができます。各言語のテキストは、**Localization**に保存されます。ネットワークとモジュールの名前は自動的に入力されるので、翻訳のみ対応が必要です。これを行うには、翻訳が必要な言語のファイルを全て開きます。次のスクリーンショットは、ドイツ語に翻訳する必要があるエントリを一覧表示しています。処理が必要なエントリの数は、Analyticsプロジェクトの複雑さによって異なります。



完了したら、**Google Chrome**ボタン（またはデフォルトブラウザの名前）をクリックしてダッシュボードを開くことができます。ブラウザでリロードするだけでは不十分で、プロジェクトを内部的に再構築する必要があります。以下の画像では、言語スイッチで切り替えられたテキストがハイライトされています。



言語は右上の旗から変更できます。



言語はDashboardバージョン2.0以降でユーザごとに設定・保存できます。

6.9.10 標準 TwinCAT HMIプロジェクトでのインタラクティブ機能の使用

インタラクティブダッシュボード機能とインタラクティブチャート機能は、ワンクリックダッシュボードからは独立して、TwinCAT HMIプロジェクトに統合できます。これには、ターゲットシステムでTwinCAT Analytics Runtime、またはRuntime Baseライセンス（TF3550またはTF3551）が必要です。

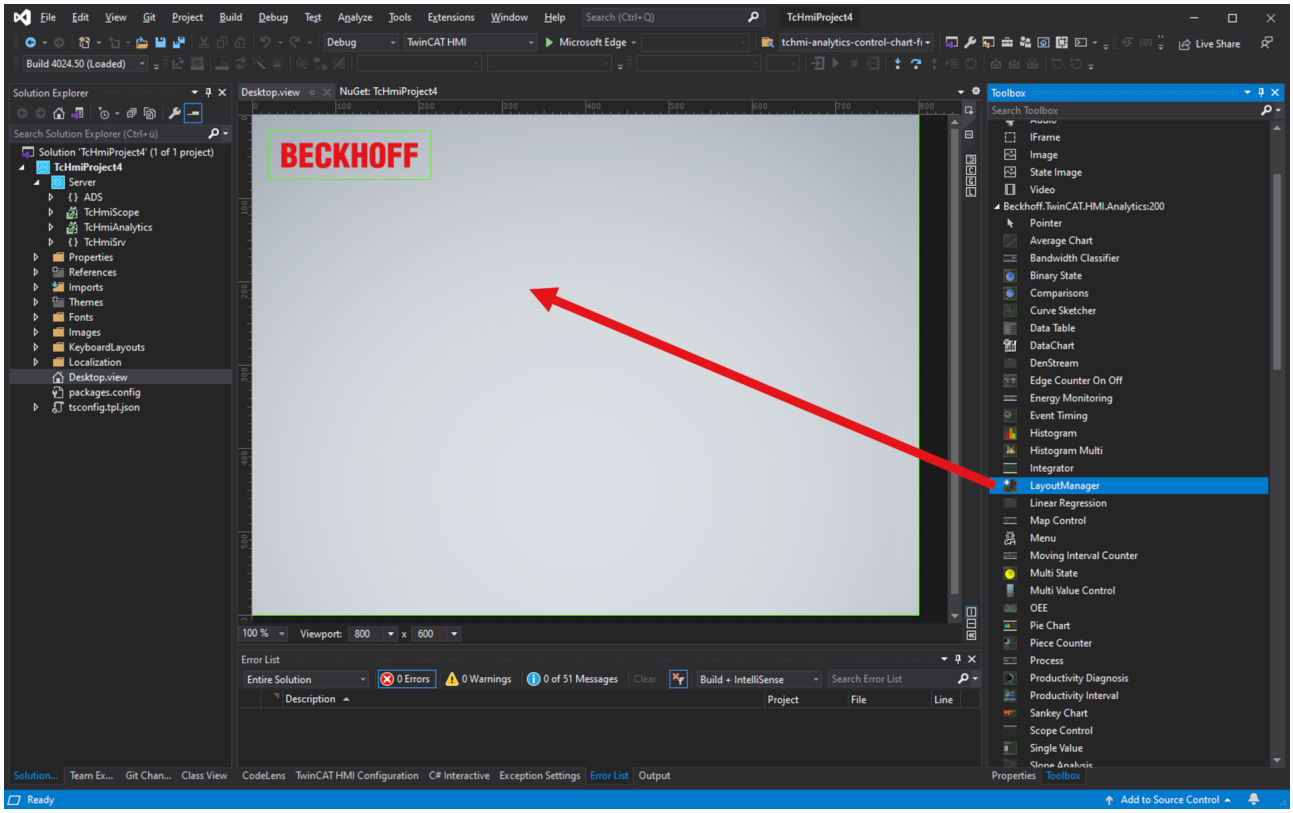
NuGetパッケージのインストール

必要なコンポーネント（Analytics Framework ControlsとAnalytics Server Extension）は、NuGetパッケージを使用してHMIプロジェクトに統合できます。そのためには、NuGet Package Managerでパッケージ「Beckhoff.TwinCAT.HMI.Analytics.ControlsExtension」と「Beckhoff.TwinCAT.HMI.Analytics.Controls」をインストールする必要があります。

インストール後、「TcHmiAnalytics 」サーバエクステンションは、HMIプロジェクト内のサーバノードの下に配置されます。自動的に起動しない場合は、エクステンションを右クリックして手動で起動する必要があります。

Layout Managerコントロールの追加

動的なレイアウトやユーザー専用のレイアウトは、Layout Managerコントロールを使用して作成できます。Layout Managerコントロールは、ツールボックスからドラッグ＆ドロップでHMIプロジェクトに挿入できます。「Beckhoff.TwinCAT.HMI.Analytics」エリアにあります。Layout Managerは、Containerコントロールに似ています。違いは、Layout Managerは、ブラウザ上でコントロールの位置を変更できる点です。



Layout Managerコントロールのプロパティ設定

Layout Managerコントロールを適切に機能させるには、以下のプロパティを設定する必要があります：

名前	説明
Identifier	サーバエクステンションを使ってレイアウトを保存した後は、識別子を変更してはいけません。変更すると、保存したレイアウトは割り当てられなくなります。
WidthMode	選択した「Value」は変更してはいけません。

名前	説明
HeightMode	選択した「Value」は変更してはいけません。

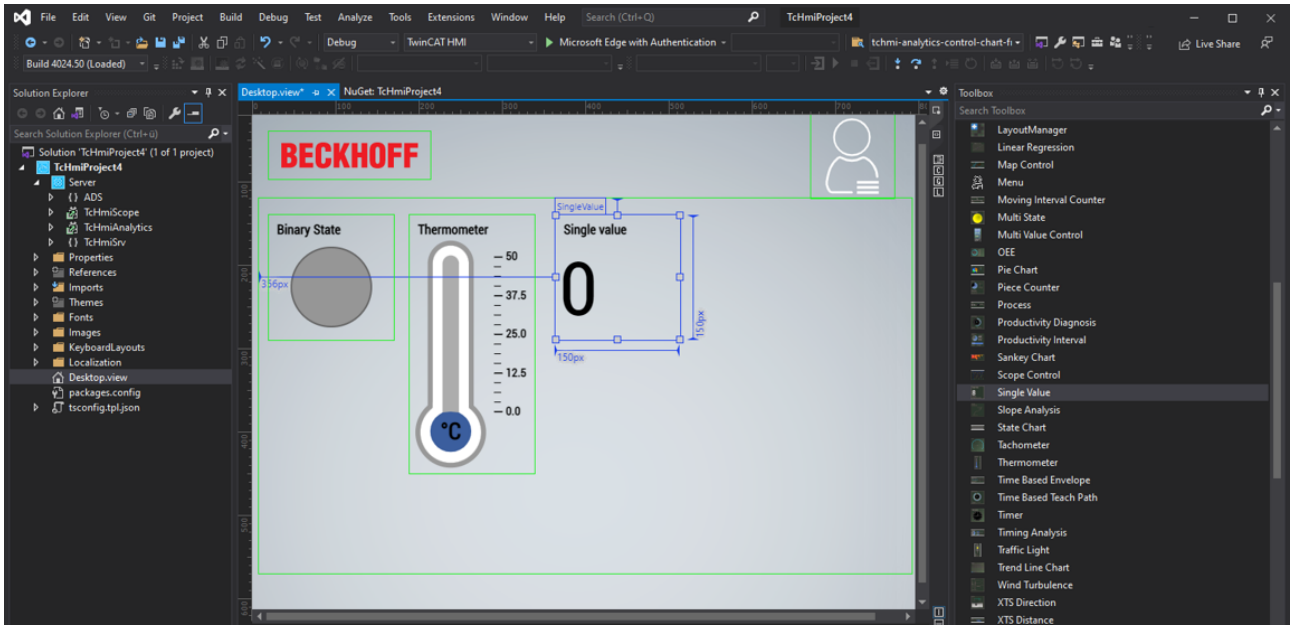
オプションで、以下のパラメータをセットできます。

名前	説明
DarkMode	コントロールをダークモードで表示するかどうかを選択します。
MobileViewWidth	モバイルビューに切り替えるためのピクセル幅。
Number of columns	通常のビューのレイアウトに含む列の数。
Grid Gap	Layout Manager 上のコントロール間の相対的な距離（%単位）。
UseCustomRowHeight	RowHeight パラメータを使用して設定できる固定行の高さを有効にするために使用します。これを有効にしない場合、自動的にスケーリングされた高さになります。
RowHeight	行の高さをピクセル単位で指定できます。
LockControls	このパラメータは、編集モード時にレイアウト内の既存コントロールとのインタラクションを防止するために使用します。
IsAnalyticsUse	このプロパティは、Layout Managerコントロールを Analytics-One-Click-Dashboardで使用している場合にのみ有効にしてください。これにより、Layout Managerコントロールのメニューがレイアウト上に適切に配置されます。
MenuControlSymbol	「Menu」タイプのコントロール（Beckhoff.TwinCAT.HMI.Analyticsエリアのコントロール）も、オプションでリンクできます。これは、常にダッシュボードに表示され、「Desktop.view」に配置されている必要があります。編集モードはMenuコントロールからも開閉できます。

コントロールの追加

その後、任意のコントロールをLayout Managerコントロールに追加できます。これらは、HMIエンジニアリングに応じて移動できます。

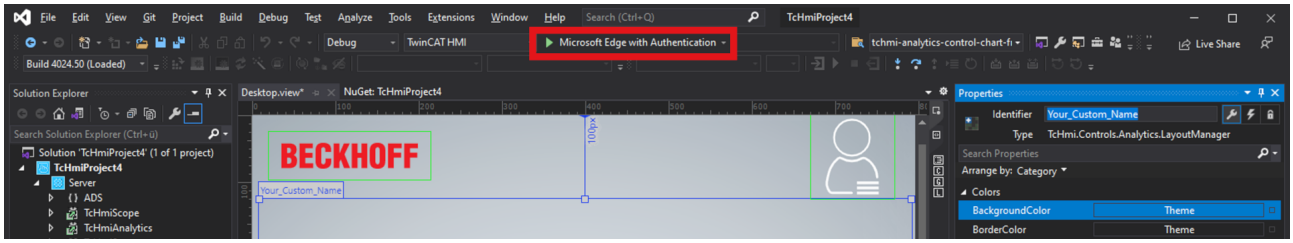
インタラクティブ機能を使ってユーザ用にレイアウトが保存されている場合、アイテムは保存されたレイアウトからロードされます。後で、他のコントロールが追加された場合は、レイアウト内の空きスペースにソートされます。一方、削除されたコントロールは、レイアウトから消えます。



インタラクティブなチャートコントロールもウェブブラウザに追加できます。

プロジェクトのビルド

インタラクティブダッシュボード機能およびインタラクティブチャートの機能を使用するには、ユーザ認証をして、プロジェクトを構築する必要があります。



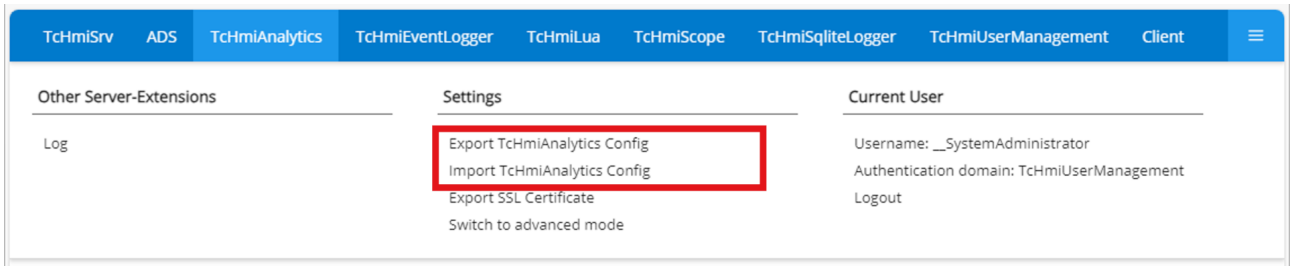
ユーザ独自の変更

インタラクティブ機能は、ウェブブラウザで使用できます。インタラクティブダッシュボード [▶ 468] とインタラクティブチャート [▶ 474]の使用に関する情報は、リンクページを参照ください。

レイアウトとチャート設定のエクスポートとインポート

「TcHmiAnalytics」サーバエクステンションに保存されたレイアウトやチャートの設定は、エクスポートやインポートが可能です。これは、設定をリモートプロジェクト（HMIサーバなどで実行）からローカルプロジェクトに転送する場合に便利です。あるいは、リモートプロジェクトの設定を更新することもできます。

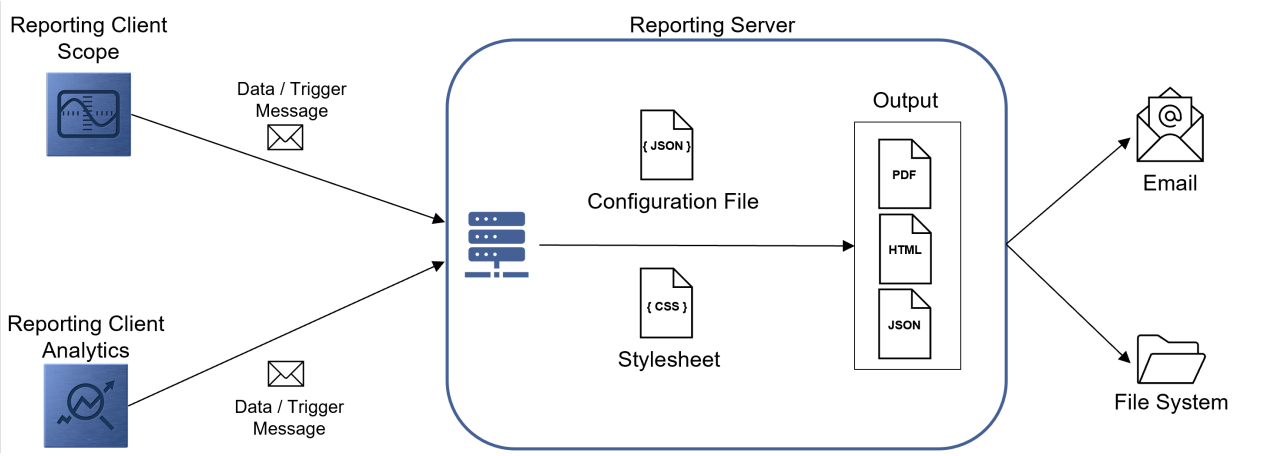
設定は、各 HMI プロジェクトのConfig ページからサーバエクステンションにエクスポートおよびインポートできます。



6.10 レポートの統合

6.10.1 基本的なコンセプト

次の図は、Reporting Clientsから、生成されたレポートの配布まで、TwinCAT Reportingの基本概念を示しています。



Reporting Clients

現在、Reporting Clientsは、TwinCAT Scope View、TwinCAT Analytics Workbench、またはTwinCAT Analytics Service Toolに統合されています。これらのツールは、常時レポーティングやオンデマンドレポーティングのための機能を提供します。

-
- 解析レポート [▶ 512]

PLC経由での常時レポーティングも可能です。ファンクションブロックは、TwinCAT Analyticsのアルゴリズムと同じ機能を備えています。

- PLC

Reporting Clientsは、TwinCAT Reporting Serverと通信します。Reporting Clients は2種類のメッセージを送信できます。1つ目に、データメッセージを関連データとともに送信できます。もう一方で、いわゆるトリガーメッセージを通信することもできます。このタイプのメッセージは、レポート作成のトリガーとなります。

TwinCAT Reporting Server

TwinCAT Reporting Server [▶ 520] には様々なタスクがあります。サーバはReporting Clientsからメッセージを受信し、データをDataStoreに格納します。レポートは、関連する設定ファイルで定義できます。レポートへのデータオブジェクトの割り当ては、レポート名とデータキーによって確立されます。レポートのデザインはスタイルシートでカスタマイズできます。トリガーメッセージが受信されるとすぐに、関連データを含むレポートが設定された形式で作成され、配信されます。

用語集

Term	説明
レポート	レポートは、PDF、HTML、JSON形式で利用可能なドキュメントです。情報と画像が含まれています。
設定ファイル	レポートの構造を記述します。また、デフォルトで固定情報をレポートに統合することもできます。さらに、Reporting Serverで対応するレポートの設定が可能です。
Reporting Server	設定ファイルを管理し、情報を受信し、この情報から個々のレポートを作成します。
Reporting Client	データをRepor（ting Serverに送信したり、レポートの作成をトリガーしたりするために使用します。
データメッセージ	Reporting Clientはデータメッセージを介して Reporting Serverに情報を送信します。含まれるデータはReporting Serverにキャッシュされます。
トリガーメッセージ	Reporting ClientはReporting Serverにトリガーメッセージを送信し、レポート生成をトリガーします。このため、Reporting Serverは、この時点までに受信したレポートに属する全てのデータを使用します。生成後、データは削除されます。

レポート名	レポートを識別するために使用されます。レポート名は、設定ファイル名から拡張子を除いたものです（設定ファイル名：Beckhoff Report Template.json / レポート名：Beckhoff Report Template)。
Data Key	レポートのデータオブジェクトを識別します。レポートとその設定ファイル内で一意である必要があります。これにより、レポート内のデータオブジェクトの割り当てとソーティングを確実に行うことができます。
常時レポート	データを収集し、自動的にレポートを作成する機能を提供します。
オンデマンドレポート	オンデマンドでカスタムレポートを作成する機能を提供します。

6.10.2 解析レポート

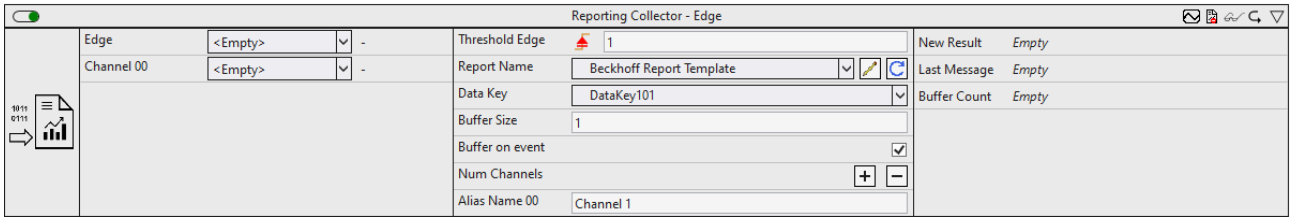
6.10.2.1 常時レポート

Reporting カテゴリのアルゴリズムを使用して、TwinCAT Analyticsに24時間365日のレポート機能を実装できます。Reporting Collector（レポートコレクタ）はデータを収集し、Reporting Server（レポートサーバ）に送信します。レポートトリガーはレポートの作成をトリガーします。

6.10.2.1.1 Reporting Collector（レポートコレクタ）

Reporting Collector（レポートコレクタ）はデータを収集し、イベント後にデータメッセージとして Reporting Server（レポートサーバ）に送信します。

6.10.2.1.1.1 Reporting Collector Edge（レポートコレクタエッジ）



Reporting Collector Edge（レポートコレクタエッジ）は入力チャンネルからデータを収集し、設定に応じて、イベント後またはバッファが満たされた後にReporting Server（レポートサーバ）にデータ送信します。入力チャンネルの信号が、設定したエッジで閾値を超過すると、イベントがトリガーされます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

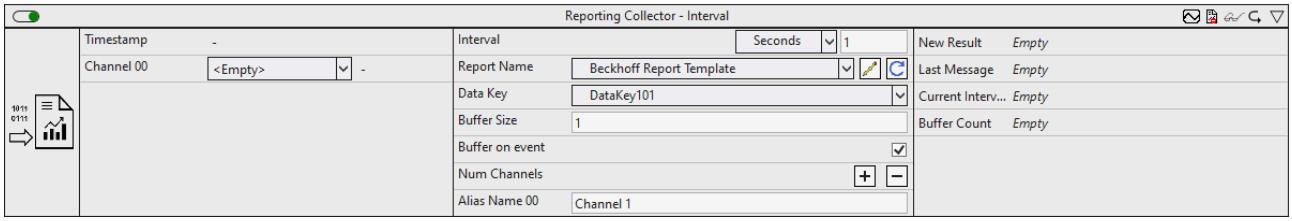
- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Tolerance（オプション）：**Equal / NotEqual 比較時に使用される許容値。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。
- **Data Key:**データキーはレポート内で一意である必要があります。データキーにより、データオブジェクトをレポートに対して一意に割り当てることが可能です。
- **Buffer Size :** バッファリングされるデータのサイズを指定します。バッファサイズが1の場合、キーと値の構造体が構築されます。バッファサイズが1より大きい場合、データは表で示されます。

