Modbus Plusドライバー

目次

Modbus Plusドライバー

ヘルプバージョン 1.063

目次

概要

Modbus Plusドライバーは Modbus Plus デバイスが HMI、SCADA、Historian、MES、ERP や多数のカスタムアプリケーションを含む OPC クライアントアプリケーションに接続するための信頼性の高い手段を提供します。これは、Modicon SA85 または Schneider PCI-85 インタフェースカードで使用するためのものです。このドライバーは同じコンピュータに SA85/PCI-85 カードが存在する構成はサポートしていません。

外部依存

このドライバーには外部依存があります。

Modbus Plus ネットワークと通信するためには SA85 または PCI-85 インタフェースアダプタとそのデバイスドライバーソフトウェア (MBPLUS または MBX ドライバー) が必要です。このインタフェースアダプタとデバイスドライバーは Modicon/Schneider から購入できます。サーバーは SA85 または PCI-85 カードにつき最大 8 つのチャネルをサポートできます。

設定

このドライバーでは、「Modbus サーバー」と「非送信請求」という用語は同じ意味になります。

チャネルとデバイスの制限値

このドライバーでサポートされているチャネルの最大数は 32 です。このドライバーでサポートされているデバイスの最大数は、1 つのチャネルにつき 8192 です。
Modbus Plusドライバーでは、SA85 カードアダプタあたり最大 8 つのチャネルがサポートされています。

サポートされる インタフェースカード

SA85 カード

注記: ユーザーは USB アダプタを介して Modbus RTU シリアルマシンから Modbus Plus ネットワークに接続することもできます。

SA85 カードの要件については、*外部依存*を参照してください。

サポートされる通信モード

Modbus Plusドライバーは3つの通信モードをサポートしており、これを使用してデバイスからデータが取得されます。モードはデバイスIDフォーマットによって指定されます。オプションには「送信請求」、「非送信請求」、「メールボックス」があります。モードの説明は次のとおりです。

- **送信請求**: このモードでは、ドライバーはデバイスに対するデータの読み取り/書き込み要求を生成します。使用可能なアドレスには出力コイル、入力コイル、内部レジスタ、保持レジスタが含まれます。出力コイルと保持レジスタのアドレスには読み取り/書き込みのアクセスが可能であり、入力コイルと内部レジスタでは読み取り専用のアクセスが可能です。送信請求モードでのデバイスIDのフォーマットは *DM.r1.r2.r3.r4.r5* または *S.r1.r2.r3.r4.r5* です。
- **非送信請求**: このモードでは、ドライバーはネットワーク上の仮想PLCとして機能します。読み取りと書き込みはドライバーからは行われません。非送信請求デバイスに対して読み書きを行うクライアントアプリケーションは、物理デバイスではなく、そのデバイスに割り当てられているローカルキャッシュにデータを渡します。このローカルキャッシュはそのドライバーを実行しているPC内にあります。ネットワーク上のデバイスは非送信請求コマンドを介して同じキャッシュに対して読み書きを行います。非送信請求モードでのデバイスIDのフォーマットは *DS.r1.r2.r3.r4.r5* です。
- **メールボックス**: このモードでは、ドライバーは定義されている各メールボックスデバイスのストレージ領域として機能します。ドライバーは非送信請求コマンドを受信すると、メッセージの送信元のルーティングパスを検出し、そのデバイスに割り当てられているストレージ領域内にデータを配置します。ルーティングパスがメールボックスデバイスとして定義されていないデバイスからメッセージが送信されている場合、そのメッセージは処理されません。メールボックスデバイスから読み取るクライアントアプリケーションは、物理デバイスではなく、そのドライバー内のストレージ領域から読み取りますが、書き込みは物理デバイスとローカルキャッシュの両方に送信されます。このモードでは保持レジスタのみがサポートされます。Doubleデータ型はサポートされません。メールボックスモードでのデバイスIDのフォーマットは *U.r1.r2.r3.r4.r5* です。
  注記: 非送信請求メールボックスコマンドは、特定のModiconデバイスで使用可能なMSTR命令によって可能になります。詳細については、デバイスIDで「例2 - 単一ネットワークのメールボックスモード」および「例3 - ブリッジネットワークのメールボックスモード」を参照してください。

SA85インタフェースカードによってサポートされる通信モードについては、以下の表を参照してください。

モード	SA85カード
送信請求	x
非送信請求	x
メールボックス	x

詳細については、*デバイスID*を参照してください。

複数のデバイスのポーリング

Modbus Plusドライバーは、Modbus Plusネットワーク上の複数のデバイスをポーリングすることができます。また、Modbus Plusネットワーク上にある、その他のデバイスにとってポーリングの対象となる単一のModbusサーバーデバイスとして機能することもできます。このドライバーでは、デバイスは8192台に制限されており、最大4つのアダプタがサポートされます。

チャネルのプロパティ - 一般

このサーバーでは、複数の通信ドライバーを同時に使用することができます。サーバープロジェクトで使用される各プロトコルおよびドライバーをチャネルと呼びます。サーバープロジェクトは、同じ通信ドライバーまたは一意の通信ドライバーを使用する多数のチャネルから成ります。チャネルは、OPCリンクの基本的な構成要素として機能します。このグループは、識別属性や動作モードなどの一般的なチャネルプロパティを指定するときに使用します。

識別

「名前」: このチャネルのユーザー定義識別情報を指定します。各サーバープロジェクトで、それぞれのチャネル名が一意でなければなりません。名前は最大 256 文字ですが、一部のクライアントアプリケーションではOPC サーバーのタグ空間をブラウズする際の表示ウィンドウが制限されています。チャネル名はOPC ブラウザ情報の一部です。チャネルの作成にはこのプロパティが必要です。
予約済み文字の詳細については、サーバーのヘルプで「チャネル、デバイス、タグ、およびタググループに適切な名前を付ける方法」を参照してください。