- **Buffer on event :** データをどのように収集し、バッファリングするかを指定します。
  パラメータがTrueの場合、入力データは入力のエッジ信号ごとにバッファリングされます。Buffer Count 出力が Buffer Size パラメータと同じ値になるとすぐに、データは TcReportingServer に送信されます。
  パラメータが False の場合、入力のデータはサイクルごとにバッファリングされます。バッファサイズに達すると、新しいデータが前のデータを置き換えます。入力にエッジ信号がある場合に、データはTcReportingServerに送信されます。
- **Num Channels :** データを収集する入力チャンネルの数。また、入力チャンネルに説明（ラベル付け）を可能にするパラメータ数を示します。
- **Alias Name:**入力チャンネルの識別のための説明（ラベル名）です。BufferSize = 1 : エイリアス名は、キーと値の構造体でキーとして機能します。BufferSize > 1: エイリアス名は、テーブル列の見出しとなります。
- **Include Timestamps (オプション):** タイムスタンプを含む列を挿入します。

出力値

- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Buffer Count :** バッファ内の要素数を指定します。

6.10.2.1.1.2 Reporting Collector Interval



Reporting Collector Intervalは入力チャンネルからデータを収集し、設定に応じて、イベント後またはバッファが満たされた後にReporting Server（レポートサーバ）にデータ送信します。設定した時間区間が経過すると、イベントがトリガーされます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

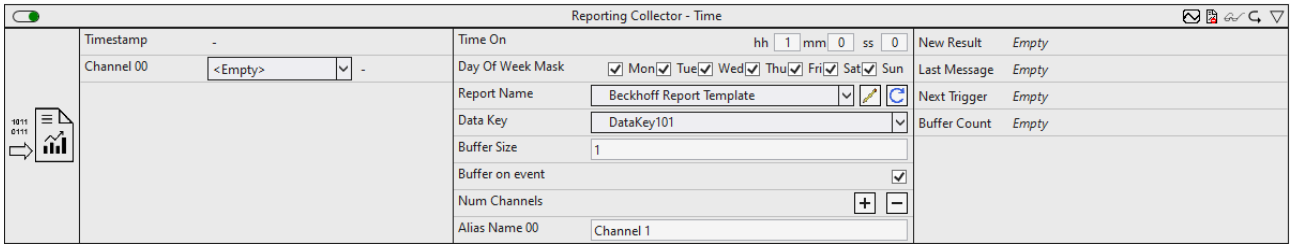
- **Interval :** データをReporting Serverに送信する時間区間。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。
- **Data Key:**データキーはレポート内で一意である必要があります。データキーにより、データオブジェクトをレポートに対して一意に割り当てることが可能です。
- **Buffer Size :** バッファリングされるデータのサイズを指定します。バッファサイズが1の場合、キーと値の構造体が構築されます。バッファサイズが1より大きい場合、データは表で示されます。
- **Buffer on event :** データをどのように収集し、バッファリングするかを指定します。
  パラメータがTrueの場合、入力データは入力のエッジ信号ごとにバッファリングされます。Buffer Count 出力が Buffer Size パラメータと同じ値になるとすぐに、データは TcReportingServer に送信されます。
  パラメータが False の場合、入力のデータはサイクルごとにバッファリングされます。バッファサイズに達すると、新しいデータが前のデータを置き換えます。入力にエッジ信号がある場合に、データはTcReportingServerに送信されます。
- **Num Channels :** データを収集する入力チャンネルの数。また、入力チャンネルに説明（ラベル付け）を可能にするパラメータ数を示します。

- **Alias Name:**入力チャンネルの識別のための説明（ラベル名）です。BufferSize = 1：エイリアス名は、キーと値の構造体でキーとして機能します。BufferSize > 1: エイリアス名は、テーブル列の見出しとなります。
- **Include Timestamps (オプション):** タイムスタンプを含む列を挿入します。

出力値

- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Current Interval Time:**現在の区間の経過時間を示します。
- **Buffer Count :** バッファ内の要素数を指定します。

6.10.2.1.1.3 Reporting Collector Time



Reporting Collector Timeは入力チャンネルからデータを収集し、設定に応じて、イベント後またはバッファが満たされた後にデータをReporting Server（レポートサーバ）に送信します。設定されたスイッチオン時間に達するとイベントがトリガーされます。スイッチオンの時間と曜日を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Time On :** スイッチオン時間。
- **Day of Week Mask:**タイマーを有効にする曜日。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。
- **Data Key:**データキーはレポート内で一意である必要があります。データキーにより、データオブジェクトをレポートに対して一意に割り当てることが可能です。
- **Buffer Size :** バッファリングされるデータのサイズを指定します。バッファサイズが1の場合、キーと値の構造体が構築されます。バッファサイズが1より大きい場合、データは表で示されます。
- **Buffer on event :** データをどのように収集し、バッファリングするかを指定します。
  パラメータがTrueの場合、入力データは入力のエッジ信号ごとにバッファリングされます。Buffer Count 出力が Buffer Size パラメータと同じ値になるとすぐに、データは TcReportingServer に送信されます。
  パラメータが False の場合、入力のデータはサイクルごとにバッファリングされます。バッファサイズに達すると、新しいデータが前のデータを置き換えます。入力にエッジ信号がある場合に、データはTcReportingServerに送信されます。
- **Num Channels :** データを収集する入力チャンネルの数。また、入力チャンネルに説明（ラベル付け）を可能にするパラメータ数を示します。
- **Alias Name:**入力チャンネルの識別のための説明（ラベル名）です。BufferSize = 1：エイリアス名は、キーと値の構造体でキーとして機能します。BufferSize > 1: エイリアス名は、テーブル列の見出しとなります。
- **Include Timestamps (オプション):** タイムスタンプを含む列を挿入します。

出力値

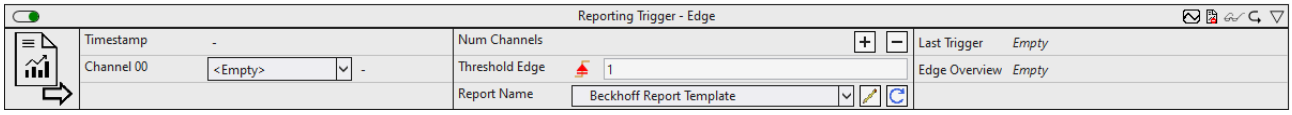
- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。

- **Next Message:**次のメッセージまでの残り時間を示します。
- **Buffer Count :** バッファ内の要素数を指定します。

6.10.2.1.2 Reporting Trigger

レポーティング・トリガーはイベント発生後にReporting Serverにトリガー・メッセージを送信し、レポートの作成をトリガーします。

6.10.2.1.2.1 Reporting Trigger Edge



Reporting Trigger Edgeは、イベントのトリガー後に、レポートの作成をトリガーします。入力チャンネル信号が指定された閾値で設定されたエッジを超えるとイベントがトリガーされます。内部的には、一度Trueになった入力は、Trueのままになります。入力は、全ての入力が少なくとも一度はTrueになった場合のみ、入力はFalseにリセットされます。これにより、出力 bNewResult を複数の Reporting Collector が共通の入力として使用できるようになり、すべての Reporting Collector がデータメッセージを送信した時点で、トリガーメッセージが送信されます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

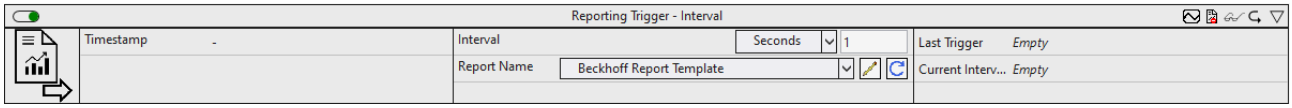
設定オプション

- **Type of the edge:**エッジカウンタが立ち上がりエッジと立ち下がりエッジのどちらに反応するかを指定します。
- **Threshold:**それぞれのエッジにおける信号の閾値。信号がこの閾値を超過したときイベントがトリガーされます。
- **Tolerance（オプション）:** Equal / NotEqual 比較時に使用される許容値。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。

出力値

- **Last Trigger:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Edge Overview:**どの入力チャンネルが少なくとも一度はTrueであったかを示します。

6.10.2.1.2.2 Reporting Trigger Interval



Reporting Trigger Intervalは、イベントのトリガー後に、レポートの作成をトリガーします。設定した時間区間が経過すると、イベントがトリガーされます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

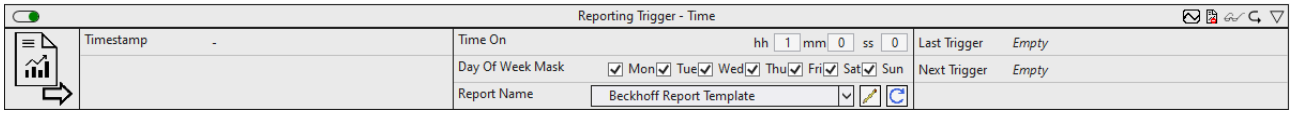
設定オプション

- **Interval :** データをReporting Serverに送信する時間区間。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。

出力値

- **Last Trigger:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Current Interval Time:**現在の区間の経過時間を示します。

6.10.2.1.2.3 Reporting Trigger Time



Reporting Trigger Timeは、イベントのトリガー後に、レポートの作成をトリガーします。設定されたスイッチオン時間に達するとイベントがトリガーされます。スイッチオンの時間と曜日を設定できます。

オプションで、*Enable Execution*入力にブール信号を選択し、信号の値が*TRUE*である場合にのみアルゴリズムが有効になるようにすることもできます。

設定オプション

- **Time On :** スイッチオン時間。
- **Day of Week Mask:**タイマーを有効にする曜日。
- **Report Name:** レポートの名前を指定します。この名前はReporting Serverの設定ファイル名と一致させる必要があります。ドロップダウンメニューには、Reporting ServerのConfigurationフォルダに格納されている全てのレポート名が表示されます。Editボタンは、選択した設定ファイルを編集するために使用できます。

出力値

- **Last Message:**最後のメッセージがいつReporting Serverに送信されたかを示します。
- **Next Trigger:**次のメッセージまでの残り時間を示します。

6.10.2.2 オンデマンドレポート

オンデマンドレポートは、アルゴリズムデータと追加データを手作業でレポートに統合する機能を提供します。

各アルゴリズムの右上には5つのアイコンがあります：左から2番目のアイコンは**Include to Report**アイコンと呼びます。このアイコンをクリックすると、マニュアルレポートにアルゴリズムを追加できます。

このアイコンは、アルゴリズムがマニュアルレポートに追加されていることを示しています：

このアイコンは、そのアルゴリズムがマニュアルレポートに含まれていないことを示しています：

ネットワークにはこれらのアイコンも含まれます。これにより、全てのネットワークとアルゴリズムをマニュアルレポートに追加したり、対象外としたりできます。

Include to Reportアイコンの機能は、プロパティでマッピングすることもできます。

ネットワークの場合、プロパティは以下の通りです：

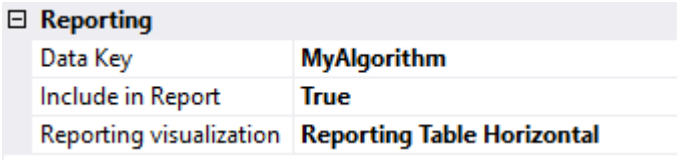
Reporting	
Data Key	**MyNetwork**
Include in Report	**False**
Include SubMember in Report	**True**
Reporting visualization	**Reporting Table Default**

クローズドネットワークは、オンデマンドレポートのアルゴリズムのように取り扱われます。

レポート

Data Key	Data Keyはレポート内のデータオブジェクトを一意に識別します。Data Keyにはデフォルトでネットワーク名がプリセットされています。
Include in Report	このオプションは、ネットワークをマニュアルレポートに追加するかどうかを指定します。表示する内容は、ネットワークの入出力またはネットワークパラメータです。
Include SubMember in Report（レポートにサブメンバーを含める）	このオプションは、ネットワークに含まれるアナリティクス要素をレポートに統合するかどうかを指定します。
レポートの可視化	このオプションは、ネットワークのデータをどのように表示するかを指定します。

アルゴリズムの場合、プロパティは以下のようになります：

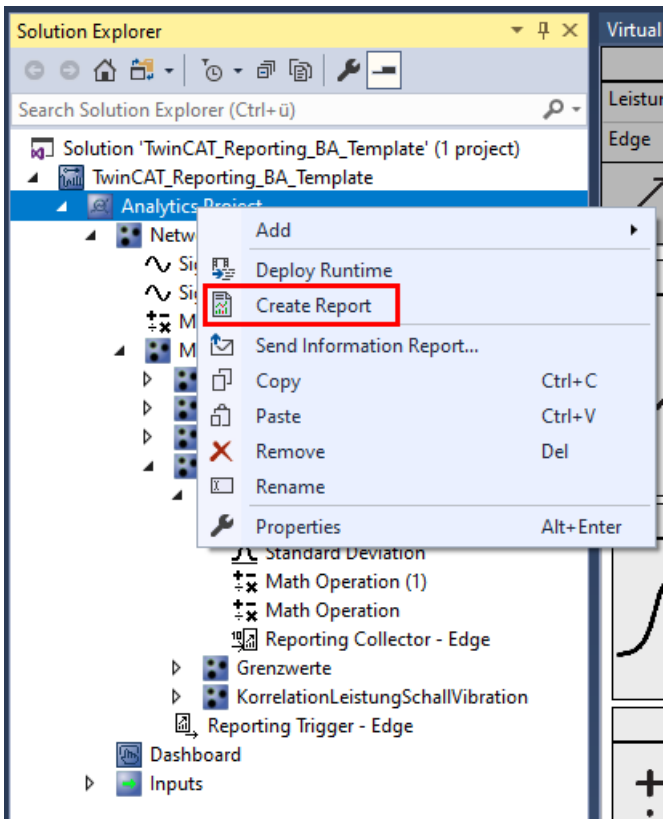


レポート

Data Key	Data Keyはレポート内のデータオブジェクトを一意に識別します。Data Keyには、デフォルトでネットワーク名とアルゴリズム名がプリセットされています。
Include in Report	このオプションは、アルゴリズムをマニュアル・レポートに追加するかどうかを指定します。
Reporting visualization	このオプションは、アルゴリズムデータの表示方法を指定します。

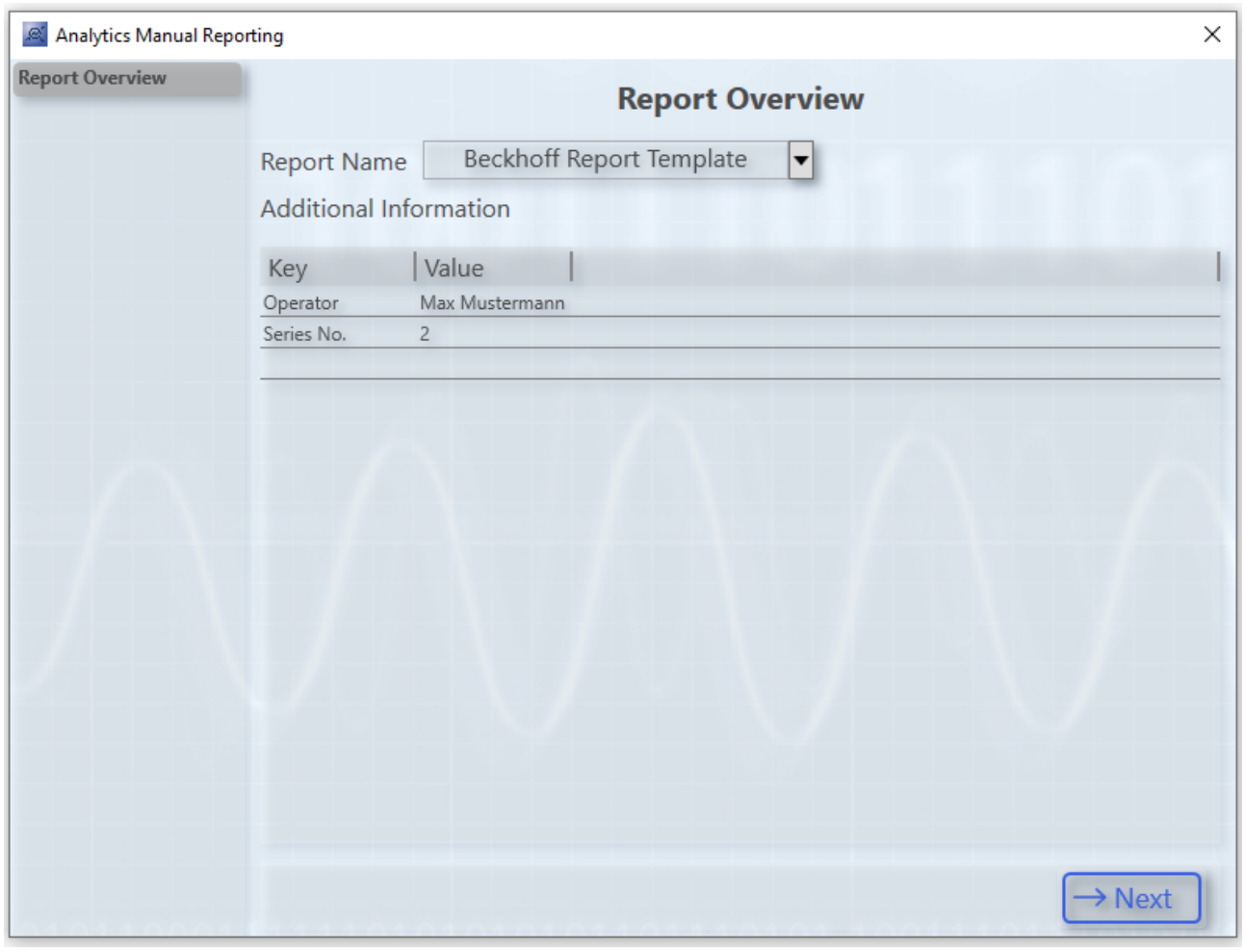
オンデマンドレポートの開始方法は2つあります。1つは、ツールバーのReporting ボタンからレポートを開く方法です：