「説明」: このチャネルに関するユーザー定義情報を指定します。
「説明」などのこれらのプロパティの多くには、システムタグが関連付けられています。

「ドライバー」: このチャネル用のプロトコル/ドライバーを指定します。チャネル作成時に選択されたデバイスドライバーを指定します。チャネルのプロパティではこの設定を変更することはできません。チャネルの作成にはこのプロパティが必要です。
注記: サーバーがオンラインで常時稼働している場合、これらのプロパティをいつでも変更できます。これには、クライアントがデータをサーバーに登録できないようにチャネル名を変更することも含まれます。チャネル名を変更する前にクライアントがサーバーからアイテムをすでに取得している場合、それらのアイテムは影響を受けません。チャネル名が変更された後で、クライアントアプリケーションがそのアイテムを解放し、古いチャネル名を使用して再び取得しようとしても、そのアイテムは取得されません。大規模なクライアントアプリケーションを開発した場合は、プロパティを変更しないようにしてください。オペレータがプロパティを変更したりサーバーの機能にアクセスしたりすることを防ぐため、適切なユーザー役割を使用し、権限を正しく管理する必要があります。

診断

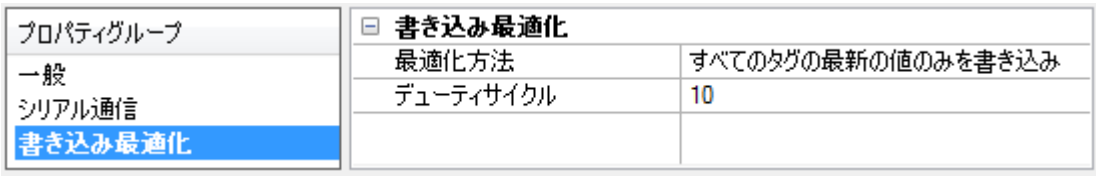
「診断取り込み」: このオプションが有効な場合、チャネルの診断情報がOPC アプリケーションに取り込まれます。サーバーの診断機能は最小限のオーバーヘッド処理を必要とするので、必要なときにだけ利用し、必要がないときには無効にしておくことをお勧めします。デフォルトでは無効になっています。
注記: ドライバーで診断機能がサポートされていない場合、このプロパティは使用できません。
詳細については、サーバーのヘルプで「通信診断」を参照してください。

タグ数

「静的タグ」: デバイスレベルまたはチャネルレベルで定義される静的タグの数を指定します。この情報は、トラブルシューティングと負荷分散を行う場合に役立ちます。

チャネルのプロパティ - 書き込み最適化

サーバーは、クライアントアプリケーションから書き込まれたデータをデバイスに遅延なく届ける必要があります。このため、サーバーに用意されている最適化プロパティを使用して、特定のニーズを満たしたり、アプリケーションの応答性を高めたりすることができます。



書き込み最適化

「最適化方法」: 基礎となる通信ドライバーに書き込みデータをどのように渡すかを制御します。以下のオプションがあります。

- **「すべてのタグのすべての値を書き込み」**: このオプションを選択した場合、サーバーはすべての値をコントローラに書き込もうとします。このモードでは、サーバーは書き込み要求を絶えず収集し、サーバーの内部書き込みキューにこれらの要求を追加します。サーバーは書き込みキューを処理し、デバイスにできるだけ早くデータを書き込むことによって、このキューを空にしようとします。このモードでは、クライアントアプリケーションから書き込まれたすべてのデータがターゲットデバイスに送信されます。ターゲットデバイスで書き込み操作の順序または書き込みアイテムのコンテンツが一意に表示される必要がある場合、このモードを選択します。
- **「非 Boolean タグの最新の値のみを書き込み」**: デバイスにデータを実際に送信するのに時間がかかっているために、同じ値への多数の連続書き込みが書き込みキューに累積することがあります。書き込みキューにすでに置か

れている書き込み値をサーバーが更新した場合、同じ最終出力値に達するまでに必要な書き込み回数ははるかに少なくなります。このようにして、サーバーのキューに余分な書き込みが累積することがなくなります。ユーザーがスライドスイッチを動かすのをやめると、ほぼ同時にデバイス内の値が正確な値になります。モード名からもわかるように、Boolean 値でない値はサーバーの内部書き込みキュー内で更新され、次の機会にデバイスに送信されます。これによってアプリケーションのパフォーマンスが大幅に向上します。

注記: このオプションを選択した場合、Boolean 値への書き込みは最適化されません。モーメンタリプッシュボタンなどの Boolean 操作で問題が発生することなく、HMI データの操作を最適化できます。

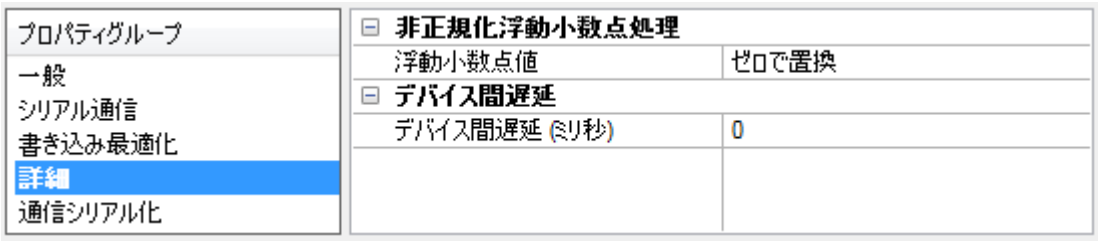
- **「すべてのタグの最新の値のみを書き込み」**: このオプションを選択した場合、2 つ目の最適化モードの理論がすべてのタグに適用されます。これはアプリケーションが最新の値だけをデバイスに送信する必要がある場合に特に役立ちます。このモードでは、現在書き込みキューに入っているタグを送信する前に更新することによって、すべての書き込みが最適化されます。これがデフォルトのモードです。