もう1つは、Analyticsプロジェクトのコンテキストメニューから起動できます：



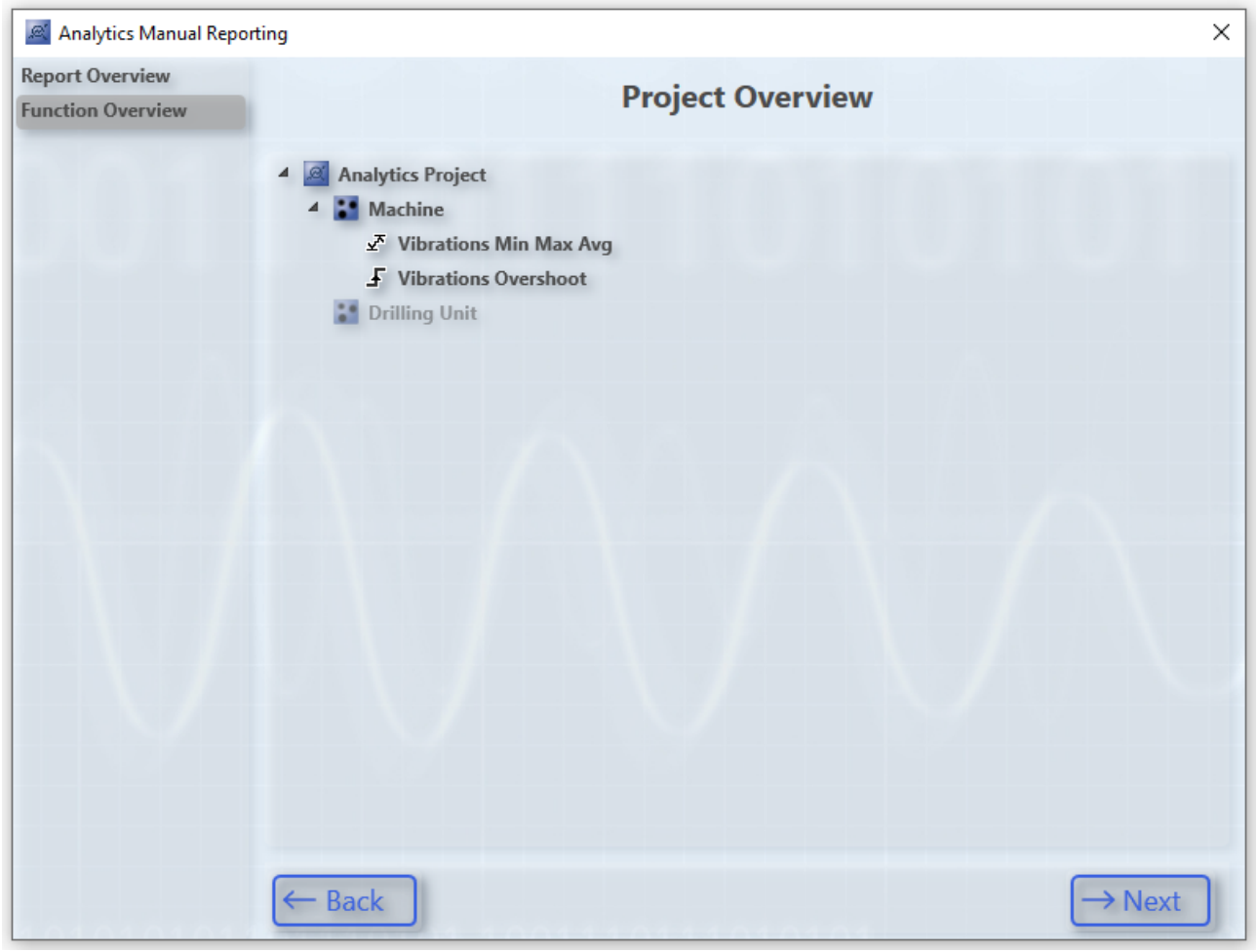
オンデマンドレポートのウィザードが開きます。

最初のページでは、レポートに詳細情報を追加できます。

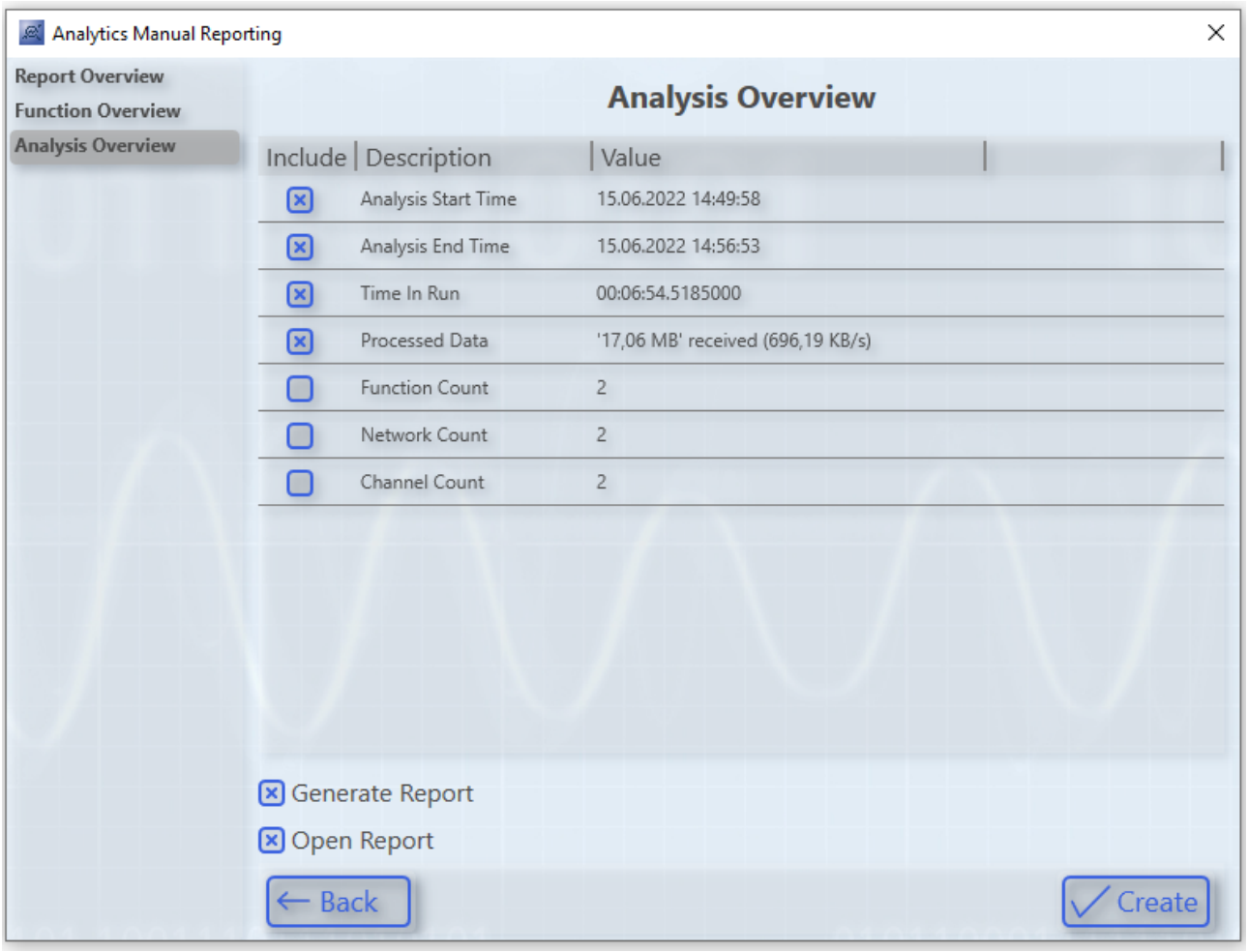


追加情報はキーバリュー構造で構築されます。空白の行をクリックすると、別の行を追加できます。選択した行は、Deleteキーを押すことで削除できます。

プロジェクトの概要は2ページ目に掲載されています。グレーで表示されているアルゴリズムとネットワークはレポートに含まれていません。



3ページ目では、解析に関する情報をレポートに追加できます。



この表では、多くの情報をレポートに統合できます。Include列のチェックボックスを使用して、個々の情報を選択または選択解除できます。

Generate Reportオプションは、設定したデータでレポートを生成するかどうかを定義します。このオプションが選択されていない場合、データのみがReporting Serverに送信されますが、レポートは生成されません。このオプションは、複数製品のデータを1つのレポートに手動で追加する場合に便利です。

Open Reportオプションは、生成後にレポートを開くかどうかを定義します。このオプションを使用するには、PDFまたはHTML形式のFilePublishLocationを設定ファイルで定義する必要があります。レポートがPDFとHTML形式で作成された場合、デフォルトでPDFレポートが開きます。

6.10.3 TwinCAT Reporting Server

6.10.3.1 設計

6.10.3.1.1 設定ファイル

設定ファイルでは、レポートの整理、並べ替え、カスタマイズが可能です。設定ファイルは、フォルダ *¥TwinCAT¥Functions¥Reporting-Server¥Configuration¥ConfigFiles*に保存する必要があります。

Reporting Serverでは以下のオプションがサポートされています。

キー	説明	サンプル	レベル
全般			

キー	説明	サンプル	レベル
Store Type	保存モードを指定します。現在は「RAM」のみがサポートされています。	`"StoreType": "Ram"`	1
Header	ヘッダーを上下に並べて3つ表示できます。2番目、3番目のヘッダーを使用しない場合は、空文字列を指定できます。	`"Header": [ "Header 1", "Header 2", "Header 3", ]`	1
Footer	PDFページのフッターは3つのエリアに分かれています。3つのエリアは全て、自由記述でカスタマイズできます。作成日時を表示するには、プレースホルダ{date} {time}を指定する必要があります。ページ番号を表示するには、プレースホルダ{page} of {total-pages}を指定する必要があります。空白にする場合は、空文字列を指定できます。	`"Footer": [ "Footer Left", "Footer Center", "Footer Right", ]`	1
Logo	ロゴは最初のページの右上に表示されます。ロゴは、*Reporting-Server¥Configuration¥ConfigFiles*フォルダへの相対パスとして指定できます。あるいは、ロゴはBase64Stringとして、直前のエンコーディングで指定することもできます。	`"Logo": ".//<Bildname.png>"` oder `"Logo": "data:image/ <Encoding>base64, <Base64String>"`	1
StyleSheet	レポートのデザインをカスタマイズできます。ユーザ固有のスタイルシートを使用するには、名前を指定する必要があります（スタイル・シート [▶ 524]を参照）。	`"StyleSheet": "StyleBeckhoff.css"`	1
SignatureSettingsFile	レポートに署名できます（署名 [▶ 527]を参照）。	`"SignatureSettingsFile": "BeckhoffSignatures.json",`	
Email text	特定のテキストが指定されない限り、全てのEメールにこのテキストが含まれます。	`"EmailText": "Custom of E-Mails send to all recipients with no or empty E-Mail-Text. Special. Ones"`	1
Required Keys	Data Keyに基づくコレクションです。これらのData Keyのデータオブジェクトが Reporting Serverで利用できない場合、レポートは生成されません。	`"RequiredKeys": ["Data101"]`	1
Trigger On Required Key Received	レポート作成をトリガーするもう1つの方法。trueが選択され、必要なData Keyを持つ全てのデータオブジェクトが存在する場合、レポートが生成されます。これにより、Reporting Clientからのトリガーメッセージが不要になります。	`"TriggerOnRequiredKeyReceived": false`	1
固定のデフォルトデータ			
Default Data	このオブジェクトを使用して、各レポートに表示するデータオブジェクトを定義できます。	`"DefaultData": {}`	1
Text Fields	この配列には、以下の要素を持つ複数のテキストフィールドオブジェクトを含めることができます。	`"TextFields": []`	2
Key	テキストフィールドの見出しとして使用されます。	`"Key": "My Text Field"`	3
Comment	データオブジェクトに対するコメント。コメントはレポートに表示されません。	`"Comment": "My Comment"`	3
Sort Priority	レポート内のデータオブジェクトの位置を指定します。	`"SortPriority": 42`	3
Value	個々の値を含み、STRINGとして指定する必要があります。	`"Value": "253-5553-9421"`	3

キー	説明	サンプル	レベル
Tables	この配列には、以下の要素を持つ複数のテーブルオブジェクトを含めることができます。	`"Tables": []`	2
Key	表の見出しとして使用されます。	`"Key": "My Table"`	3
Comment	データオブジェクトに対するコメント。コメントはレポートに表示されません。	`"Comment": "My Comment"`	3
Sort Priority	レポート内のデータオブジェクトの位置を指定します。	`"SortPriority": 42`	3
Value	このオブジェクトは、以下の要素を含みます。	`"Value": {}`	3
Header	この配列は、テーブルの列タイトルを格納します。列タイトルの数は、テーブルの列数と同じである必要があります。	`"Header": ["Table Header 1", "Table Header 2"]`	4
DataTypes	この配列は、テーブルの列のデータ型を格納します。以下のデータ型が使用できます：BOOL, BYTE, SBYTE, CHAR, DECIMAL, DOUBLE, FLOAT, INT, UINT, LONG, ULONG, SHORT, USHORT, STRING. データ型の数はテーブルの列数と同じである必要があります。	`"DataTypes": ["string", "double"]`	4
Values	この2次元配列は、テーブルの値を格納します。配列の配列は、テーブルの列を表します。テーブルの列は指定されたデータ型と一致する必要があります。配列の数はテーブルの列数と同じである必要があります。配列は同じサイズである必要があります。	`"Values": [[],[]]`	4
Images	この配列は、以下の要素を持つ複数の画像オブジェクトを含めることができます。	`"Images": []`	2
Key	画像のキャプションとして使用。	`"Key": "My Image"`	3
Comment	データオブジェクトに対するコメント。コメントはレポートに表示されません。	`"Comment": "My Comment"`	3
Sort Priority	レポート内のデータオブジェクトの位置を指定します。	`"SortPriority": 42`	3
Value	PNGエンコーディングのBase64文字列。	`"Value": "<Base64String PNG-Encoding>"`	3
Value Pairs	この配列は、以下の要素を持つ複数のキーと値のペアのオブジェクトを含めることができます。	`"ValuePairs": []`	2
Key	キーと値のペアに基づくコレクションの見出しとして使用されます。	`"Key": "My Value Pair"`	3
Comment	データオブジェクトに対するコメント。コメントはレポートに表示されません。	`"Comment": "My Comment"`	3
Sort Priority	レポート内のデータオブジェクトの位置を指定します。	`"SortPriority": 42`	3
Value Pair Collection	キーと値のペアのコレクションとして機能します。この配列は、以下の要素を持つ複数のオブジェクトを含めることができます。	`"ValuePair_Collection": []`	3
Key	個々の値を含み、左側に表示されます。	`"Key": "Serial number"`	4
Data Type	「Value」要素のデータ型を記述します。	`"DataType": "string"`	4
Value	個々の値を含み、右側に表示されます。値はSTRINGで指定する必要があります。	`"Value": "253-5553-9421"`	4
Data Configurations			

キー	説明	サンプル	レベル
Data Configurations	Reporting Clientsから動的に送信されるデータの並べ替えや個別化に使用します。この配列は、以下の要素を持つ複数のオブジェクトを含めることができます。	`"DataConfigurations": []`	1
Key	データオブジェクトのData Keyに対応します。これにより、このデータ設定が一意のデータオブジェクトに適用されます。（必須）	`"Key": "DataTable101"`	2
Title	見出しとして、より適切な名称を指定するオプションを提供します。（オプション）	`"Title": "Rotations and vibrations"`	2
SortPriority	レポート内のデータオブジェクトの位置を指定します。（必須）	`"SortPriority": 42`	2
IsRequired	このデータオブジェクトがレポートの生成に必要かどうかを示します。(参照：TriggerOnRequiredKey)	`"IsRequired": false`	2
StoreMode	個々のデータオブジェクトを特定の方法で保存し、それに応じて表示するオプションを提供します。現在、4種類の保存モードがサポートされています： • **Blocking**：最初のデータオブジェクトが保存され、同じデータキーを持つ後続のデータオブジェクトは全て破棄されます。 • **Overwriting**：新しく受信したデータオブジェクトは、同じデータキーを持つ最後のデータオブジェクトを置き換えます。 • **Appending**：新しく受信したデータオブジェクトを保存します。過去に受信したデータオブジェクトも保持され、レポートに表示されます。テーブルの場合、データはマージされます。 • **Appending_Series**：この保存モードは保存モード「Appending」に該当します。ただし、データオブジェクトには番号が振られ、個々の表で表示されます。	`"StoreMode": "Overwriting"`	2
RoundTo	指定した小数点以下の桁数に丸めるオプションを提供します。	`"SortPriority": 4`	2
Timestamp Format	タイムスタンプを指定された形式に変換するオプションを提供します。デフォルトでは、以下の書式が選択されています：yyyy-MM-dd HH:mm:ss.ffff（参照：Custom date and time format strings）	`"TimestampFormat": "HH:mm:ss.ffff"`	2
IsAsc	データテーブルの並べ替え順を指定するオプションを提供します。この値がtrueの場合、最初のデータセットが最初の行に表示されます。	`"IsAsc": true`	2
StyleClass	カスタムCSSスタイルクラスを使用するオプションを提供します。CSSスタイルクラスは、CSSファイルで定義し、StyleSheetオプションでファイルを参照する必要があります。	`"StyleClass": "CustomClass"`	2
Publish Locations			
Publish Locations		`"PublishLocations": []`	1
Email Publish Location	レポートをEメールとして転送するために使用します。この配列は、以下の要素を持つ複数のオブジェクトを含めることができます。	`"EmailPublishLocations": []`	2
Type	ディストリビューションのタイプを指定します。	`"Type": "File"`	3

キー	説明	サンプル	レベル
Format	選択したファイル形式のみをEメールで送信するオプションを提供します。	`"Format": ["PDF", "HTML", "JSON"]`	3
Address	複数のEメールアドレスにレポートを送信する機能を提供します。	`"Address": [ "m.mustermann@tester.com "]`	3
Use Bcc	BCCにメールアドレスを設定するオプションを提供します。	`"UseBcc": false`	3
Text	特定のEメールテキストを指定するオプションを提供します。このテキストはデフォルトのEメールテキストを置き換えます。	`"Text": "Hello, \n insert email text here.\n Your Reporting Service.`	3
File Publish Location	指定した場所にレポートをコピーするために使用します。この配列は、以下の要素を持つ複数のオブジェクトを含めることができます。	`"FilePublishLocations": []`	2
Type	ディストリビューションのタイプを指定します。	`"Type": "File"`	3
Format	選択したファイル形式のみをEメールで送信するオプションを提供します。	`"Format": ["PDF", "HTML", "JSON"]`	3
Path	レポートの保存先パスを指定できます。保存先パスは、管理者権限なしで設定できるようにする必要があります。	`"Path": "C:\\tmp\\CpyFolder"`	3

6.10.3.1.2 スタイルシート

スタイルシートを使用すると、レポートのデザインをカスタマイズできます。*StyleBeckhoff.css* のスタイルシートはテンプレートとして機能します。スタイルシートの名前は、対応する設定ファイルで指定する必要があります。スタイルシートは、*.¥TwinCAT¥Functions¥Reporting-Server¥Configuration¥ReportStyleSheets* フォルダに保存する必要があります。
基本的に、HTMLレポートに含まれる全てのオブジェクトは、カスタムのスタイルシートで個別に設定できます。
以下のCSSオブジェクトはよく使用されるもので、カスタマイズに最適です：

CSS要素	説明
html	レポート全体を参照します。
h1	レポートのメインヘッダー。
h3	レポートのサブヘッダー、およびデータオブジェクトの見出し。

以下のCSSクラスは、データオブジェクトをデザインするために使用します：

CSSクラス	説明
reporting-header	レポートのヘッダークラス。
reaporting-header img	ヘッダー右上のアイコンのクラス。
reporting-header table	ヘッダーは表形式で、このクラスでカスタマイズできます。
reporting-body	レポートコンテンツのクラス。
reporting-body caption	データオブジェクトの見出しはCaption要素の中にあります。
reporting-body valuetable	レポート内のデータテーブルのクラス。
reporting-body valuepair	レポート内のキーと値のペアからコレクションを作成するためのクラス。
reporting-body infotable	追加情報や解析、あるいは手動レポートからの情報を記録するためのクラス。

CSSクラス	説明
reporting-body figure	画像用のクラス（レポート内のスコープチャートなど）。
reporting-body img	このクラスには実際の画像が含まれます。これは図の要素にあります。

6.10.3.2 レポート形式

レポートは以下の形式で作成されます。

1. PDF
2. HTML
3. JSON

6.10.3.3 ネットワーク設定

考えられるユースケースは、Reporting ServerとReporting Clientが異なるシステムで実行される場合です。Reporting ServerとReporting Clientは同じネットワーク上になければなりません。このユースケースに対応するためには、JSONファイルを拡張する必要があります。ファイルは以下のパス以下にあり、適切な名前にする必要があります：*¥TwinCAT¥Functions¥Reporting-Server¥Configuration¥Network¥ReportingNetworkConfiguration.json*
ネットワーク構成は以下のようになります：

名前	説明	データ型
ConfigIds	対応するレポート名	List<string>
AmsNetId	ターゲット・システムのAmsNetId。Reporting Serverはターゲットシステム上で実行されます。	string

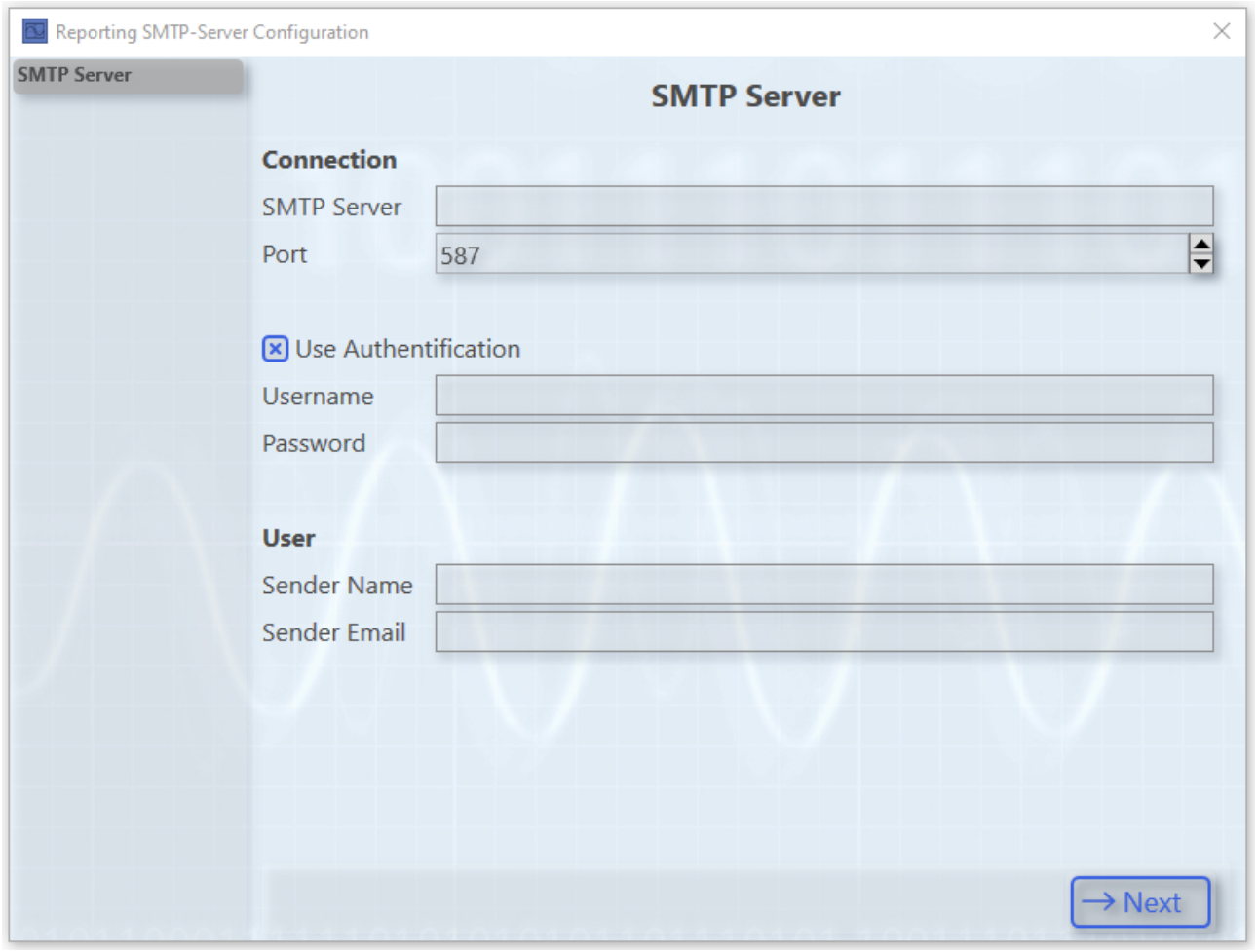
6.10.3.4 メールクライアントの設定

TwinCAT Reporting Serverは、SMTPサーバに接続できます。SMTPサーバはユーザが設定する必要があり、TwinCAT Reportingの一部ではありません。TwinCAT Reporting Serverは、設定したレポート付きメールをSMTPサーバに送信します。メールが受信者に転送されます。