「デューティサイクル」: 読み取り操作に対する書き込み操作の比率を制御するときに使用します。この比率は必ず、読み取り 1 回につき書き込みが 1 から 10 回の間であることが基になっています。デューティサイクルはデフォルトで 10 に設定されており、1 回の読み取り操作につき 10 回の書き込みが行われます。アプリケーションが多数の連続書き込みを行っている場合でも、読み取りデータを処理する時間が確実に残っている必要があります。これを設定すると、書き込み操作が 1 回行われるたびに読み取り操作が 1 回行われるようになります。実行する書き込み操作がない場合、読み取りが連続処理されます。これにより、連続書き込みを行うアプリケーションが最適化され、データの送受信フローがよりバランスのとれたものとなります。

注記: 本番環境で使用する前に、強化された書き込み最適化機能との互換性が維持されるようにアプリケーションのプロパティを設定することをお勧めします。

チャネルのプロパティ - 詳細

このグループは、チャネルの詳細プロパティを指定するときに使用します。すべてのドライバーがすべてのプロトコルをサポートしているわけではないので、サポートしていないデバイスには詳細グループが表示されません。



「非正規化浮動小数点処理」: 非正規化値は無限、非数 (NaN)、または非正規化数として定義されます。デフォルトは「ゼロで置換」です。ネイティブの浮動小数点処理が指定されているドライバーはデフォルトで「未修正」になります。「非正規化浮動小数点処理」では、ドライバーによる非正規化 IEEE-754 浮動小数点データの処理方法を指定できます。オプションの説明は次のとおりです。

- **「ゼロで置換」**: このオプションを選択した場合、ドライバーが非正規化 IEEE-754 浮動小数点値をクライアントに転送する前にゼロで置き換えることができます。
- **「未修正」**: このオプションを選択した場合、ドライバーは IEEE-754 非正規化、正規化、非数、および無限の値を変換または変更せずにクライアントに転送できます。

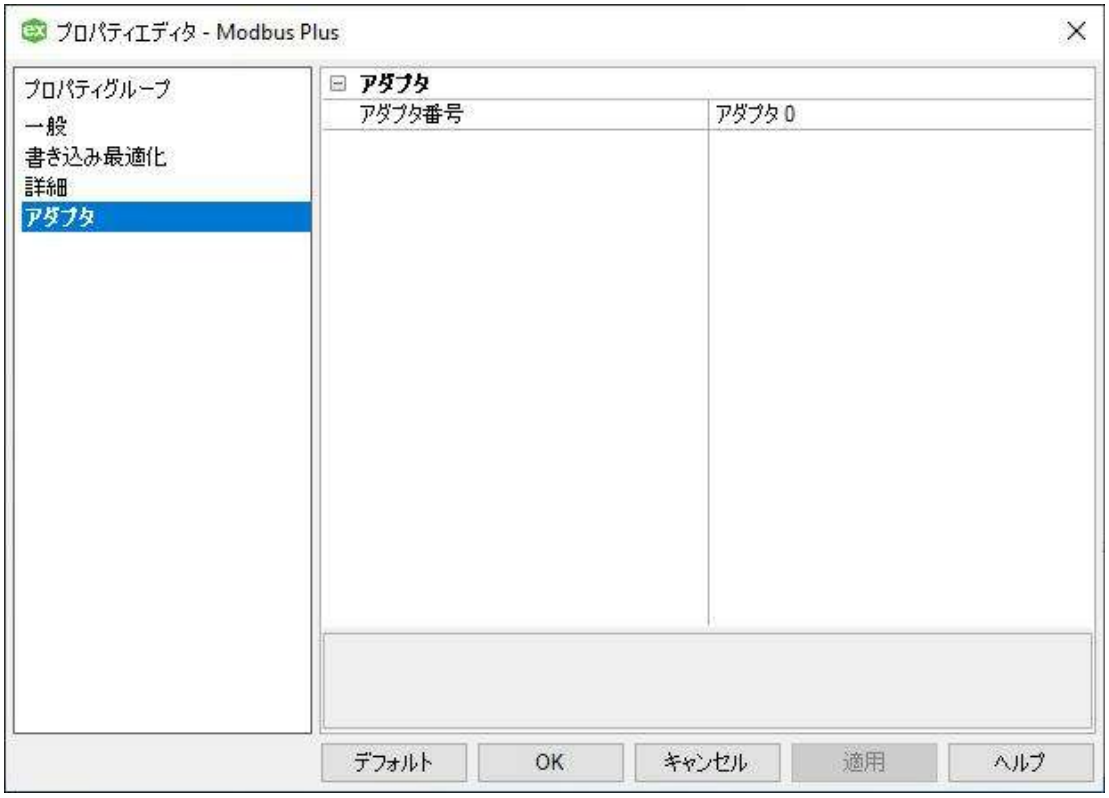
注記: ドライバーが浮動小数点値をサポートしていない場合や、表示されているオプションだけをサポートする場合、このプロパティは無効になります。チャネルの浮動小数点正規化の設定に従って、リアルタイムのドライバータグ (値や配列など) が浮動小数点正規化の対象となります。たとえば、EFM データはこの設定の影響を受けません。

浮動小数点値の詳細については、サーバーのヘルプで「非正規化浮動小数点値を使用する方法」を参照してください。

「デバイス間遅延」: 通信チャネルが同じチャネルの現在のデバイスからデータを受信した後、次のデバイスに新しい要求を送信するまで待機する時間を指定します。ゼロ (0) を指定すると遅延は無効になります。

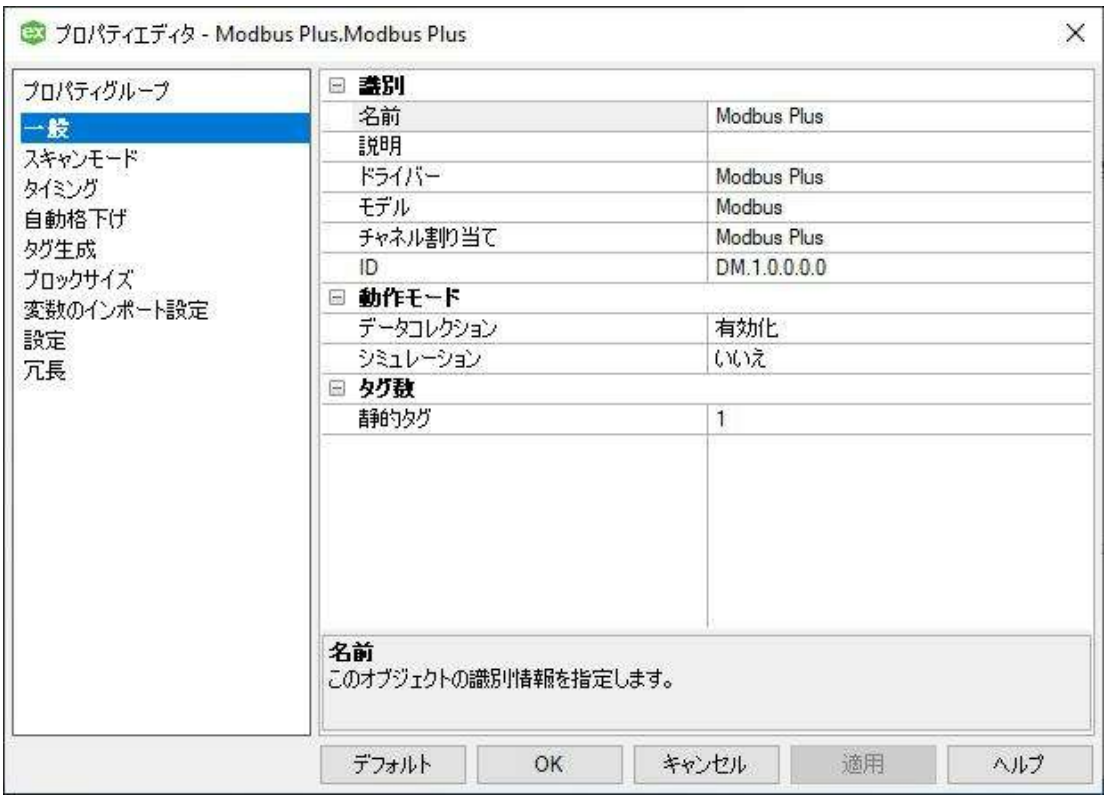
注記: このプロパティは、一部のドライバー、モデル、および依存する設定では使用できません。

チャネルのプロパティ - アダプタ



「アダプタ番号」: Modbus Plus カードによって使用されるアダプタの番号を指定します。有効なアダプタ番号は 0 から 3 です。カードの詳細については、***設定***を参照してください。

デバイスのプロパティ - 一般



識別

「名前」: このデバイスのユーザー定義の識別情報。

「説明」: このデバイスに関するユーザー定義の情報。

「チャネル割り当て」: このデバイスが現在属しているチャネルのユーザー定義の名前。

「ドライバー」: このデバイスに設定されているプロトコルドライバー。

「モデル」: このデバイスのバージョン。

「ID」: ネットワーク上のノードへのパスを指定します。デバイス ID はネットワーク上のノードへのパスを示します。これは 5 つの連続するルーティングバイトとモード指定子から成ります。
詳細については、デバイス ID を参照してください。

動作モード

「データコレクション」: このプロパティでは、デバイスのアクティブな状態を制御します。デバイスの通信はデフォルトで有効になっていますが、このプロパティを使用して物理デバイスを無効にできます。デバイスが無効になっている場合、通信は試みられません。クライアントから見た場合、そのデータは無効としてマークされ、書き込み操作は許可されません。このプロパティは、サーバーヘルプ内のこのプロパティまたはデバイスのシステムタグを使用していつでも変更できます。

「シミュレーション」: このオプションは、デバイスをシミュレーションモードにします。このモードでは、ドライバーは物理デバイスとの通信を試みませんが、サーバーは引き続き有効な OPC データを返します。シミュレーションモードではデバイスとの物理的な通信は停止しますが、OPC データは有効なデータとして OPC クライアントに返されます。シミュレーションモードでは、サーバーはすべてのデバイスデータを自己反映的データとして扱います。つまり、シミュレーションモードのデバイスに書き込まれたデータはすべて再び読み取られ、各 OPC アイテムは個別に処理されます。アイテムのメモリマップはグループ更新レートに基づきます。(サーバーが再初期化された場合などに) サーバーがアイテムを除去した場合、そのデータは保存されません。デフォルトは「いいえ」です。

注記:

1. システムタグ (_Simulated) は読み取り専用であり、ランタイム保護のため、書き込みは禁止されています。このシステムタグを使用することで、このプロパティをクライアントからモニターできます。
2. シミュレーションモードでは、アイテムのメモリマップはクライアントの更新レート (OPC クライアントではグループ更新レート、ネイティブおよび DDE インタフェースではスキャン速度) に基づきます。つまり、異なる更新レートで同じアイテムを参照する 2 つのクライアントは異なるデータを返します。

シミュレーションモードはテストとシミュレーションのみを目的としています。本番環境では決して使用しないでください。

デバイス ID

デバイス ID はネットワーク上のノードへのパスを示します。これはモード指定子と 5 つの連続するルーティングバイトから成ります。モードは、データクライアント (DM)、データサーバー (DS)、メールボックスです。このドライバーでは、「Modbus サーバー」と「非送信請求」という用語は同じ意味になります。

送信請求モード (データクライアント)

データクライアントパスは DM または S というプレフィックスで始まり、ネットワーク上の別のノードとの通信に使用されます。このタイプの通信では、ホストシステムが Modbus クライアントとして機能します。DM パスには Modbus の読み取り/書き込みコマンドに応答可能な PLC やその他のデバイス (Modbus Plus ドライバーを実行している別のホストを含む) を指定できます。DM パスのフォーマットは *DM.r1.r2.r3.r4.r5* または *S.r1.r2.r3.r4.r5* です。

非送信請求モード (データサーバー)