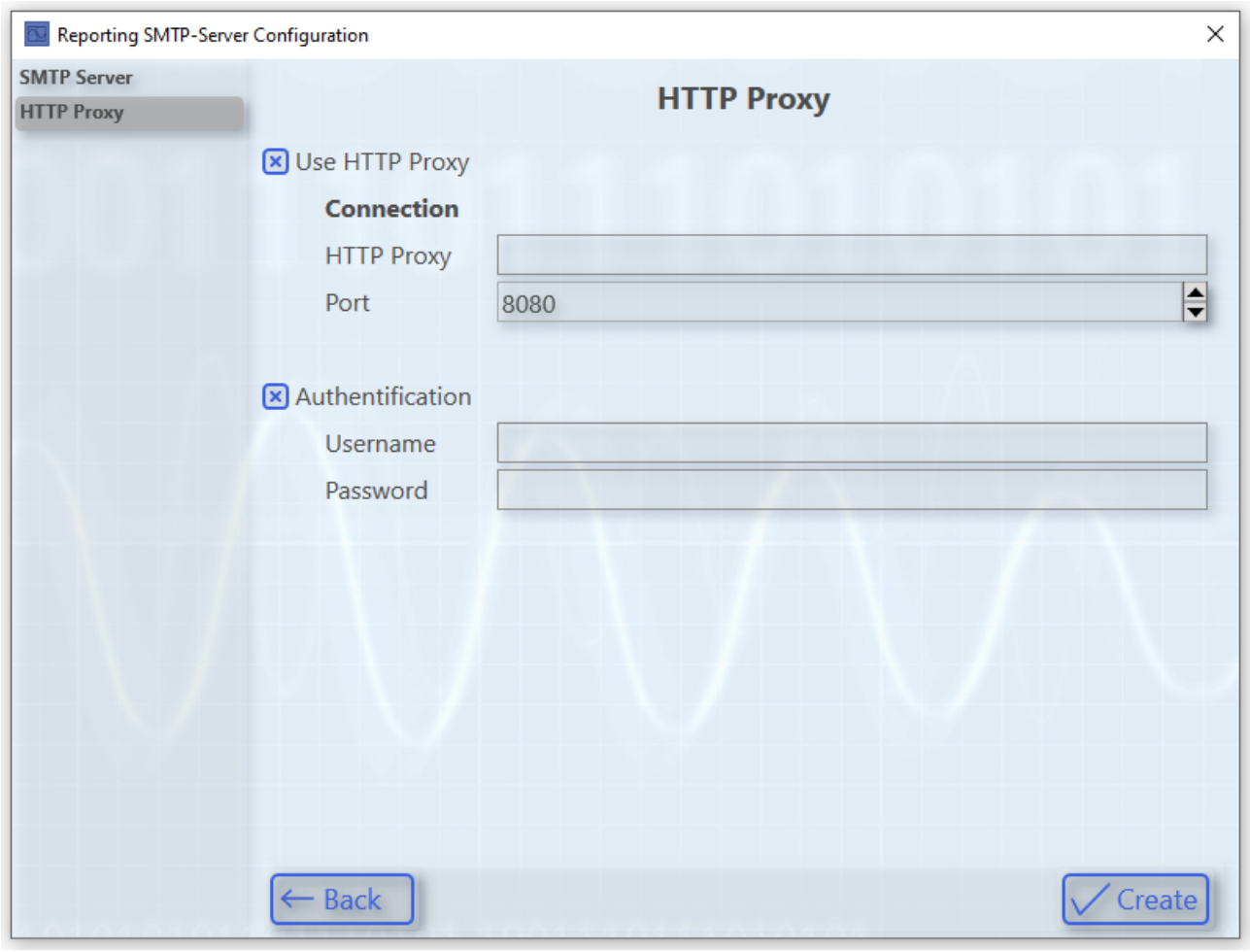
SMTPサーバ接続を設定するためのウィザードが用意されています。ウィザードは2つの方法で開くことができます。

1. **TwinCAT > Analytics > Reporting Email**
2. **Scope > Reporting Email**

最初のダイアログでは、SMPTサーバへの接続を設定できます。



2番目のダイアログでは、HTTPプロキシへの接続を設定できます。このオプションは、ユーザのアーキテクチャが必要とする場合に、有効にできます。



Createボタンをクリックすると、設定が対応するファイルに保存されます。

6.10.3.5 署名

レポートを認証するために、PDFレポートをTwinCAT Reportingで署名できます。デジタル署名と手書き署名に対応しています。署名を設定するためのウィザードは、以下の方法で開くことができます。

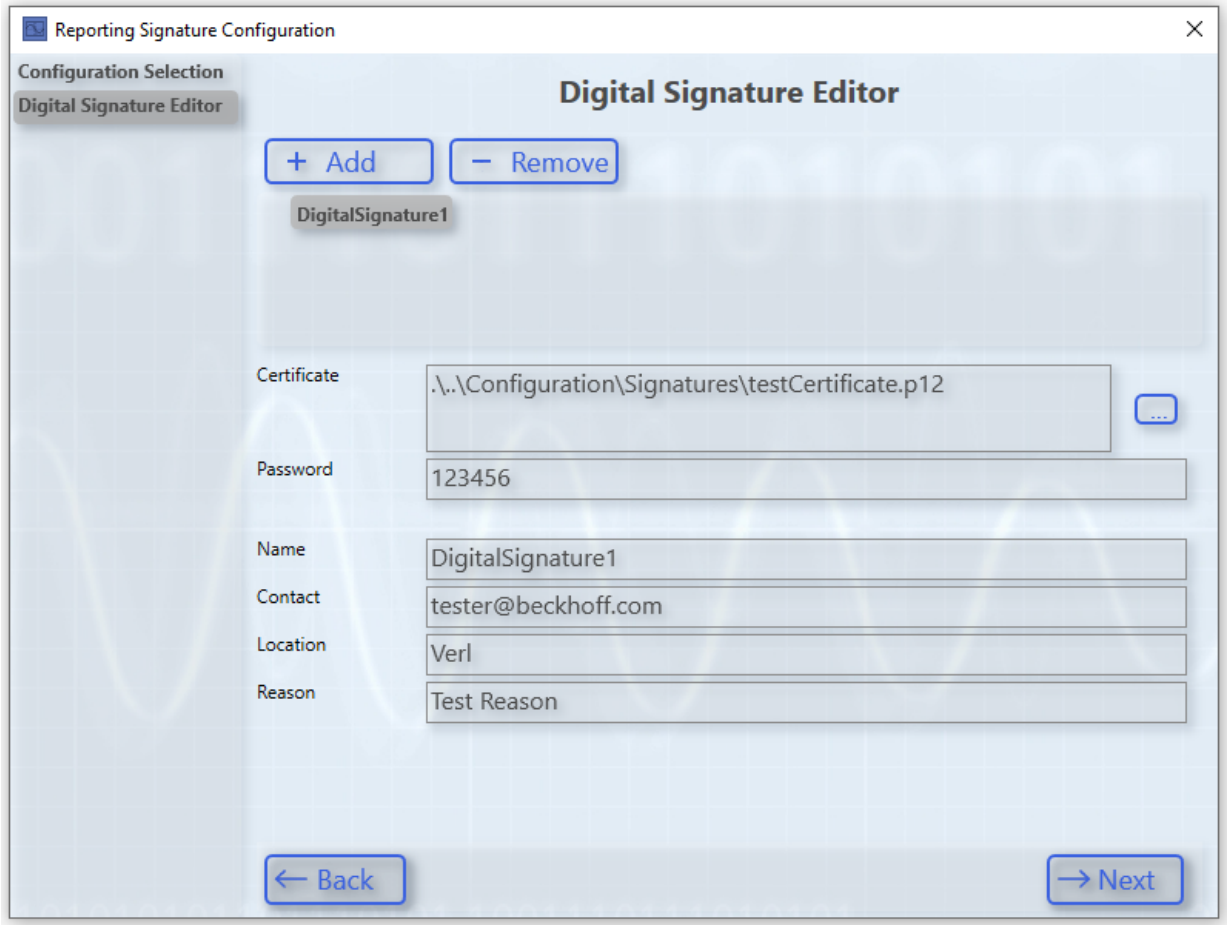
- **TwinCAT > Analytics > Reporting > Reporting Signatures**
- **Scope > Reporting > Reporting Signatures**

✓ ウィザードでは、複数の署名を設定可能で、複数のレポート設定で再利用できます。

1. この設定名は、キー「SignatureSettingsFile」を介して設定ファイル内で参照する必要があります（設定ファイル [▶ 520] 参照）。
   2つのチェックボックスで、次のページで表示される設定済み署名を使用するかどうかを選択できま

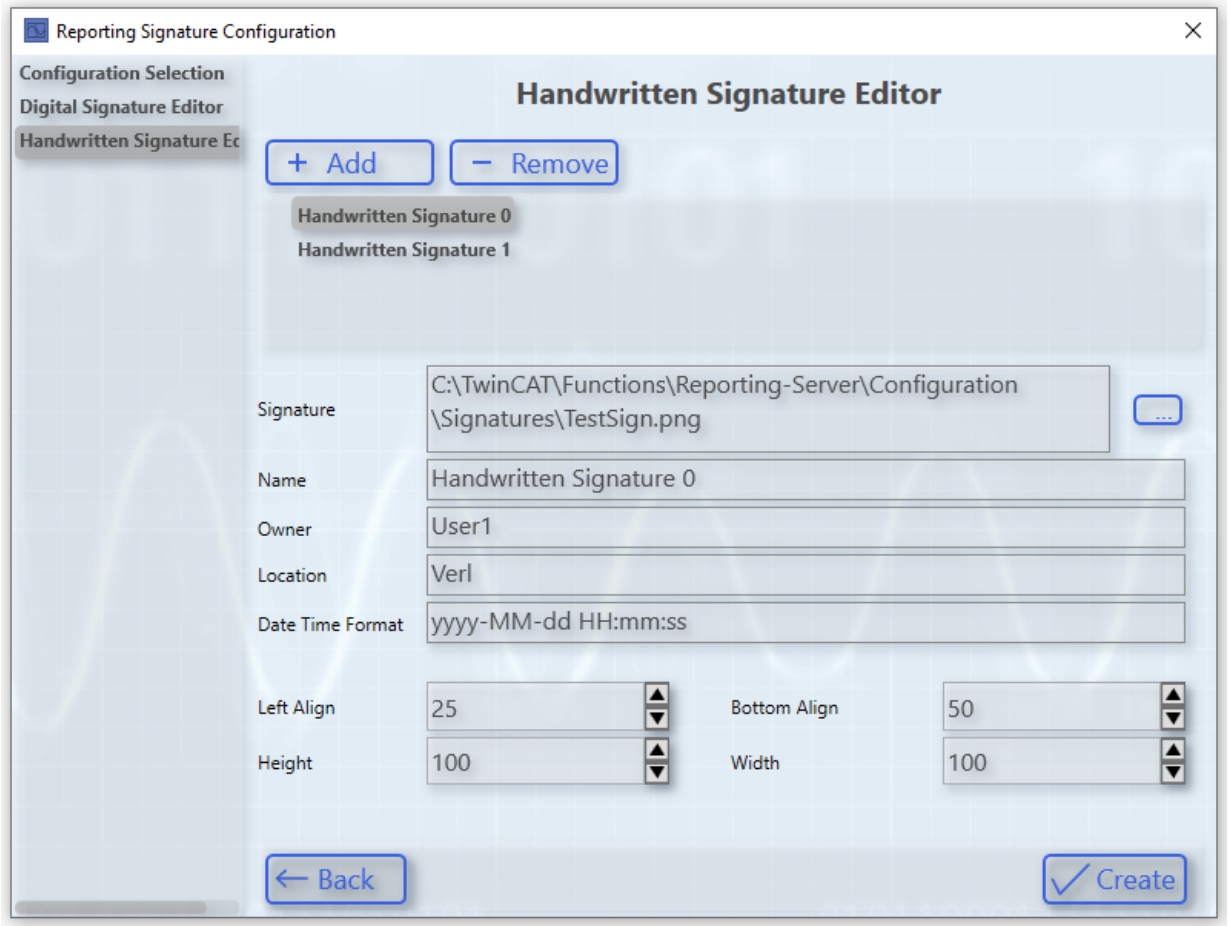
す。
Nextボタンをクリックすると、選択した設定を編集できます。

2. ここでデジタル署名を定義できます。



⇨ TwinCAT Reportingでは、PfxおよびP12証明書がサポートされています。

3. ここでは、手書きの署名を定義できます。



⇨ 手書きの署名を使用する場合、レポートに追加の署名ページが挿入されます。

4. 署名の位置は左揃えおよび下揃えにできます。署名ページの左下が原点です。
以下の画像は手書き署名の一例です：

⇨ 署名構成情報は、プレーンテキストのファイルとしてReporting Serverフォルダに格納されます。

6.10.3.6 ライセンス認証

TwinCAT Reportingのライセンスは、フルバージョンと7日間のトライアルバージョンがあります。
以下の製品には、TwinCAT Reportingライセンスが含まれています。

製品番号	製品型番
TE1300	TC3 Scope View Professional
TF3300	TC3 Scope Server
TE3500	TC3 Analytics Workbench
TE3520	TC3 Analytics Service Tool

以下の機能は、TwinCAT Reporting Serverのフルバージョンまたは7日間のトライアルバージョンで利用できます。

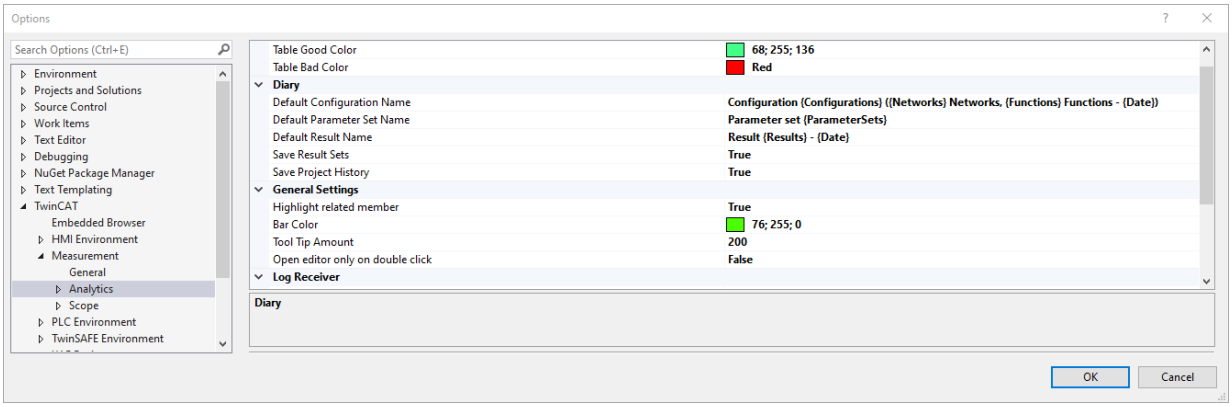
機能	TwinCAT Reporting Server	
	7日間トライアル	フルライセンス
全般:		
レポート	✔ (最大1)	✔ (無制限)
デジタル署名	✘	✔
手書き署名	✘	✔

Reporting Clientのトライアル制限については、製品概要ページを参照してください。

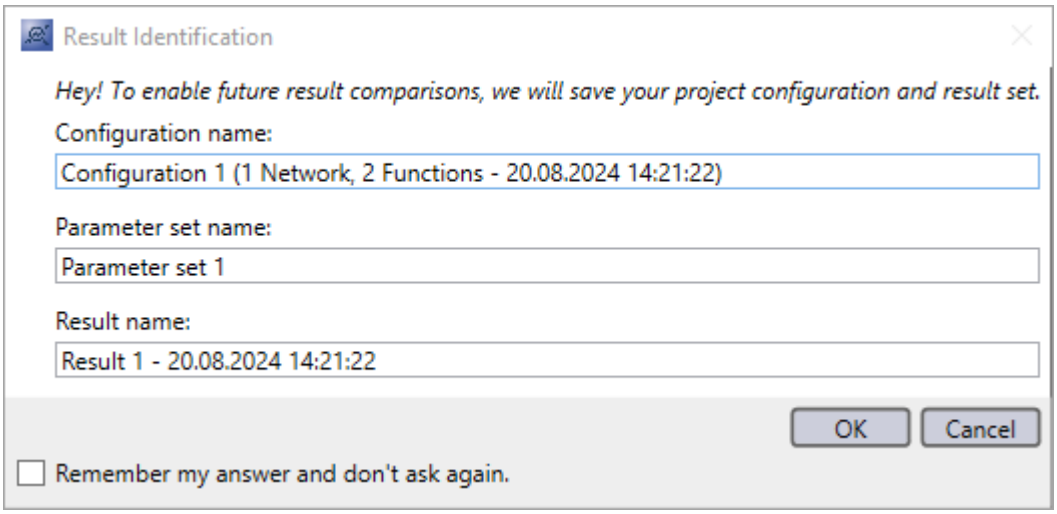
6.11 解析結果の比較

様々な解析結果をAnalytics Diaryで比較できます。Analytics Diaryは、探索的データ解析や、アルゴリズムのパラメータを含む適切な解析手法の特定において、さまざまな利点を提供します。

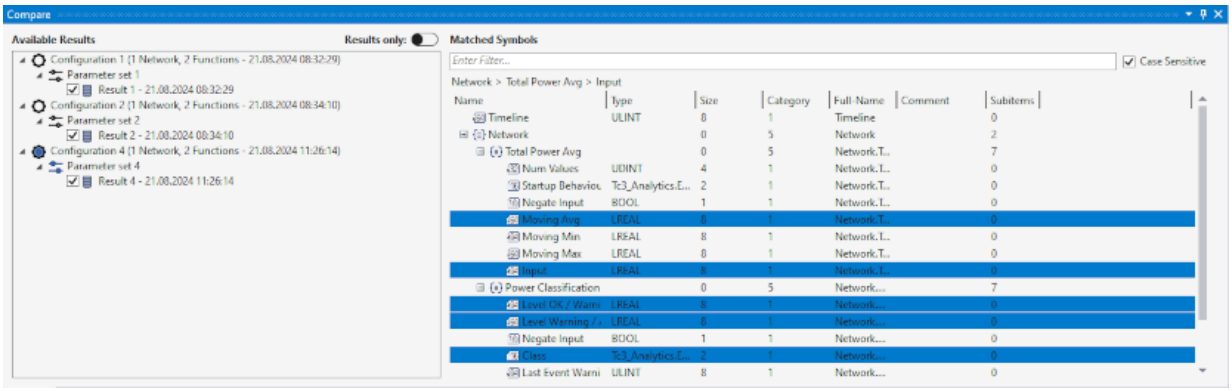
1. デフォルトでは、プロジェクト設定を含む全ての解析結果は、解析を実行するたびに保存されます。この設定は、
   Tools > Options > TwinCAT > Measurement > Analytics > Diaryで変更できます。



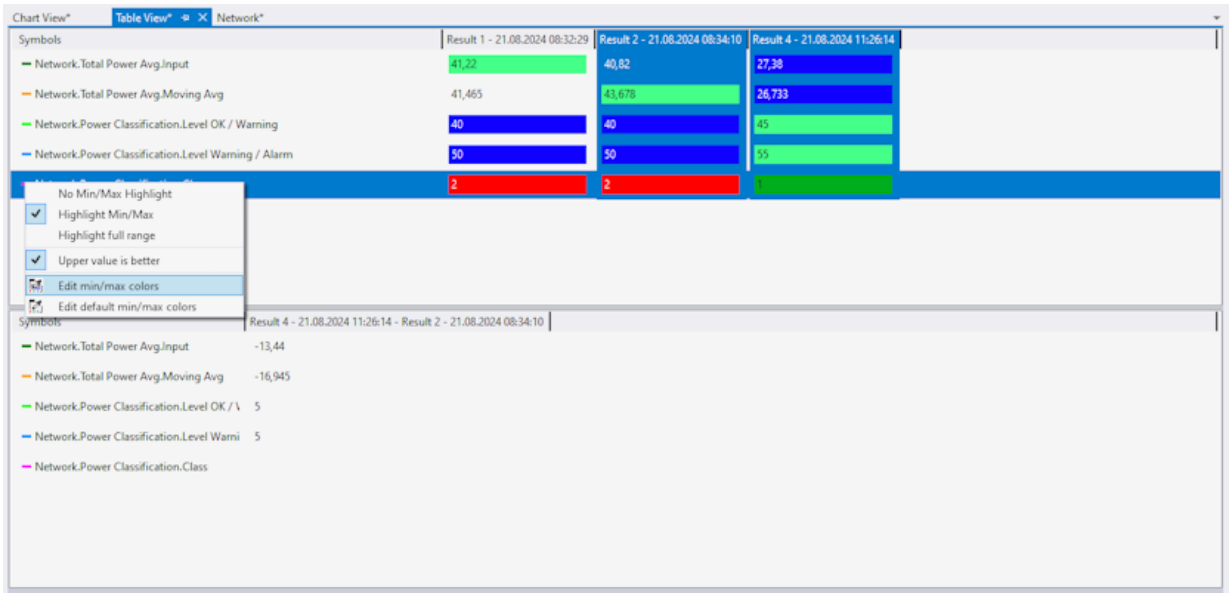
2. 解析を開始すると、Diaryウィンドウが表示されます。ここでは、設定、パラメータセット、結果の名前を設定できます。プロジェクトに何も変更がなければ、データだけが保存されます。同様に、パラメータ設定のみが変更された場合、解析設定は生成されません（グローバルパラメータ [▶ 79]も参照）。