1 つの SA85 カードにつき 1 つのデータサーバーパスを設定することができます。パスのフォーマットは *DS.1.0.0.0.0* です。サーバーパスを定義することにより、Modbus Plus ドライバーが稼働しているホスト上で、ほかのデバイスからの読み取り/書き込み要求に応答可能なネットワーク上のデバイスをシミュレートすることができます。その他のデバイスはこのシミュレーション対象のデバイスへのデータクライアントパスを開くことによってこのデバイスと通信できます。

シミュレーション対象のデバイスは、Modbus バイトオーダーを使用します。32 ビット値と 64 ビット値のデータエンコーディングでは、最初の Word が DWord の下位 Word となり、64 ビット値では最初の DWord が下位 DWord となります。したがって、非送信請求デバイスのデータエンコーディングオプションは次のように設定する必要があります。

- 「Modbus バイトオーダー」
- 「最初の Word を下位とする」
- 「最初の DWord を下位とする」

*詳細については、**設定**を参照してください。*

1 から 65536 のアドレスは出力コイル、入力コイル、内部レジスタ、保持レジスタ用に実装されています。ドライバーは外部デバイスからのこれらの値を読み書きする有効なすべての要求に応答します (ファンクションコード [10 進] 01、02、03、04、05、06、15、16)。これらの位置には、Modbus サーバーデバイスに割り当てられたタグとしてホスト PC からローカルにアクセスすることもできます。非送信請求デバイスでは書き込み専用アクセスは許可されません。

Modbus サーバーパスが有効になっている場合、Modbus Plus ドライバーにより、各 SA85 カードで 8 つの Modbus サーバーパスが有効になります。これにより、リモート PLC とその他の Modbus Plus デバイスは、8 つの Modbus サーバーパスのいずれかを使用して、このドライバーの Modbus サーバーメモリにアクセスできるようになります。どの場合もアクセス先のメモリは同じです。MSTR 命令として、ユーザーはこのドライバーがサービスを提供する SA85 カードで接続するパスを定義する際にパス 1 から 8 を指定できます。これは多数のリモートデバイスが PC にデータを送信する用途で役立ちます。その場合、8 つの Modbus サーバーパスを利用して、リモートノードからの負荷を分散することができます。このドライバーの各 Modbus サーバーパスでは、専用の実行スレッドによって非常に高いパフォーマンスを実現します。

プロジェクトで Modbus サーバーデバイスが定義されていない場合、ドライバーは受信した非送信請求読み取り/書き込み要求をすべて無視します。

メールボックスモード

メールボックスパスは U というプレフィックスで始まり、物理デバイスへのパスを示します。プロジェクトで定義されている Modbus サーバーデバイス内のこの物理デバイスにストレージ領域が割り当てられます。物理デバイスはこのストレージ領域に非送信請求書き込みを送信しますが、これらの書き込みには、Modbus サーバーデバイスに割り当てられたタグとしてホスト PC からローカルにアクセスすることもできます。メールボックスパスのフォーマットは *U.r1.r2.r3.r4.r5* です。

ドライバーは非送信請求メールボックスデータを受信する際に、必ず Modbus サーバーパスを開きます。ドライバーが開くパスは *DS.1.0.0.0.0* です。同じ Modbus Plus ネットワーク上のデバイスは、DM でデータクライアントのパスを開くことによ

り、ドライバーとの通信を行います。<ローカルノード> .1.0.0.0 と指定した場合、ホストコンピュータの SA85 カードで設定されているアドレスがローカルノードになります。ブリッジネットワーク上のデバイスが使用するパスについては、例 3 を参照してください。

デバイスは Modbus Plus の MSTR 命令を使用してドライバーにデータを提供します。ドライバーがデータを特定のデバイスと関連付けることができるように、デバイス ID パスが受信データの最初の 5 つのレジスタに埋め込まれている必要があります。データの最初の 5 つのレジスタがプロジェクト内のデバイスのデバイス ID パスと一致しない場合、受信データは破棄されます。MSTR 命令では書き込みコマンドのみがサポートされています。

注記:

1. デバイス ID パスは、ホスト PC へのデバイスパスではなく、ホスト PC からデバイスへのパスに埋め込まれます。
2. TIO モジュールデバイスでは、Modbus サーバーネットワークアドレスはサポートされていません。
3. OPC クライアントが接続している間はデバイス ID を変更してはなりません。変更した場合、すべての OPC クライアントを切断してから再び接続するまでその変更は有効になりません。

例 1 - 送信請求モード

1 つのネットワークが 4 つのノードから構成され、ノード 1 と 4 は Modbus Plus ドライバーを使用するソフトウェアを実行しているホスト PC です。ノード 2 と 3 は PLC です。次の表は、このネットワークの各ノードを起点とするアドレス指定を示しています。

起点	宛先ノード 1	宛先ノード 2	宛先ノード 3	宛先ノード 4
ノード 1	----------	DM.2.0.0.0.0	DM.3.0.0.0.0	DM.4.1.0.0.0
ノード 2	DM.1.1.0.0.0	----------	DM.3.0.0.0.0	DM.4.1.0.0.0
ノード 3	DM.1.1.0.0.0	DM.2.0.0.0.0	----------	DM.4.1.0.0.0
ノード 4	DM.1.1.0.0.0	DM.2.0.0.0.0	DM.3.0.0.0.0	----------

注記: ホスト PC 上のシミュレーション対象のデバイスにアクセスするには、パスのゼロでない最後の数値が必ず 1 でなければなりません。この値は、ドライバーで使用されている Modbus サーバーパスを示しています。

例 2 - メールボックスモード 単一ネットワーク

デバイスのレジスタ 40020 から 40029 をホスト PC の位置 40001 から 40010 に転送します。制御ブロックの位置は異なる場合があります。ホスト PC のアドレスは 7.0.0.0.0 です。デバイスのアドレスは 3.0.0.0.0 です。

MSTR 命令

制御ブロック	40001	-
データ領域	40015	5 つ前のレジスタから開始します
長さ	15	実際のデータより 5 多い

制御ブロック

レジスタ	コンテンツ	説明
40001	1	書き込み操作
40002	0	エラーコードが格納されます
40003	15	転送するレジスタの数
40004	1	ホスト PC での開始位置 (レジスタ 40001)
40005	7	ホスト PC へのパス
40006	1	ホスト PC へのパス
40007	0	ホスト PC へのパス
40008	0	ホスト PC へのパス
40009	0	ホスト PC へのパス

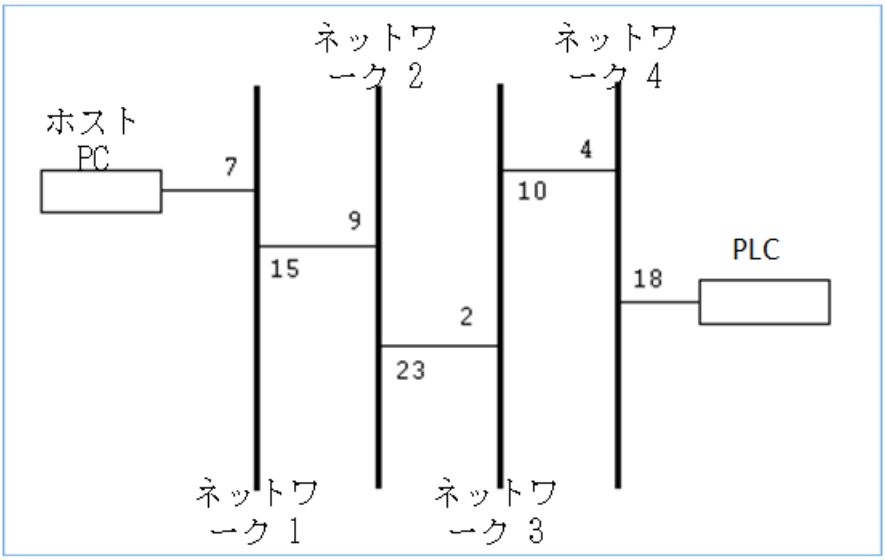
データ領域

レジスタ	コンテンツ	説明
40015	3	ホスト PC からデバイスへのパス (デバイス ID)
40016	0	ホスト PC からデバイスへのパス
40017	0	ホスト PC からデバイスへのパス
40018	0	ホスト PC からデバイスへのパス
40019	0	ホスト PC からデバイスへのパス
40020	-	実際のデータ先頭
40029	-	実際のデータ末尾

非送信請求メッセージを受信すると、ドライバーは以下を行います。

1. メッセージを理解した場合、ドライバーは送信元デバイスに肯定応答を送信します。**複数レジスタをプリセット** (コード 0x10) 以外のファンクションメッセージを受信した場合、ドライバーはファンクションが実装されていないことを示す応答を返します。「複数レジスタをプリセット」は MSTR 命令の受信側にあるデバイスで使用されるファンクションコードです。メッセージが理解されないか不完全である場合、ドライバーは例外応答を返します。
2. ドライバーは受信データの最初の 5 つのレジスタをプロジェクト内のデバイスのデバイス ID パスと照合します。何も見つからない場合、データは破棄されます。データが 6 レジスタ未満である場合、ただちに破棄されます。
3. ドライバーは受信データの 6 番目のレジスタから開始するデータの n - 5 個のレジスタを (メッセージで示されている位置から開始する) デバイスの内部で管理されているイメージマップにコピーします。これがこれらの位置で受信する最初のデータである場合、ドライバーはイメージマップにストレージを割り当てる必要があります。
4. ドライバーのクライアントがこのデータを使用可能になります。この例に示すデータは、デバイス ID が U.3.0.0.0.0 であるデバイスのアドレス 40001 から 40009 を表すタグとして参照されます。クライアントはこのデバイスがプロジェクトで作成されたときに割り当てされた論理名を使用してデバイスを参照します。データは 40001[10] や 40001[2][5] などの配列として参照することもできます。

例 3 - メールボックスモード ブリッジネットワーク



PLC から見たホスト PC のアドレスは 4.2.9.7.1 です。ホスト PC から見た PLC のアドレスは 15.23.10.18.0 です。これはデバイス ID のパスです。同じレジスタが PLC からホスト PC 内の同じ位置に転送された場合、MSTR 命令では以下の表に従って制御ブロックとデータ領域が使用されます。メッセージは同様に処理されます。

注記: このドライバーを使用している場合、ホスト PC をデバイスから最大で 3 つ離れたネットワークに配置できます。

MSTR 命令

制御ブロック	40001	
データ領域	40015	5 つ前のレジスタから開始します。
長さ	15	実際のデータより 5 多い。

制御ブロック

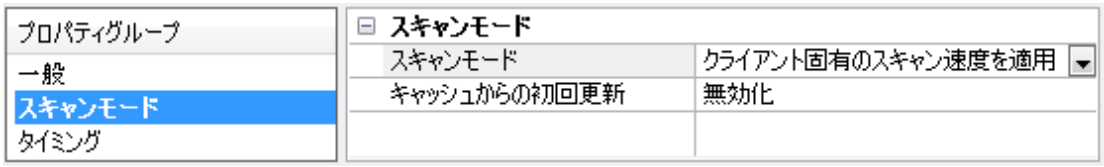
レジスタ	コンテンツ	説明
40001	1	書き込み操作
40002	0	エラーコードが格納されます
40003	15	転送するレジスタの数
40004	1	ホスト PC での開始位置 (レジスタ 40001)
40005	4	ホスト PC へのパス
40006	2	ホスト PC へのパス
40007	9	ホスト PC へのパス
40008	7	ホスト PC へのパス
40009	1	ホスト PC へのパス