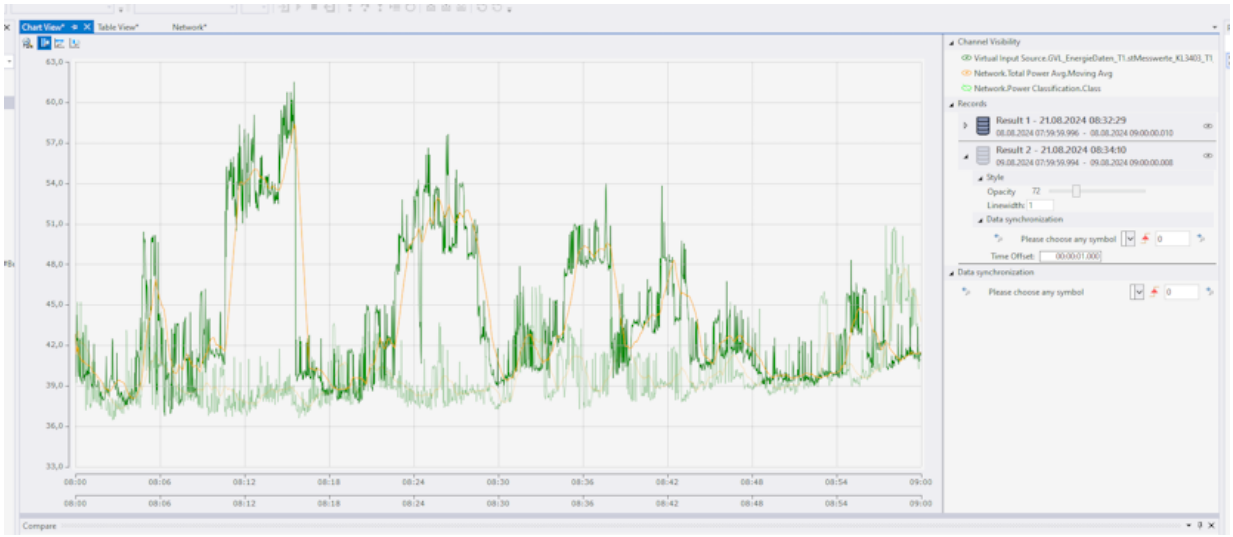
3. これらのエントリーは、**Compare**ウィンドウのツリーに表示され、**Solution Explorer**でクリックして開くことができます。これは解析の実行履歴を表示します。複数の結果データセットと一致するシンボルを選択し、**Table View**と **Chart View**ウィンドウで表示できます。したがって、データセットとシンボルの選択は、他の2つのウィンドウと同期しています。デフォルトでは常に最後のデータセットが選択されます。



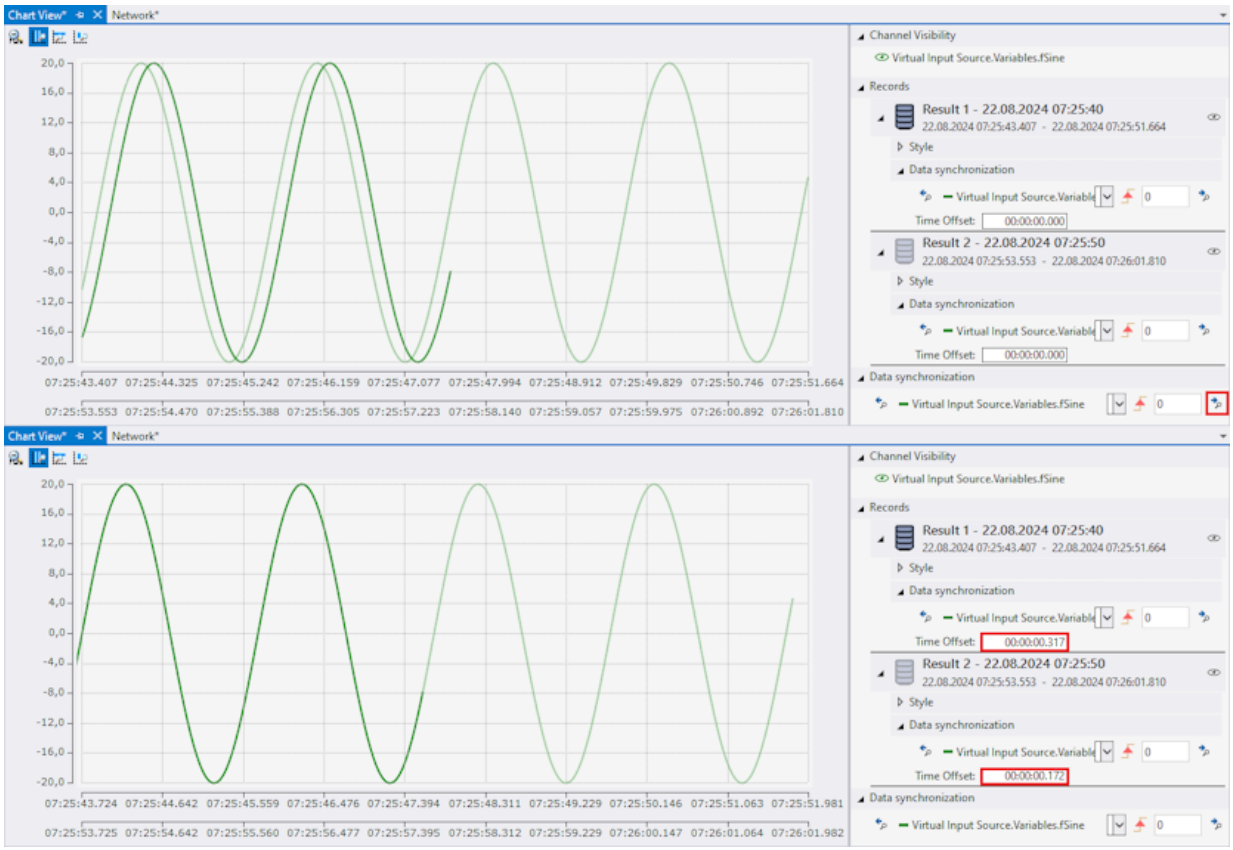
4. **Solution Explorer**をクリックすると、**Table View**と **Chart View**のウィンドウを開くことができます。**Table View**には2つのテーブルが表示されます。1つ目は、選択した全てのデータセットとシンボルの最後の結果を表示します。データセットは列ごとに、記号は行ごとに表示されます。オプションのカラーディスプレイを使用すると、シンボル間の違いを強調できます。2つ目のテーブルでは、各シンボルの違いを示すことができます。1つ目のテーブルから、少なくとも2列または2対のデータセットを選択する必要があります。



5. *Chart View*は、結果のデータセットを時系列で比較するために使用します。シンボルには異なる色が割り当てられています。シンボルは選択されたデータセットの数に応じて表示され、それぞれの線の不透明度が段階的に下がります。データセットを区別するために、不透明度と線の太さを手動で調整できます。より狭い範囲を見るには、スコープと同様にズーム機能を使用します。

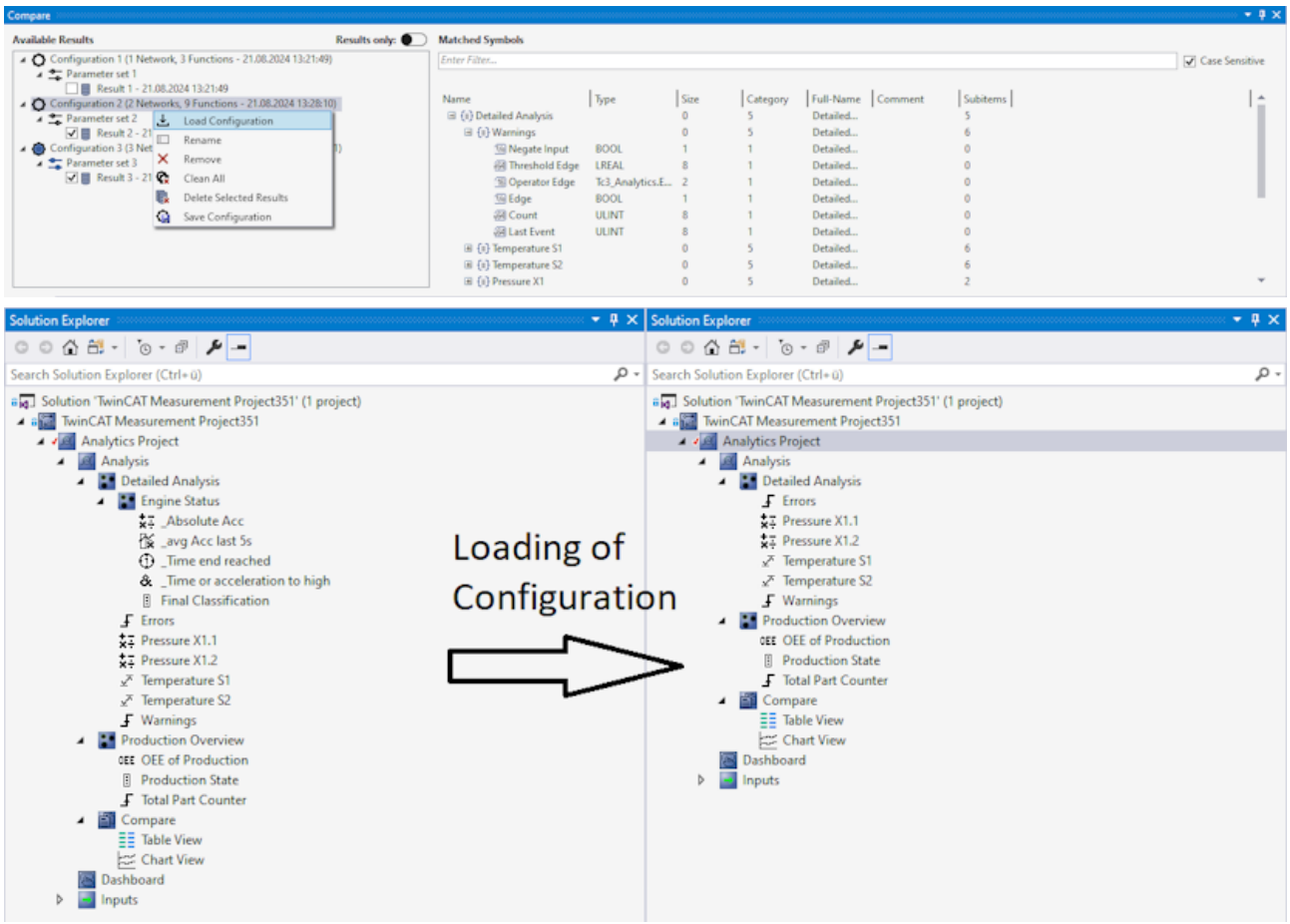


6. データセットの開始時間が一致しない場合は、比較のために調整できます。これは、時間オフセットを手動で入力するか、自動エッジサーチによって行うことができます。



6.11.1 プロジェクト設定の読み込みと保存

解析の実行履歴は、*Compare*ウィンドウに一覧表示されます。設定ごとに対応するプロジェクトステータスが保存され、後で読み込むことができます。設定を読み込むには、目的の設定（またはパラメータセット）を右クリックし、コンテキストメニューから*Load Configuration*オプションを選択します。これにより、古いプロジェクトのステータスが復元されます。Configurationアイコンが青くなり、最後に読み込まれた設定を示します。



解析開始時に設定を自動的に保存するだけでなく、手動で保存することもできます。これを行うには、コンテキストメニューから*Save Configuration*を選択します。

7 API

以下に示すTwinCAT Analytics Data Exchange APIを使用して、Analytics Workflowにデータを送信したり、Analytics Workflowからデータを取得したりできます。これにより、外部ソースに接続したり、他のソフトウェアコンポーネントで解析データを再利用したりできます。

7.1 Data Exchange API

Analytics Data Exchange API を使用して、サポートされている全てのプログラミング言語およびスクリプト言語で、独自のロガーまたはランタイムを構築できます。

ロガーによって、外部ソースからのデータをAnalyticsに保存したり解析したりできます。ランタイムは、Analyticsのデータを他のシステムで使用するようにプログラムされています。運用方法によっては、適切なライセンスが必要となります。ロガーについては、TF3500 Analytics Loggerライセンスです。ランタイム実装の場合は、TF3551 Analytics Runtime BaseまたはTF3550 Analytics Runtimeとなります。これらの製品が既にシステムで使用されている場合、別のプログラム実装のための追加ライセンスは必要ありません。

7.1.1 .NET

インストール方法

Analytics Data Exchange APIを使用するには、次の2つのコンポーネントをインストールする必要があります：

1. Microsoft Visual C++ Redistributable 2019 (x64/ x86)以上。このパッケージがない場合、シンボルの処理時にエラーが発生します。
2. TwinCAT 3 ADS

NuGetパッケージ

Analytics Data Exchange APIはNuGetパッケージとして提供されます。3種類のパッケージが用意されています：

1. Beckhoff.TwinCAT.Analytics.DataExchange.Core（.Net-Core3.1または.Net-5以上）
2. Beckhoff.TwinCAT.Analytics.DataExchange.フレームワーク（.Net-Framework 4.5.2以上）
3. Beckhoff.TwinCAT.Analytics.DataExchange（マルチターゲットプロジェクト）

7.1.1.1 ロガー機能のマッピング

ここでは、Analytics Data Exchange APIを使用してAnalytics形式でデータを送信または保存する方法について説明します。

ライセンス

TwinCAT Analytics形式でデータを書き込むには、TwinCAT 3 Analytics Loggerのライセンスが必要です。Data Exchange APIは7日間のトライアルライセンスでテストできます。

データストレージの作成と設定

Analytics形式でデータを送信または保存するには、まずビルダーを作成してデータストレージの設定をします：

```
ISinkFactory factory = new SinkFactory();

IFileSinkBuilder fileBuilder = factory.ToFile("C:\Data\Demo");

IMqttSinkBuilder mqttBuilder = factory.ToMessageBroker("127.0.0.1", 1883);
```

現在、ファイルへの書き込みとMQTTデータ送信の2種類のビルダーが利用できます。ビルダーは、メソッド呼び出しを連結することで設定されます：

```
mqttBuilder.WithSymbol("Symbol1", TypeCode.Double).WithCompressionWidth(8)
```

シンボルの設定には以下の方法があります：

一般パラメータ	説明
WithCompression(…)	データシンクの圧縮方法を設定します。
WithCompressionWidth(..)	CompressionWidthを大きくすると、圧縮率は下がりますが、処理は速くなります。3を下回る値は推奨されません。デフォルト: 8
WithMaxSamplesPerChunk(…)	MQTTメッセージまたはファイル内の最大データ数。デフォルト：MQTTの場合は32、ファイルの場合は500
WithMaxChunkSize(…)	MQTTメッセージまたはファイルの最大データサイズ。標準：制限なし
WithCycleTime(…)	データのサイクルタイム：デフォルト 10ms
シンボル:	
WithSymbol(…)	シンボルを追加します。オプションで、配列の長さを指定することもできます。
WithSymbols(Action)	複数のシンボルを同時に追加できます。
WithSymbol<T>	汎用データ型のシンボルを追加します。汎用データ型 [▶ 536]の項で、許可されるデータ型についての情報を参照できます。
MQTTのみ：	
ToTopic(…)	データが通信されるトピック。
ファイルのみ：	
WithIndexing(…)	圧縮を使用する場合の、インデックス付きの2つのデータポイント間の距離。インデックスを作成すると、データポイントを読み込む際のパフォーマンスが向上しますが、メモリ使用量は増加します。0はインデックスを無効にするもので、デフォルトで設定されています。有効化のためには、この値を少なくとも100に設定する必要があります。

MQTT通信用の特別なパラメータは、以下の方法で設定します：

WithTLS(…)	証明書が TLS を使用するように設定します。
WithCredentials(...)	メッセージブローカーにアクセスするためのユーザ名とパスワード。
WithTcpBufferSize(…)	メッセージブローカーへの TCP 接続のバッファサイズを設定します。デフォルトは32kB です。サイズの大きなメッセージを送信する場合は、この値を増やすことができます。
WithKeepAlivePeriod(…)	MQTT接続のKeepAlive時間を設定します。（デフォルトは60秒です。）
WithCommunicationTimeout(…)	メッセージブローカーへのTCP通信のタイムアウト。（デフォルトは3秒です。）
ToTopic(…) / FromTopic(…)	データが通信されるトピック。

データシンク設定は、設定したデータシンクを返すビルドメソッドを呼び出すことで完了します。データシンクには以下のメソッドがあります。データシンクを使用しなくなった場合は、Disposeメソッドを呼び出して全てのリソースを解放する必要があります。

InitWrite()	MQTT接続を初期化するか、書き込むファイルを開きます。また、最初の値が書き込まれたときにも自動的に呼び出されます。
Write(…)	バッファに新しい値を追加します。
Flush()	バッファに格納されているデータを送信します。Write によって自動的に呼び出されます。MaxSamplesまたはMaxSizeを超えた場合。
CompleteWrite()	バッファに保存されているデータを送信し、開いているMQTT接続またはファイルアクセスを閉じます。
Dispose()	バッファを送信せずに、開いている全てのMQTT接続またはファイルアクセスを閉じ、全てのリソースを解放します。その後、シンクは使用できなくなります。

汎用データ型：

Analytics Data Exchange APIは、汎用データ型として「アンマネージ構造体」のみをサポートしています。その結果、以下のような制約が生じます。

1. 構造体内で使用できるのは、構造体または基本データ型のみです。
2. 構造体内の配列は「unsafe」コードでのみ可能です。(Fixed-Array-Size)
3. 基本データ型の配列のみ使用可能。構造体の配列は不可能です。

さらに、構造体には逐次レイアウトか明示レイアウトしか使用できません。（参照：（https://docs.microsoft.com/de-de/dotnet/api/system.runtime.interopservices.structlayoutattribute?view=net-6.0 ）

1. データを書き込む際には、明示的なレイアウトを使用する必要があります。
2. データを書き込むとき、レイアウトはパディング（Pack = 1）を含まないようにします。
3. データを読み込む際には、パディングがデータソースに対応している必要があります(https://infosys.beckhoff.com/index.php?content=../content/1033/tc3_plc_intro/3539428491.html&id=)

エラーメッセージ

Data Exchange API の使用中にエラーが発生した場合、そのエラーは例外としてユーザに通知されます。以下の例外があります：

クラス名	スーパークラス	説明
DataExchangeException	Exception	API内のその他全ての例外のスーパークラス。
InvalidSymbolDefinitionException	DataExchangeException	設定されたシンボルが間違っています。
SymbolNotFoundException	InvalidSymbolDefinitionException	設定されたシンボルがデータストリームで見つかりません。
InvalidSymbolNameException	InvalidSymbolDefinitionException	入力されたシンボル名が無効です。
InvalidSymbolOffsetException	InvalidSymbolDefinitionException	入力されたシンボルのオフセットが 0 より小さいです。
InvalidSymbolByteSizeException	InvalidSymbolDefinitionException	入力されたバイト数が0以下です。
InvalidSymbolTypeCodeException	InvalidSymbolDefinitionException	シンボルに入力された TypeCode が無効です。
InvalidSymbolRegexPatternException	InvalidSymbolDefinitionException	入力されたシンボルの正規表現パターンが無効です。
InvalidSymbolFilterException	InvalidSymbolDefinitionException	正規表現パターンに入力されたシンボルフィルタが無効です。
DataExchangeLicenseException	DataExchangeException	APIの有効なライセンスが見つからないか、TwinCATがランモードになっていません。
DataExchangeTimeoutException	DataExchangeException	タイムアウト時間を超過しました。
QueueFullException	DataExchangeException	処理が遅すぎたため、データ処理用のキューが満杯になりました。
FileSourceNotFoundException	DataExchangeExeption	指定したフォルダにAnalyticsファイルが見つかりませんでした。
ValueExtractException	DataExchangeException	データストリームからの値の読み取り中にエラーが発生しました。詳細情報は、内部の例外に記載されています。
DataExchangeMQTTException	DataExchangeException	MQTT 接続でエラーが発生しました。
BrokerConnectException	DataExchangeMQTTException	ブローカーへの接続が確立できませんでした。

クラス名	スーパークラス	説明
TopicNotFoundException	DataExchangeMQTTException	指定されたトピックがブローカーで見つかりませんでした。
InvalidMqttStreamFormat	DataExchangeMQTTException	指定されたストリーム形式が間違っています。

サンプル：

Analytics Data Exchange APIの使用方法のサンプルはこちらから参照できます：

https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12970252171.zip

7.1.1.2 ランタイムファンクションのマッピング

このドキュメントでは、Analytics Data Exchange APIを使用してAnalyticsファイル形式からデータを読み取る方法について説明します。

ライセンス

TwinCAT Analyticsファイル形式からデータを読み込むには、TwinCAT 3 Analytics Runtime または TwinCAT 3 Analytics Runtime Base のライセンスが必要です。Data Exchange APIは7日間のトライアルライセンスでテストできます。

データソースを作成、設定する：

最初のステップは、Analytics形式からデータを読み込むためにビルダーを作成し、設定します。

```
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);
```

現在、3種類のビルダーが用意されています：

1. IFileSourceBuilder：ファイルからのデータ。
2. IMqttLiveSourceBuilder: Analytics Loggerデータストリームからのデータ。
3. IStorageProviderSourceBuilder: Analytics Storage Providerの記録データ。

ビルダーはメソッド呼び出しの連結によってパラメータ化されます。提供可能なオプションは以下の通りです。

シンボル	説明
WithSymbol(…)	読み取り用のシンボルを追加します。利用可能なシンボルは、TwinCAT Target Browserから読み込むことができます。
WithSymbols(...)	複数のシンボルを同時に追加できます。
WithSymbols(pattern, …)	正規表現を使って複数の記号を追加します。さらに、RegexOptionsとSymbolFilterも指定できます。 シンボルフィルタ： Include Parent：子シンボルが適用されても親シンボルを使用します。 IncludeChilds：親シンボルが適用されても、子シンボルを使用します。 IncludeChildsまたはInlcudeParentのいずれかを設定する必要があります。
WithSymbol<T>()	汎用データ型のシンボルを追加します。汎用データ型 [▶ 540]の項で、許可されるデータ型についての情報を参照できます。
MQTTのみ：	