データ領域

レジスタ	コンテンツ	説明
40015	15	ホスト PC からデバイスへのパス (デバイス ID)
40016	23	ホスト PC からデバイスへのパス
40017	10	ホスト PC からデバイスへのパス
40018	18	ホスト PC からデバイスへのパス
40019	0	ホスト PC からデバイスへのパス
40020	-	実際のデータ先頭
40029	-	実際のデータ末尾

デバイスのプロパティ - スキャンモード

「スキャンモード」では、デバイスとの通信を必要とする、サブスクリプション済みクライアントが要求したタグのスキャン速度を指定します。同期および非同期デバイスの読み取りと書き込みは可能なかぎりただちに処理され、「スキャンモード」のプロパティの影響を受けません。



「スキャンモード」: 購読しているクライアントに送信される更新についてデバイス内のタグをどのようにスキャンするかを指定します。オプションの説明は次のとおりです。

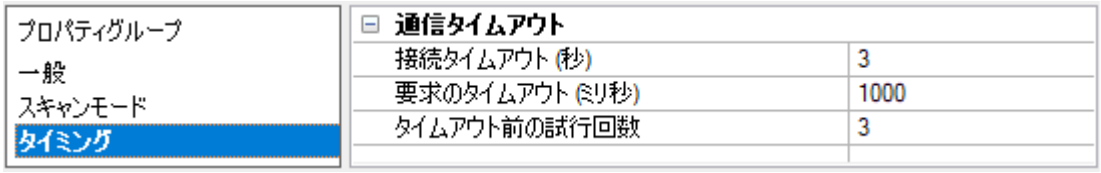
- **「クライアント固有のスキャン速度を適用」**: このモードでは、クライアントによって要求されたスキャン速度を使用します。
- **「指定したスキャン速度以下でデータを要求」**: このモードでは、最大スキャン速度として設定されている値を指定します。有効な範囲は 10 から 99999990 ミリ秒です。デフォルトは 1000 ミリ秒です。
  注記: サーバーにアクティブなクライアントがあり、デバイスのアイテム数とスキャン速度の値が増加している場合、変更はただちに有効になります。スキャン速度の値が減少している場合、すべてのクライアントアプリケーションが切断されるまで変更は有効になりません。
- **「すべてのデータを指定したスキャン速度で要求」**: このモードでは、指定した速度で購読済みクライアント用にタグがスキャンされます。有効な範囲は 10 から 99999990 ミリ秒です。デフォルトは 1000 ミリ秒です。
- **「スキャンしない、要求ポールのみ」**: このモードでは、デバイスに属するタグは定期的にポーリングされず、アクティブになった後はアイテムの初期値の読み取りは実行されません。更新のポーリングは、_DemandPoll タグに書き込むか、個々のアイテムについて明示的なデバイス読み取りを実行することによって、OPC クライアントが行います。詳細については、サーバーのヘルプで「デバイス要求ポール」を参照してください。
- **「タグに指定のスキャン速度を適用」**: このモードでは、静的構成のタグプロパティで指定されている速度で静的タグがスキャンされます。動的タグはクライアントが指定したスキャン速度でスキャンされます。

「キャッシュからの初期更新」: このオプションを有効にした場合、サーバーは保存 (キャッシュ) されているデータから、新たにアクティブ化されたタグ参照の初回更新を行います。キャッシュからの更新は、新しいアイテム参照が同じアドレス、ス

キャン速度、データ型、クライアントアクセス、スケール設定のプロパティを共有している場合にのみ実行できます。1 つ目のクライアント参照についてのみ、初期更新にデバイス読み取りが使用されます。デフォルトでは無効になっており、クライアントがタグ参照をアクティブ化したときにはいつでも、サーバーがデバイスから初期値の読み取りを試みます。

デバイスのプロパティ - タイミング

デバイスのタイミングのプロパティでは、エラー状態に対するデバイスの応答をアプリケーションのニーズに合わせて調整できます。多くの場合、最適なパフォーマンスを得るためにはこれらのプロパティを変更する必要があります。電気的に発生するノイズ、モデムの遅延、物理的な接続不良などの要因が、通信ドライバーで発生するエラーやタイムアウトの数に影響します。タイミングのプロパティは、設定されているデバイスごとに異なります。



通信タイムアウト

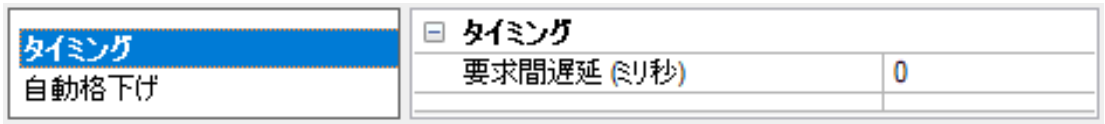
「接続タイムアウト」: このプロパティ (イーサネットベースのドライバーで主に使用) は、リモートデバイスとのソケット接続を確立するために必要な時間を制御します。デバイスの接続時間は、同じデバイスへの通常の通信要求よりも長くかかることがよくあります。有効な範囲は 1 から 30 秒です。デフォルトは通常は 3 秒ですが、各ドライバーの特性によって異なる場合があります。この設定がドライバーでサポートされていない場合、無効になります。
注記: UDP 接続の特性により、UDP を介して通信する場合には接続タイムアウトの設定は適用されません。

「要求のタイムアウト」: すべてのドライバーがターゲットデバイスからの応答の完了を待機する時間を決定するために使用する間隔を指定します。有効な範囲は 50 から 9,999,999 ミリ秒 (167 分) です。デフォルトは通常は 1000 ミリ秒ですが、ドライバーによって異なる場合があります。ほとんどのシリアルドライバーのデフォルトのタイムアウトは 9600 ボー以上のボーレートに基づきます。低いボーレートでドライバーを使用している場合、データの取得に必要な時間が増えることを補うため、タイムアウト時間を増やします。

「タイムアウト前の試行回数」: ドライバーが通信要求を発行する回数を指定します。この回数を超えると、要求が失敗してデバイスがエラー状態にあると見なされます。有効な範囲は 1 から 10 です。デフォルトは通常は 3 ですが、各ドライバーの特性によって異なる場合があります。アプリケーションに設定される試行回数は、通信環境に大きく依存します。このプロパティは、接続の試行と要求の試行の両方に適用されます。