シンボル	説明
FromTopic(…)	データを読み込むMQTTトピック。
WithDataTimeOut(…)	2つのMQTTメッセージ間のタイムアウト（デフォルト500ms）。-1msは無限のタイムアウトを意味します。
WithSymbolTimeOut(…)	メッセージブローカーからのシンボル情報受信タイムアウト（デフォルト1s）。-1msは無限のタイムアウトを意味します。
WithSortQueueSize(...)	受信メッセージのバッファサイズを設定します。バッファはメッセージの並べ替えにも使用されます。 (デフォルト： 最小：5 最大：25)
WithStreamFormat(…)	受信データの形式を指定します。何も指定しない場合、形式は自動検出されます。自動検出のためには、データストリームが起動時にオンラインである必要があります。
Storage Providerとファイル	
From(…)	データ検索プロセスの開始。
To(…)	データ検索プロセスの終了。
StorageProviderのみ	
FromStorageProivder(…)	データの取得に使用するブローカーのストレージプロバイダを指定します。
FromHistoricalStream(…)	ProviderIdとMainTopicは、指定したトピックから読み込まれます。指定していない場合は、FromStorageProviderメソッドで渡す必要があります。
FromLayout(…)	履歴データストリームのレイアウト。
FromOriginSystem(…)	データが最初に送信されたシステムID。
FromOriginStreamTopic (…)	データが最初に送信されたトピック。
FromStreamAlias(…)	データストリームが記録され、TargetBrowserに表示されるエイリアス。

FromLayoutメソッド、FromOriginSystemメソッド、FromOriginStreamTopicメソッド、FromStreamAliasメソッドは、正しいデータセットを識別するために使用されます。全てのパラメータを指定することは必須ではありませんが、データセットの混乱を避けるため、できるだけ多くのパラメータを指定することが推奨されます。この情報を読む最も簡単な方法は、Target Browserを使用することです。これを行うには、データセットのシンボルをテキストエディタ（Notepad ++など）にドラッグします。必要な情報は、XMLのTarget Infoセクションに記載されています。

MQTT通信用の特別なパラメータは、以下の方法で設定します：

WithTLS(…)	証明書が TLS を使用するように設定します。
WithCredentials(...)	メッセージブローカーにアクセスするためのユーザ名とパスワード。
WithTcpBufferSize(…)	メッセージブローカーへの TCP 接続のバッファサイズを設定します。デフォルトは32kB です。サイズの大きなメッセージを送信する場合は、この値を増やすことができます。
WithKeepAlivePeriod(…)	MQTT接続のKeepAlive時間を設定します。（デフォルトは60秒です。）
WithCommunicationTimeout(…)	メッセージブローカーへのTCP通信のタイムアウト。（デフォルトは3秒です。）
ToTopic(…) / FromTopic(…)	データが通信されるトピック。

データソース設定は、設定したデータソースを返すビルドメソッドを呼び出すことで完了します。データソースは様々な方法を提供しています。データソース使用しなくなった場合は、Disposeメソッドを呼び出して全てのリソースを解放する必要があります。

InitSymbols()	ReadSymbolsコレクションを初期化し、既存の正規表現を展開・解釈します。
ReadSamples(...)	呼び出し元コンテキスト内の全てのデータを同期して問い合わせます。
ReadAsync(...)	全てのデータを非同期で問い合わせます。ReadDelegateは日付ごとに呼び出されます。

Dispose()	開いている全てのMQTT接続またはファイルアクセスを閉じ、全てのリソースを解放します。その後、データソースは使用できなくなります。

汎用データ型：

Analytics Data Exchange APIは、汎用データ型として「アンマネージ構造体」のみをサポートしています。その結果、以下のような制約が生じます。

1. 構造体内で使用できるのは、構造体または基本データ型のみです。
2. 構造体内の配列は「unsafe」コードでのみ可能です。(Fixed-Array-Size)
3. 基本データ型の配列のみ使用可能。構造体の配列は不可能です。

さらに、構造体には逐次レイアウトか明示レイアウトしか使用できません。（参照：（https://docs.microsoft.com/de-de/dotnet/api/system.runtime.interopservices.structlayoutattribute?view=net-6.0 ）

1. データを書き込む際には、明示的なレイアウトを使用する必要があります。
2. データを書き込むとき、レイアウトはパディング（Pack = 1）を含まないようにします。
3. データを読み込む際には、パディングがデータソースに対応している必要があります(https://infosys.beckhoff.com/index.php?content=../content/1033/tc3_plc_intro/3539428491.html&id=)

構造化データ型：

汎用データ型が指定されていない場合、Analytics Data Exchange APIはシンボルのデータ型を自動的に検出します。使用されているシンボルが構造化データ型の場合、IStructuredValueが返されます。サブシンボルの値は、一方ではIStructuredValueから問い合わせることができます。もう一方で、これはJSONテキストに直接変換できます。最後の手段として、XMLReaderのようにシンボル全体を解析するためのIStructuredValueReader を作成する方法があります

```
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
    }
  }
}
```

エラーメッセージ

Data Exchange API の使用中にエラーが発生した場合、そのエラーは例外としてユーザに通知されます。以下の例外があります：

クラス名	スーパークラス	説明
DataExchangeException	Exception	API内のその他全ての例外のスーパークラス。
InvalidSymbolDefinitionException	DataExchangeException	設定されたシンボルが間違っています。
SymbolNotFoundException	InvalidSymbolDefinitionException	設定されたシンボルがデータストリームで見つかりません。
InvalidSymbolNameException	InvalidSymbolDefinitionException	入力されたシンボル名が無効です。
InvalidSymbolOffsetException	InvalidSymbolDefinitionException	入力されたシンボルのオフセットが 0 より小さいです。
InvalidSymbolByteSizeException	InvalidSymbolDefinitionException	入力されたバイト数が0以下です。

クラス名	スーパークラス	説明
InvalidSymbolTypeCodeException	InvalidSymbolDefinitionException	シンボルに入力された TypeCode が無効です。
InvalidSymbolRegexPatternException	InvalidSymbolDefinitionException	入力されたシンボルの正規表現パターンが無効です。
InvalidSymbolFilterException	InvalidSymbolDefinitionException	正規表現パターンに入力されたシンボルフィルタが無効です。
DataExchangeLicenseException	DataExchangeException	APIの有効なライセンスが見つからないか、TwinCATがランモードになっていません。
DataExchangeTimeoutException	DataExchangeException	タイムアウト時間を超過しました。
QueueFullException	DataExchangeException	処理が遅すぎたため、データ処理用のキューが満杯になりました。
FileSourceNotFoundException	DataExchangeExeption	指定したフォルダにAnalyticsファイルが見つかりませんでした。
ValueExtractException	DataExchangeException	データストリームからの値の読み取り中にエラーが発生しました。詳細情報は、内部の例外に記載されています。
DataExchangeMQTTException	DataExchangeException	MQTT 接続でエラーが発生しました。
BrokerConnectException	DataExchangeMQTTException	ブローカーへの接続が確立できませんでした。
TopicNotFoundException	DataExchangeMQTTException	指定されたトピックがブローカーで見つかりませんでした。
InvalidMqttStreamFormat	DataExchangeMQTTException	指定されたストリーム形式が間違っています。

サンプル：

Analytics Data Exchange APIの使用方法のサンプルはこちらから参照できます：

https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12970252171.zip

7.1.2 Python

インストール方法

Analytics Data Exchange APIを使用するには、次の2つのコンポーネントをインストールする必要があります：

1. Microsoft Visual C++ Redistributable 2019 (x64/ x86)以上。このパッケージがない場合、シンボルの処理時にエラーが発生します。
2. TwinCAT 3 ADS

7.1.2.1 ロガー機能のマッピング

ここでは、Analytics Data Exchange APIを使用してAnalytics形式でデータを送信または保存する方法について説明します。

ライセンス

TwinCAT Analytics形式でデータを書き込むには、TwinCAT 3 Analytics Loggerのライセンスが必要です。Data Exchange APIは7日間のトライアルライセンスでテストできます。

データストレージの作成と設定

Analytics形式でデータを送信または保存するには、まずビルダーを作成してデータストレージの設定をします：

```
Factory = SinkFactory()

Builder = Factory.ToFile(FilePath)

Builder = Factory.ToMessageBroker(Broker, 1883)
```

ビルダーによって、設定方法は異なります。シンボルは WithSymbolメソッドを使って追加できます。記号はいくつでも使用できます。シンボルの実装は、ほとんどの基本的なデータ型で既に利用可能です。構造化データ型や配列も実装できます。

```
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)
```

ビルダーの設定が完了すると、Buildメソッドを使用して新しいシンクが作成され、InitWriteで初期化されます。その後、Writeメソッドを使ってシンボルの現在値を書き込みます。シンボルの値は、Writeメソッドを複数回呼び出す間に更新できます。全てのデータが書き込まれたら、CompleteWriteで書き込み処理を終了します：

```
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()
```

サンプル:

完全なサンプルはこちらにあります：https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/17763893899.zip

7.1.2.2 ランタイムファンクションのマッピング

このドキュメントでは、Analytics Data Exchange APIを使用してAnalyticsファイル形式からデータを読み取る方法について説明します。

ライセンス

TwinCAT Analyticsファイル形式からデータを読み込むには、TwinCAT 3 Analytics Runtime または TwinCAT 3 Analytics Runtime Base のライセンスが必要です。Data Exchange APIは7日間のトライアルライセンスでテストできます。

データソースを作成、設定する：

```
Factory = SourceFactory()

Builder = Factory.FromLiveStream(Broker, 1883)

Builder = Factory.FromFile(FilePath)
```

ビルダーによって、設定方法は異なります。シンボルは WithSymbol、WithName、WithRegex メソッドを使用して追加できます：

```
TestValue = UInt64("Variables.nGrowSteady")
Builder.WithSymbol(TestValue)

Builder.WithName("TestValues.nGrowSteady")
Builder.WithRegex("Variables.f", 0, RegexSymbolFilter.IncludeChilds)
```

ビルダーの設定が完了すると、ビルドメソッドを使用して新しいシンクが作成されます。Readメソッドを呼び出すとデータ問い合わせが開始します。Callbackメソッドがパラメータとして渡されます。サンプルにシンボルが含まれていない場合、全てのデータは既に読み出されています。Callbackメソッドが0以外の値を返した場合、読み取りは中断されます。

```
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)
```

サンプル:

完全なサンプルはここにあります：https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/17763978891.zip

8 サンプル

8.1 ピック&プレース アプリケーション

次のサンプルは、XTSとピック&プレースロボットを組み合わせたアプリケーションの解析です。2つのオプションがあります：

1. サンプルデータを使用して、独自のTwinCAT Analyticsプロジェクトを構築できます。
2. 既成のTwinCAT Analyticsプロジェクトを使用して、TwinCAT Analyticsの動作を理解できます。

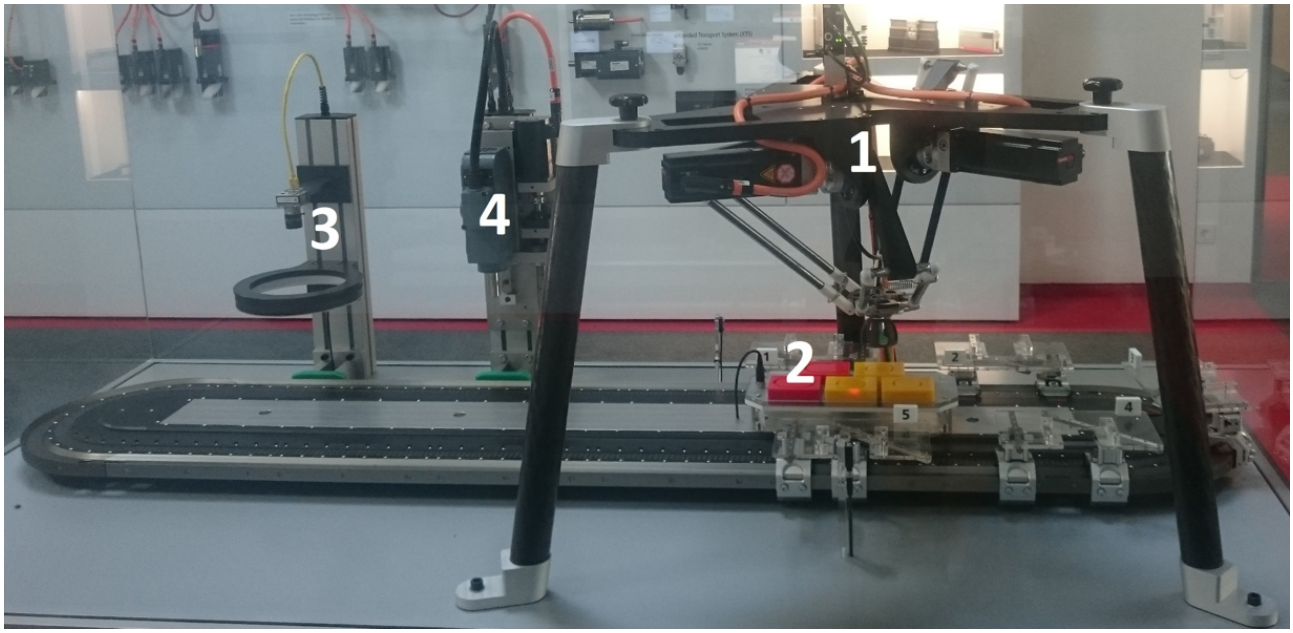
いずれの場合も、解析データを提供する機械アプリケーションへの理解を深めることは有用です。アプリケーションのシーケンスは動画で確認できます：

動画:https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/6815846411.mp4.

アプリケーションの説明

この例では、生産財の加工シミュレーションについて説明します。このアプリケーションは、ベッコフのXTSシステムを使用して、ワークを保管場所からドリル加工ステーションまで搬送します。ワークピースの色によって、それぞれ異なるパターンがフライス加工されます。XTSシステムの全長は4000mmです。可動子は10台で、2台1組で1つのトレーを搬送します。ピック&プレースロボット(1)がワークピース保管場所(2)に配置されています。赤ワーク2個と黄ワーク3個が、5つのトレーにの配置され、ピッカーは真空ポンプを使ってワークをピック&プレースします。XTSの復路側には、ワークの色を識別するためのビジョンシステム(3)と、特定のフライス加工を施すためのフライス盤(4)があります。

トレーはワークピース格納場所まで走行します。ピッカーはワークをピックし、保管場所のトレーに積み下ろしをします。その後、トレーはスキャナーに送られ、そこでワークピースの色がスキャンされ、下流のフライス盤に伝送します。ワークの色によって加工が変わります。黄色の部品には円形を、赤色の部品には楕円形を加工します。ワークはその後、ピッカーによって保管場所に戻され、保管されます。トレーには別の部品が載せられ、搬送が継続します。



変数の説明

以下の変数は、TwinCATアナリティクスロガーによって付属のAnalyticsファイル形式で記録されます。このデータは、Analyticsコンフィグレータプロジェクトに使用することも、提供されたtnzipファイルにあるデフォルトのコンフィグレータに使用することもできます。各変数の意味と機能は以下の表の通りです。

変数名	データ型	説明
bMillerMilling	BOOL	ドリルのアクティビティに関する情報：ドリルがアクティブな場合は1、非アクティブな場合は0。
bMillerMovingZ	BOOL	ドリルのZ方向の動きに関する情報：ドリルがZ方向に動く場合は1、動かない場合は0。
bMillerSpindleRotation	BOOL	ドリルの針の動きに関する情報：針が動いていてドリルがアクティブなときは1、動いていないときは0。
bNewWorkpieceColorScanned	BOOL	新しい色がスキャンされたかどうかを登録し(= 1)、ドリルに通知します。色が変わる場合は、ドリルのパターンを変更する必要があります。色が同じ（=0）の場合、ドリルのパターンは変化しません。
bPickerHoldingWorkpiece	BOOL	ピッカーがワークを保持している場合は1、保持していない場合は0。
bPickerInStorage1Position	BOOL	ピッカーが保管位置 1 にある場合は 1。
bPickerInStorage2Position	BOOL	ピッカーが保管位置2にある場合は1。
bPickerInStorage3Position	BOOL	ピッカーが保管位置3にある場合は1。
bPickerInStorage4Position	BOOL	ピッカーが保管位置4にある場合は 1。
bPickerInStorage5Position	BOOL	ピッカーが保管位置5にあるときは 1。
bPickerInXtsPosition	BOOL	ピッカーがXTS位置、つまりXTSの上にある場合は1。
bPickerMoving	BOOL	ピッカーの情報： 、ピッカーが動いている (=1)、または動いていない (=0)。
fMillerPositionZ	LREAL	ドリルのZ位置を示します。224の場合、ドリルはドリル加工位置にあり、245の場合、ドリルは非アクティブです。
fMillerSpindleRotation	LREAL	ドリル針の1分あたりの回転数を示します。
nColorOfLastScannedWorkpiece	INT	最後にスキャンしたワークの色を登録します。 5 →黄色のワークピース 4 → 赤色のワークピース
nCurrentProductionTime	ULINT（64ビット）	現在の生産時間を登録します。
nVacuumPressure	DINT	ピッカーは圧縮空気でワークを把持します。この変数は、現在の圧力をmbarで示します。
stMachineVibrations aMachineVibrations[1...10]	LREAL配列を含むSTRUCT	stMachine Vibrations 構造体は、機械ベースの振動をオーバーサンプリングで、単位m/s^2で示します。
stPickerPosition aPickerPosition[1...3]	LREAL配列を含むSTRUCT	stPickerPosition構造体には、3つの値を持つ配列が含まれます。この値はピッカーのXYZ位置を示します。
stTableInMillingPosition aTableInMillingPosition[1...5]	BOOL配列を含むSTRUCT	stTableInMillingPosition構造体は5つの値を持つ配列を含み、各値はトレーを表します。インデックス1はトレー1、2はトレー2を示します。値が1の場合、該当するトレーはドリル位置にあります。
stTableInStoragePosition aTableInStoragePosition[1...5]	BOOL配列を含むSTRUCT	stTableInStoragePosition構造体は5つの値を持つ配列を含み、各値はトレーを表します。インデックス1はトレー1、2はトレー2を示します。値が1の場合、該当するトレーは保管位置にあります。

変数名	データ型	説明
stWorkpieceColors aWorkpieceColors[1...5]	INT配列を含むSTRUCT	stWorkpieceColors構造体は5つの値を持つ配列を含み、各値は1つのワークの色を（Int値として）示します： Workpiece 1: yellow Workpiece 2: yellow Workpiece 3: red Workpiece 4: red Workpiece 5: yellow
stWorkpiecePositions aWorkpiecePosition[1...5]	INT配列を含むSTRUCT	stWorkpiecePositions構造体には、各ワークに対し5つの値を持つ配列が含まれます。Int値はワークピースの現在位置を示します： 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]	LREAL配列を含むSTRUCT	stXtsMoverAcceleration構造体は、可動子1～10の10個の値を持つ配列を含み、特定の可動子の加速度を示します。
stXtsMoverMovingNegative aXtsMoverMovingNegative[1...10]	BOOL配列を含むSTRUCT	stXtsMoverMovingNegative構造体は、可動子1～10の10個の値を持つ配列を含み、特定の可動子が後方に移動している場合は1となります。
stXtsMoverMovingPositive aXtsMoverMovingPositive[1...10]	BOOL配列を含むSTRUCT	stXtsMoverMovingPositive構造体は、可動子1～10の10個の値を持つ配列を含み、特定の可動子が前進している場合は1となります。
stXtsMoverPositionsX aXtsMoverPositionsX[1...10]	LREAL配列を含むSTRUCT	stXtsMoverPositionsX構造体には、可動子1～10の10個の値が配列されています。この値は、XTS上の各可動子のX座標を示します。
stXtsMoverVelocity aXtsMoverVelocity[1...10]	LREAL配列を含むSTRUCT	stXtsMoverAcceleration構造体は、可動子1～10の10個の値を持つ配列を含み、特定の可動子の速度を示します。

データの入手方法

ダウンロードエリアから、XTSシステムの30分間の記録が入手可能で、詳しい解析に活用できます。これらのデータにアクセスするには、Analytics Storage Providerを使用して、履歴データを利用できるようにする必要があります。そのためには、メッセージブローカーが必要です。提供のサンプルは、全てのコンポーネントがローカルシステム上で動作していることを想定しています。

Analytics Storage Providerのインストール 指示に従ってください。

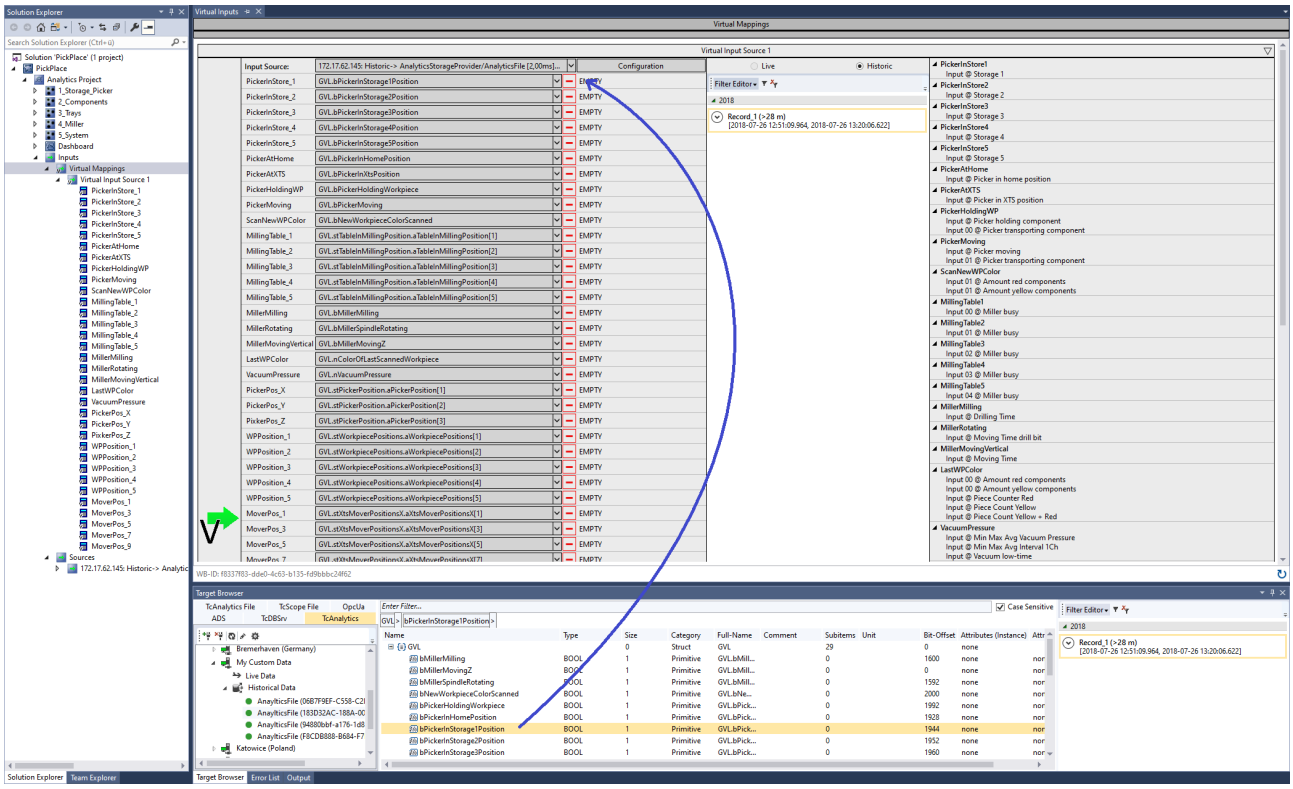
1. AnalyticsファイルデータをStorage Providerにインポートします（Analyticsファイルのインポートを参照）。
2. Analytics Storage Provider コンフィグレータを使用して、Storage ProviderWindowsサービスを開始します。

3. TcXaeShellまたはVisual Studio®でTwinCAT Target Browserを開きます。TcAnalyticsタブとメッセージブローカーを選択します。左側のツリーには履歴データの項目があり、選択すると右側のシンボルリストに表示されます。
4. 個々のシンボルをAnalyticsアルゴリズムの入力に接続します。
5. 解析を開始します。

Analytics Workbenchによる解析

ダウンロードエリアのTwinCAT Analyticsプロジェクトの対象となる解析について以下に説明します。完成したプロジェクトをダウンロードして開きます。

Virtual Mappings ノードを選択し、Target Browser でストレージ プロバイダが指定したトピックを参照します。エディタでデータを選択し、適切な仮想入力にリンクします。



Analyticsプロジェクトの説明

ネットワーク名	使用する変数	使用するアルゴリズム	説明
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: このネットワークはピッカーのアクションを解析します。ピッカーがワークを保持している場合、スタート位置にある場合、または XTS 位置にある場合に数量をカウントします。また、ピッカーがどれくらいの頻度でワークを搬送しているか、ピッカーがどれくらいの頻度で可動子にロードしているか、さらにそのXYZ位置も解析されます。 ピッカーはどのくらいの頻度でワークを積み込んでいるか？ Storage:

ネットワーク名	使用する変数	使用するアルゴリズム	説明
			このネットワークは、ピッカーに隣接する5つの保管場所の使用状況を解析します。保管場所の使用量と割合がカウントされます。 各保管場所が均等に使用されているか？
2_Components	bNewWorkpieceColorScanned, bMillerMilling, nColorOfLastScannedWorkpiece, stWorkpiecePosition	Logic Operation Counter, Threshold String Classificator, Histogram, Trend Line	このネットワークはワークそのものを解析します。スキャナとフライス盤で各色のワークがいくつ処理されているか、また各ワークの現在位置が検出できます。Histogramアルゴリズムは、ワークピースの位置の分布を示します。 赤色のワークピースはいくつ処理されたか？
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	距離： このネットワークは、各トレーの移動距離を解析し、全ての可動子の総距離を計算します。XTS長（4000mm）は既知であるため、各トレーの平均周回数を導き出すことができます。Threshold Analysisでは、40周で警告、45周でアラームが出ます。 全トレーの総移動距離は？ トレーの位置： このネットワークは、トレーが保存場所やフライス盤で停止する頻度を解析し、トレーが1周移動するのにかかる時間を記録します。 全トレーのフライス盤で停止する頻度は？ 速度： このネットワークは、トレーの現在の速度と、直近20秒間の最小および最大速度を解析します。トレーは後方に動くことができます。したがって、最小速度は負になる可能性があります。 読み取り可能な最高速度は？
4_Miller	bMillerMilling, bMillerSpindleRotating, bPickerMoving	Event Timing Analysis, Lifetime Analysis	このネットワークはフライス盤の解析をします。 Drilling time：フライス盤が作動している時間。 Drill bit movement time：ドリルビット（ドリル刃）が作動している時間。 Drill movement time：ドリルが動作している時間。 3つのアクションのうち、最も時間を費やしたのは？
5_System	bNewWorkpieceColorScanned, bMillerMilling,	Min Max Avg, Min Max Avg Interval, Timing Analysis,	生産性：

ネットワーク名	使用する変数	使用するアルゴリズム	説明
	nVacuumPressure, stMachineVibrations	Math Operation Lifecycle Analysis, Threshold Classificator, Continuous Piece Counter	生産性は、20秒ごとに処理されたワークピースをカウントし、前の区間と比較することで解析されます。最大カウントに基づき、後続の区間の生産性が導き出されます。 区間内に処理されたワークピースの最大数は？ 技術コンポーネント： このネットワークは、ピッカーの真空圧を解析し、吸引ユニットの耐用年数が定義された閾値を下回ると、警告またはアラームメッセージを発します。 警告までの残存時間はどのくらいか？

PLCコード生成とHMI

Analytics Workbenchでの解析が完了したら、PLCコードとHMIダッシュボードを生成できます（24時間解析アプリケーション [▶ 34] も参照してください）：

1.プロジェクトノードをダブルクリックすると、概要ページが開きます。**deploy Runtime**を選択します。

2.**Deploy Wizard**が開き、ステップバイステップでガイドします。個別の設定を必要としない場合は、そのまま進みます。

3.オプション： PLC タブで、解析を実行する CPU コアを選択します。仮想マシンでの実行には、独立したコアを選択する必要があります。

4.HMI Dashboard タブで、生成を有効にします。お好みに合わせてデザインを調整してください。

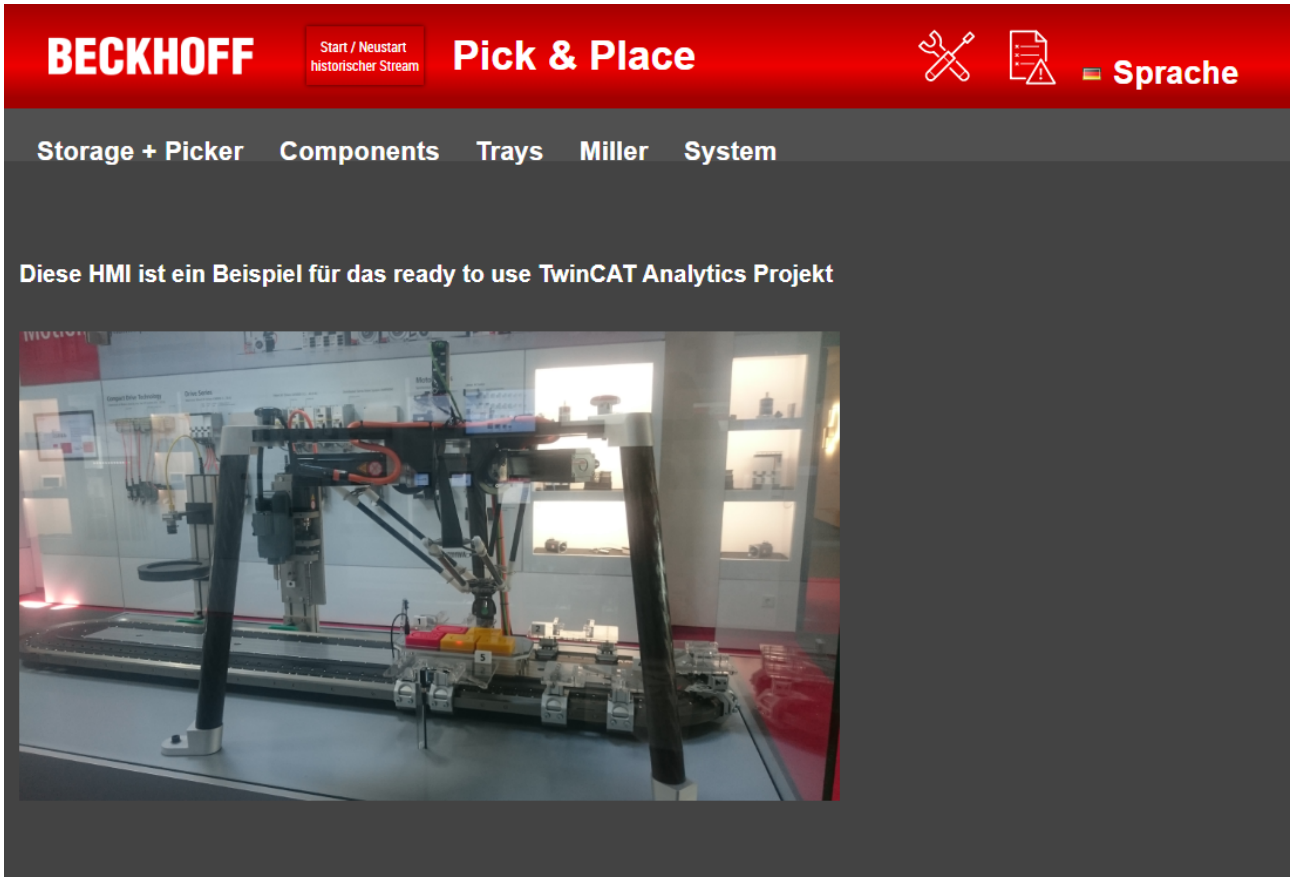
5.**Deploy**ボタンで確認すると、新しい開発環境が開きます。TwinCAT XAEプロジェクトとTwinCAT HMIプロジェクトが自動的に生成されます。

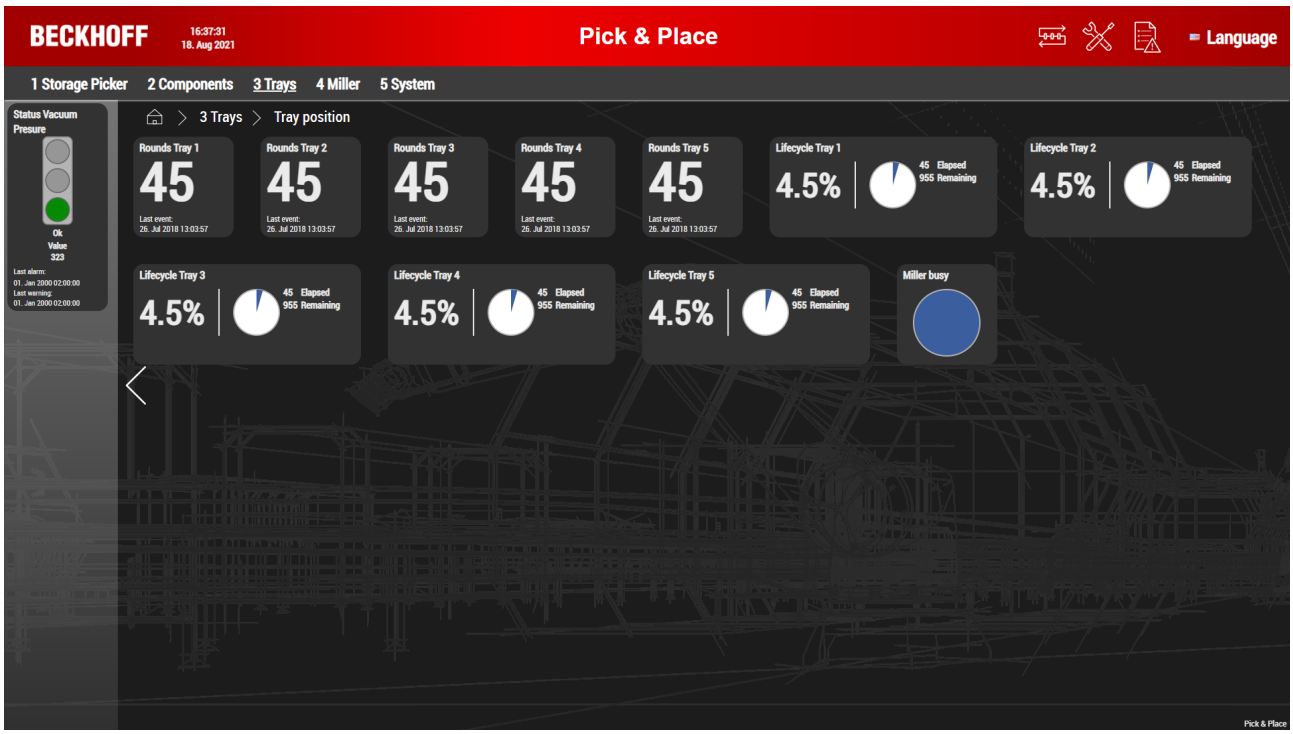
6.生成が終わると、ブラウザにダッシュボードが開きます。ナビゲーションを使用して、独自のネットワークを確認できます。

7.過去のデータを解析するためには、解析を開始する必要があります。これを行うには、上部の右から3番目のアイコンから該当のオプションを開きます。

8.解析結果を追跡するために、解析する記録データの時間範囲を選択し、記録を開始します。

BECKHOFF
16:57:11
18. Aug 2021
Pick & Place
Language
1 Storage Picker
2 Components
3 Trays
4 Miller
5 System
Status Vacuum Presure
Alarm Value -458
2 Components
Positions component 1
Positions component 2
Positions component 3
Positions component 4
Amount red components
157
Amount yellow components
237
Position component 1
Tray
Position component 2
Storage
Position component 3
Tray
Position component 4
Storage
Position component 5
Tray
Positions component 5
Trend Line
1 Hour
12 Hours
1 Day
3 Days
7 Days
Red components
Yellow components
Pick & Place


BECKHOFF
Start / Neustart historischer Stream
Pick & Place
Sprache
Storage + Picker
Components
Trays
Miller
System
Diese HMI ist ein Beispiel für das ready to use TwinCAT Analytics Projekt



完成した生成PLCコードのダウンロード

Analytics PLCプロジェクトと付帯するHMIを見たい場合は、ダウンロードも可能です（Sample_Analysis_PickNPlace_PLC）。このプロジェクトを実行するには、メイントピックとストレージプロバイダの識別用GUIDをコード内で適切に設定する必要があります。適切な警告が出力され、それによりTwinCAT内の該当箇所へ簡単に移動できます。また、その場所には関連情報への手がかりも表示されます。

ダウンロード

- 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 ピック&プレースアプリケーション：DenStream

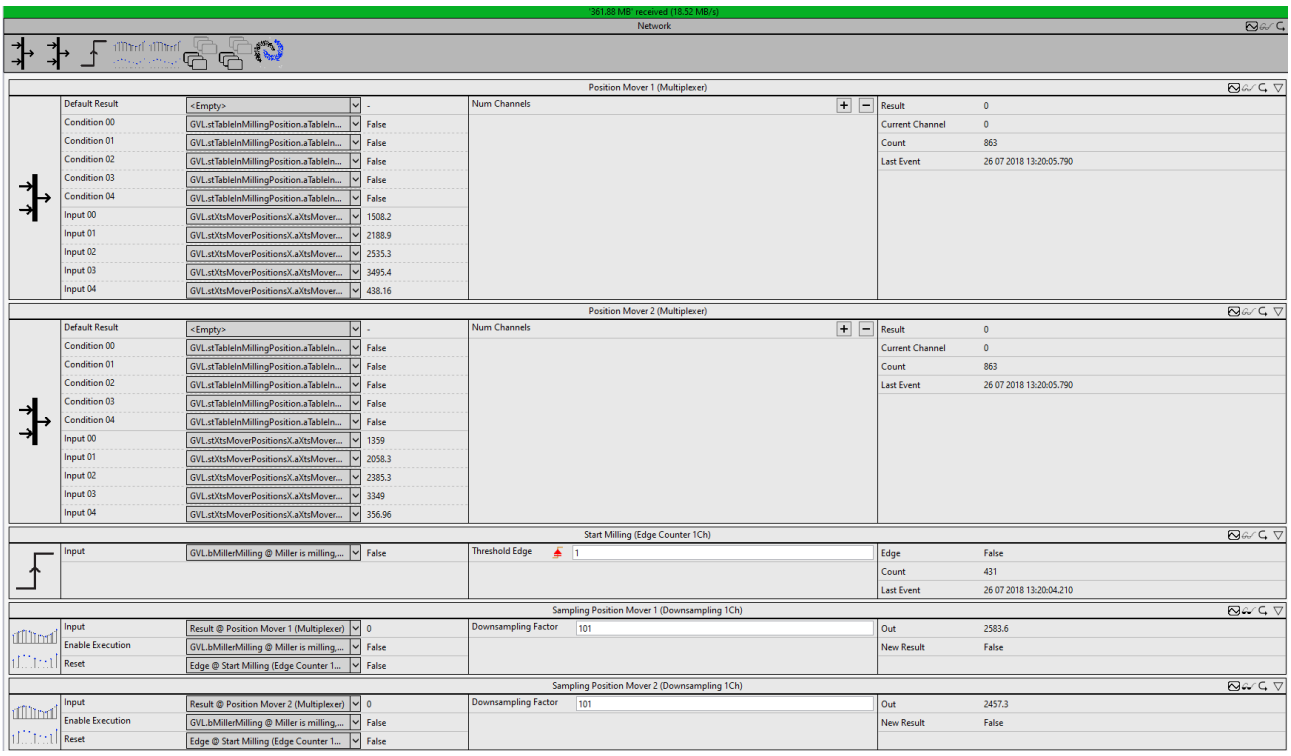
以下は、DenStreamファンクションブロックの例示的なアプリケーションです。

サンプルはピック&プレースのアプリケーションデータに基づいています。ピック&プレースのサンプルをよく理解することが、このアプリケーションを理解するための基礎となります。データへのアクセスも同様に行われます。

このアプリケーションの目的は、フライス加工中の動きに基づいて、フライス加工された形状を楕円形（赤）と丸形（黄）の2つのグループに分けることです。

下の図は、アプリケーションの各ファンクションブロックを示しています。

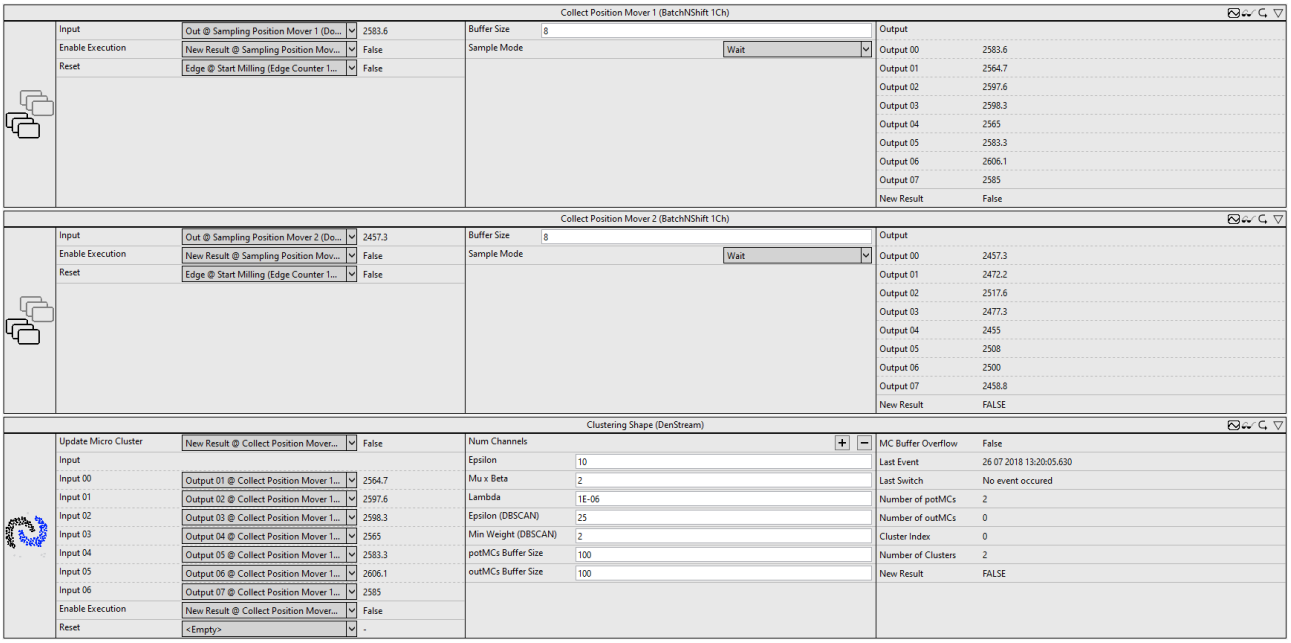
ワークピース用のトレーは、XTSトラック上にそれぞれ2つの可動子で固定されています。そのため、1つのトレーにつき2つの位置値があり、Scopeプロジェクトで加工形状をマッピングするために使用できます。この場合、切削プロセス中の7つの異なるタイミングでサンプリングされるクラスタを特定するには、1つの位置情報だけで十分です。これらの7つの値は、DenStreamファンクションブロックの入力として同時に処理され、7次元の特徴空間を形成します。



Position Mover 1 (Multiplexer) と**Position Mover 2 (Multiplexer)** のファンクションブロックにより、現在加工中のワークトレーの2つの可動子の位置が検出されます。

ファンクションブロックの**Edge Counter (Start Milling)**は、フライス加工開始時に立ち上がりエッジを出力します。これは、後続のファンクションブロックの入力として必要です。

2つのダウンサンプリング用のファンクションブロック（**Sampling Position Mover 1**および**Sampling Position Mover 2**）は、*bMillerMilling*の立ち上がりエッジから始まる切削プロセスを通して、100サイクル（ダウンサンプリング係数）ごとに位置値を保存します。ファンクションブロックは、切削加工が行われている場合のみアクティブになります（*bMillerMilling*== TRUE）。

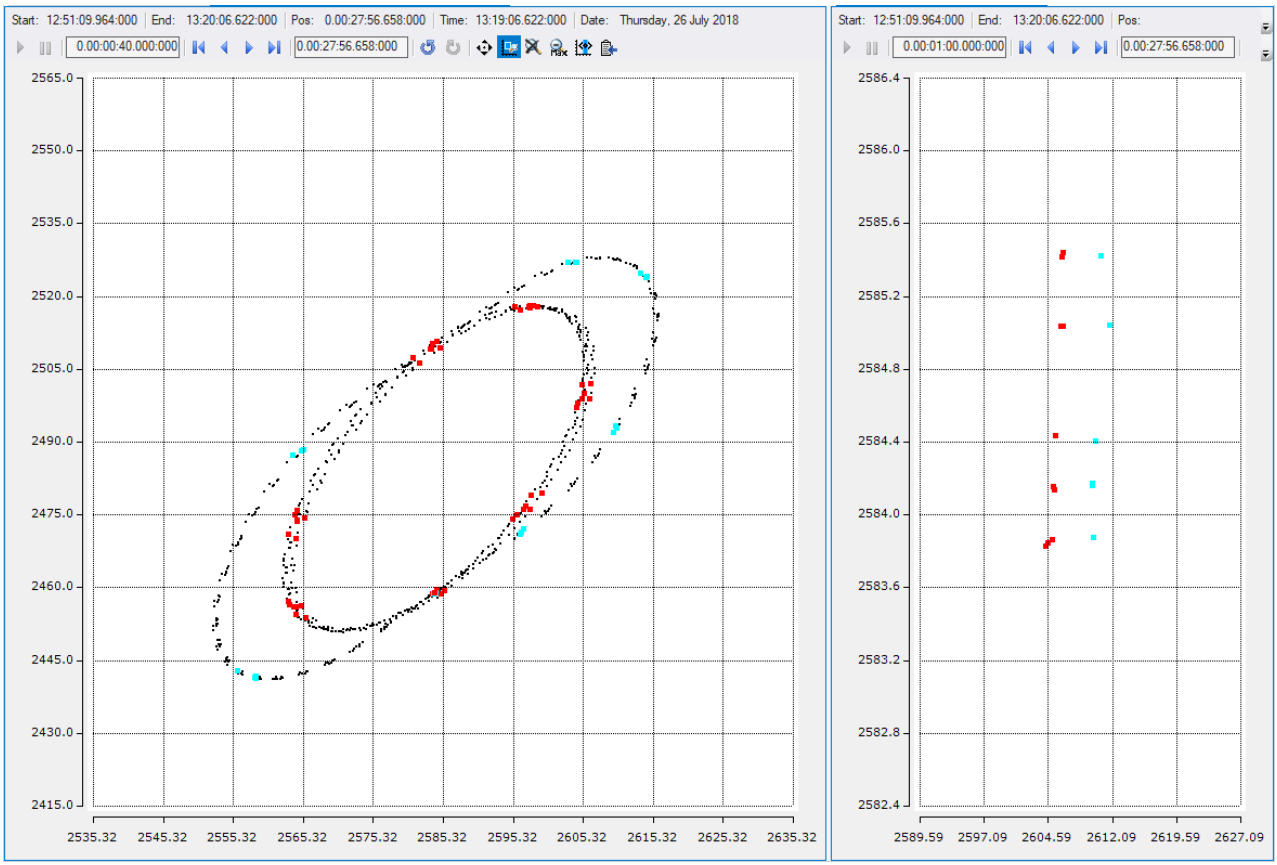


ダウンサンプリングブロックの出力は、それぞれBatchNShiftファンクションブロック（**Collect Position Mover 1**&**Collect Position Mover 2**）の入力にリンクされています。これらはバッチ内の値を要約したものです。バッチは、8つの値の数（*Buffer Size*）に達したときにのみ出力されます。

最後に、1つ目のBatchNShiftの出力は、DenStreamファンクションブロックの入力にリンクされます。フライス加工開始時の位置は加工形状に依存しないため、分類のための情報を含みません。このため、最初の値は破棄されます。

マイクロクラスタは、DownsamplingBufferファンクションブロックがその出力を更新するときにのみ更新されます。*Epsilon*と *Epsilon(DBSCAN)*のパラメータ値を正しく推定するために、異なるフライス形状の入力変数間の距離をプロットで見ることができます。フライス加工形状は時間と共に変化しないため、パラメータ値*Lambda*は非常に低く設定します。0に設定することも可能です。

スコープチャートは解析結果を示しています。



トレーの2つの可動子の位置をXYチャートで示すと、フライス加工中にトレーがたどる移動曲線が表れます。移動曲線は左のプロットで黒く示されています。ここに示した期間中に、両方のフライス形状で複数回加工が行われました。DenStreamファンクションブロックの入力として使用されるポイントはカラーで表示されています。ダイナミックスタイルでは、関連付けられたクラスタインデックスに応じて、ポイントが異なる色で表示されます。クラスタが正しく割り当てられていることがわかります。楕円形のフライス形状を表す点はターコイズブルーで、その他の点は赤で、異なるクラスタ指数が割り当てられています。

クラスタの特徴空間は表示されません。これは7次元なので、直接可視化することができません。2つの特徴（Input 05とInput 06）が右側のプロットに示されています。2つのクラスタのポイントが、互いに空間的に離れていることがわかります。

ダウンロード

- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12177933195.zip

8.3 ピック&プレースアプリケーション：相関（Correlation）

以下は、CorrelationFunctionReference ファンクションブロックの アプリケーション例です。

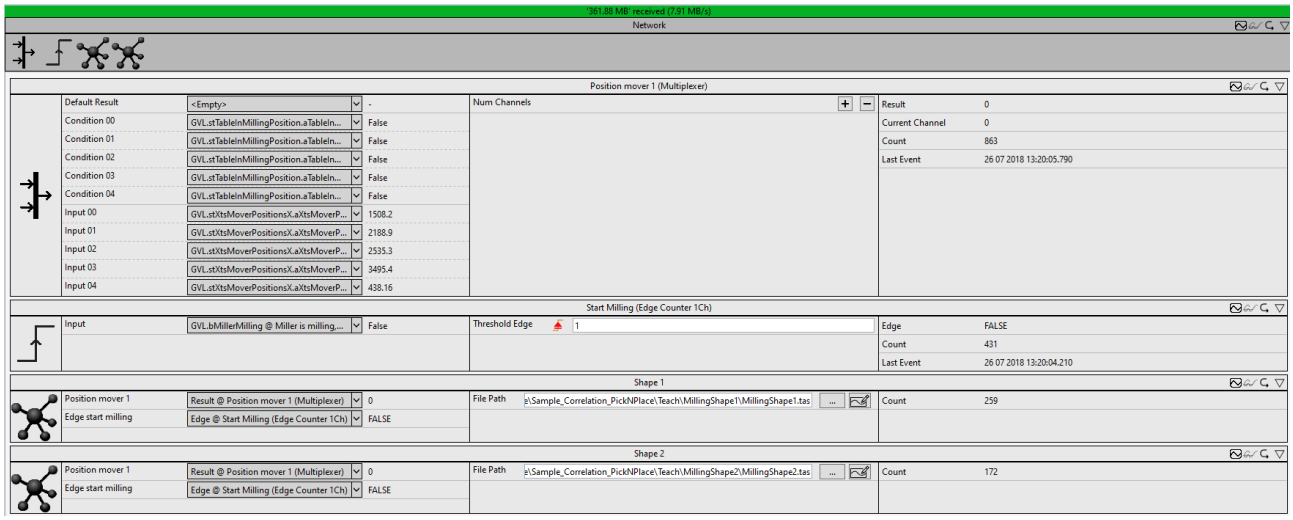
サンプルはピック&プレースのアプリケーションデータに基づいています。ピック&プレースのサンプルをよく理解することが、このアプリケーションを理解するための基礎となります。データへのアクセスも同様に行われます。

このアプリケーションの目的は、フライス加工中の動きに基づいて、フライス加工された形状を楕円形（赤）と丸形（黄）の2つのグループに分けることです。

トレーを取り付けた2つの可動子のうち1つの可動子の動きは、2つの異なるミリング形状について事前に記録され、*MillingShape1.tas*と *MillingShape2.tas*というファイルで入手できます。これらのシーケンスは、それぞれ現在の移動パターンと相関しています。CorrelationFunctionReferenceファンクションブロックが出力する最大係数を用いて、形状が一致するか否かを判定します。

さらに、CorrelationFunctionReferenceファンクションブロックの *Maximize Lag* 出力値は、フライス盤の起動後に形状が追従を開始するまでの遅延を、サイクル数で表して決定します。

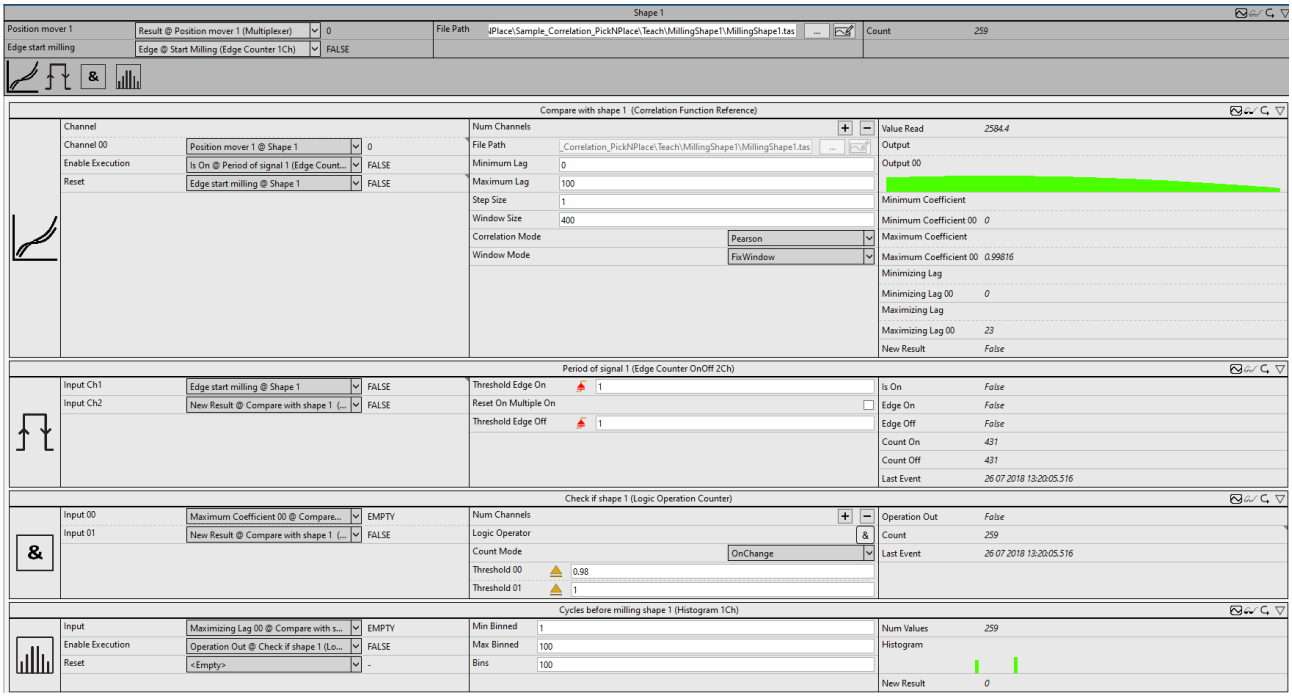
下の図は、アプリケーションの各ファンクションブロックを示しています。



Position Mover 1 (Multiplexer)ファンクションブロックは、ワークが現在加工されているトレーに属する1つ目の可動子の位置を検出するために使用されます。これは、2つの形状のフライス加工プロセス全体にわたってモーションプロファイルが取得できる可動子でもあります。解析のために、トレーの両方の可動部を考慮する必要はありません。可動部は互いに一定の距離で取り付けられており、したがって同じモーションプロファイルを持つためです。

ファンクションブロックの**Edge Counter (Start Milling)**は、フライス加工開始時に立ち上がりエッジを出力します。このエッジは、後続のファンクションブロックの入力として必要です。

相関関係の計算は、各形状について、別のネットワークで個別に実行されます。

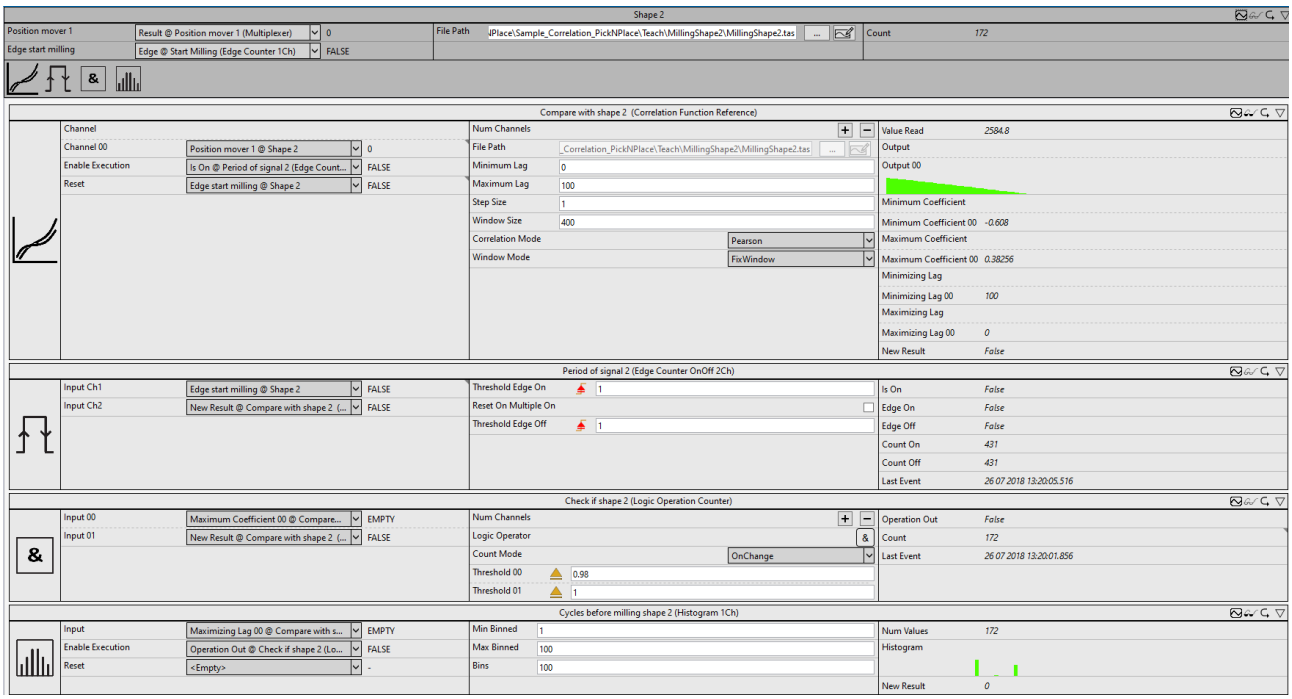


ファンクションブロック**Compare with shape 1 (Correlation Function Reference)**では、可動子の位置(マルチプレクサファンクションブロックの結果出力)が最初の形状の記録信号と相関します。相関計算は、粉砕機のスイッチを入れるところから始まる。これは、*Reset*入力をフライス盤をオンにする立ち上がりエッジ（**Start Milling (Edge Counter 1Ch)**から出力されるエッジ）にリンクさせることで実現されます。これは、このファンクションブロックの立ち上がりエッジごとに相関計算が再スタートするためです。

アプリケーションを起動する前に、参照ファイルのパスを設定する必要があります。リファレンスファイルはプロジェクトディレクトリの*"TeachMillingShape1 "*フォルダにあります。

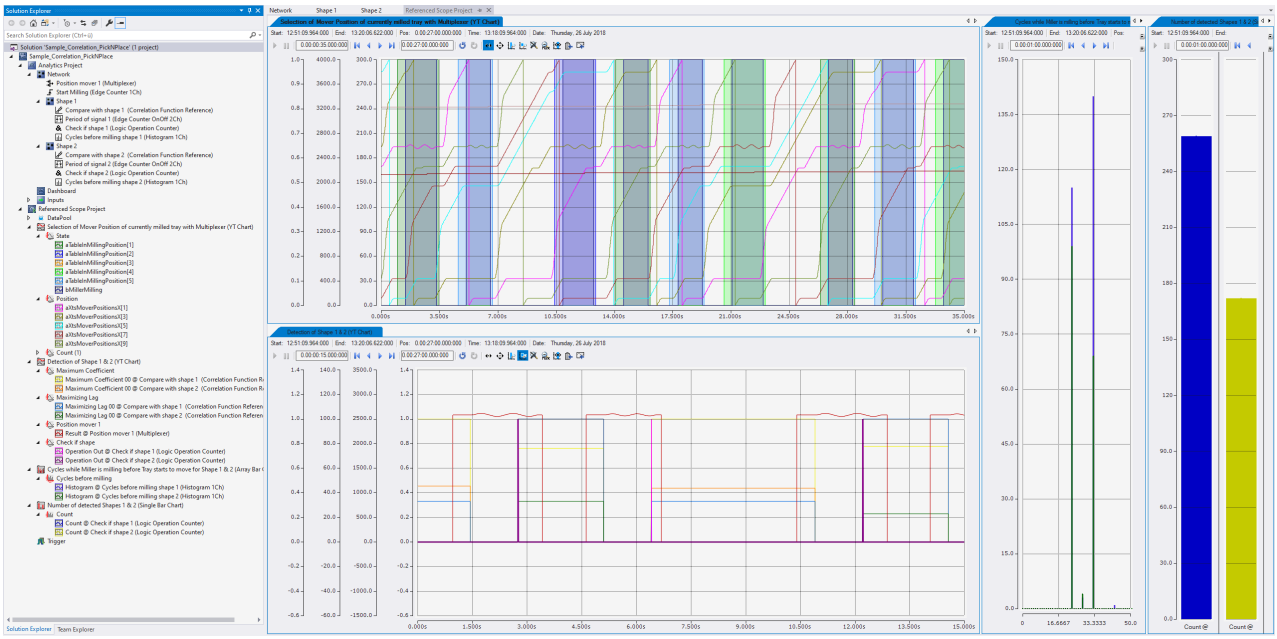
フライス加工中にのみ新しい結果を得るには、相関ファンクションブロックの*Enable Execution*入力をリンクします。ここでは、後続ファンクションブロック **Period of Signal 1 (Edge Counter OnOff 2Ch** の*Is On*出力がリンクされています。これは、ミラーがオンになったときに TRUE になり、ファイルからの全ての値が相関計算に一度使用され、相関ファンクションブロックがその出力を更新した場合に再び FALSE になります。**Check if shape 1 (Logic Operation Counter)**ファンクションブロックは、シーケンスを比較した結果、最大ピアソン相関係数が0.98より大きい場合にTRUEを返します。こうすることで、形状1のフライス加工部品の数もカウントされます。

こうして最初の形状に割り当てられたフライス加工手順について、ファンクションブロック**Cycles before Milling Shape 1 (Histogram 1Ch)**は、フライスのスイッチを入れてから輪郭に沿って駆動するまでに経過したサイクル数の分布を決定します。



第2のフライス形状の相関計算は、別のネットワークで同様に行われます。

以下のスコーププロットは、解析結果を可視化したものです。



左上の図は、マルチプレクサの入力と結果 出力を示しています。対応するトレーが粉砕機の位置にあるとき、可動子は正弦波状の経路をたどることがわかる。

左下の図では、マルチプレクサの結果出力が赤で示されています。2つのファンクションブロック**Check if shape 1 (Logic Operation Counter)**と**Check if shape 2 (Logic Operation Counter)**は、それぞれ、相関シーケンスが0.98を超える最大相関係数（黄色およびオレンジ）に達した場合、ピーク（ピンクおよび紫）を示します。表示したセクションで、これらの結果がマルチプレクサの*Result*出力のカーブと一致していることを確認できます。また、相関ファンクションブロックの対応する*Maximizing Lag*出力が青と緑で示されています。

2つの棒グラフの左には、相関ファンクションブロックの*Maximizing Lag*出力を示しています。これらは、ヒストグラムファンクションブロックの結果から生じます。どちらの形状も、ラグ23およびラグ33で大きなたわみを、ラグ28で小さなたわみを生じます。最初の形状は、43サイクルという特有のラグを生じます。

右側の棒グラフは、形状1と形状2に割り当てられたワークの数を示しています。形状1には259個のワークが、形状2には172個のワークが割り当てられています。

ダウンロード

- https://infosys.beckhoff.com/content/1033/TE3500_TC3_Analytics_Workbench/Resources/12177935499.zip

9 付録

9.1 FAQ - よくある質問

本項では、TwinCAT Analytics Workbenchでの作業に役立つ、よくある質問を掲載しています。ご不明な点がございましたら、サポートチーム（support@beckhoff.com）までお問い合わせください。

1. TwinCAT計測製品には、オープンソースのソフトウェアコンポーネントが使用されていますか？ [▶ 558]

TwinCAT計測製品には、オープンソースのソフトウェアコンポーネントが使用されていますか？

はい、様々なオープンソースのコンポーネントが使用されています。以下のディレクトリから、一覧とライセンス要件を確認できます。*...¥TwinCAT¥¥Functions¥TwinCAT Measurement¥Legal*.

Trademark statements

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.

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.

詳細はこちら:
www.beckhoff.com/te3500

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
+49 5246 9630
info@beckhoff.com
www.beckhoff.com