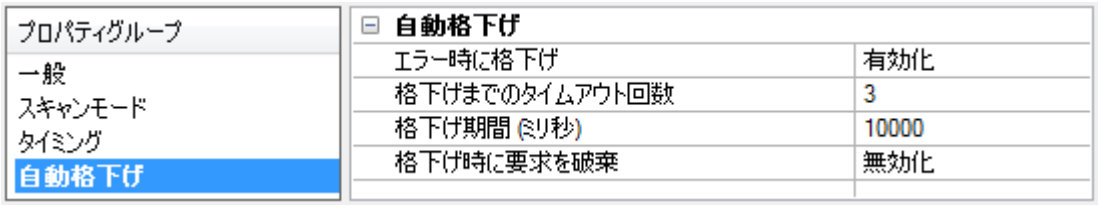
タイミング

「要求間遅延」: ドライバーがターゲットデバイスに次の要求を送信するまでの待ち時間を指定します。デバイスに関連付けられているタグおよび 1 回の読み取りと書き込みの標準のポーリング間隔がこれによってオーバーライドされます。この遅延は、応答時間が長いデバイスを扱う際や、ネットワークの負荷が問題である場合に役立ちます。デバイスの遅延を設定すると、そのチャネル上のその他すべてのデバイスとの通信に影響が生じます。可能な場合、要求間遅延を必要とするデバイスは別々のチャネルに分けて配置することをお勧めします。その他の通信プロパティ (通信シリアル化など) によってこの遅延が延長されることがあります。有効な範囲は 0 から 300,000 ミリ秒ですが、一部のドライバーでは独自の設計の目的を果たすために最大値が制限されている場合があります。デフォルトは 0 であり、ターゲットデバイスへの要求間に遅延はありません。
注記: すべてのドライバーで「要求間遅延」がサポートされているわけではありません。使用できない場合にはこの設定は表示されません。



デバイスのプロパティ - 自動格下げ

自動格下げのプロパティを使用することで、デバイスが応答していない場合にそのデバイスを一時的にスキャン停止にできます。応答していないデバイスを一定期間オフラインにすることで、ドライバーは同じチャネル上のほかのデバイスとの通信を引き続き最適化できます。停止期間が経過すると、ドライバーは応答していないデバイスとの通信を再試行します。デバイスが応答した場合はスキャンが開始され、応答しない場合はスキャン停止期間が再開します。



「エラー時に格下げ」: 有効にした場合、デバイスは再び応答するまで自動的にスキャン停止になります。
ヒント: システムタグ _AutoDemoted を使用して格下げ状態をモニターすることで、デバイスがいつスキャン停止になったかを把握できます。

「格下げまでのタイムアウト回数」: デバイスをスキャン停止にするまでに要求のタイムアウトと再試行のサイクルを何回繰り返すかを指定します。有効な範囲は 1 から 30 回の連続エラーです。デフォルトは 3 です。

「格下げ期間」: タイムアウト値に達したときにデバイスをスキャン停止にする期間を指定します。この期間中、そのデバイスには読み取り要求が送信されず、その読み取り要求に関連するすべてのデータの品質は不良に設定されます。この期間が経過すると、ドライバーはそのデバイスのスキャンを開始し、通信での再試行が可能になります。有効な範囲は 100 から 3600000 ミリ秒です。デフォルトは 10000 ミリ秒です。

「格下げ時に要求を破棄」: スキャン停止期間中に書き込み要求を試行するかどうかを選択します。格下げ期間中も書き込み要求を必ず送信するには、無効にします。書き込みを破棄するには有効にします。サーバーはクライアントから受信した書き込み要求をすべて自動的に破棄し、イベントログにメッセージを書き込みません。

デバイスのプロパティ - タグ生成

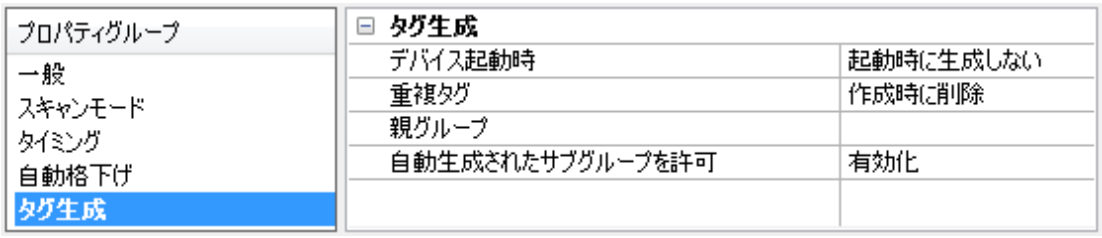
自動タグデータベース生成機能によって、アプリケーションの設定がプラグアンドプレイ操作になります。デバイス固有のデータに対応するタグのリストを自動的に構築するよう通信ドライバーを設定できます。これらの自動生成されたタグ (サポートしているドライバーの特性によって異なる) をクライアントからブラウズできます。

一部のデバイスやドライバーは自動タグデータベース生成のフル機能をサポートしていません。また、すべてのデバイスやドライバーが同じデータ型をサポートするわけではありません。詳細については、データ型の説明を参照するか、各ドライバーがサポートするデータ型のリストを参照してください。

ターゲットデバイスが独自のローカルタグデータベースをサポートしている場合、ドライバーはそのデバイスのタグ情報を読み取って、そのデータを使用してサーバー内にタグを生成します。デバイスが名前付きのタグをネイティブにサポートしていない場合、ドライバーはそのドライバー固有の情報に基づいてタグのリストを作成します。この 2 つの条件の例は次のとおりです。

1. データ取得システムが独自のローカルタグデータベースをサポートしている場合、通信ドライバーはデバイスで見つかったタグ名を使用してサーバーのタグを構築します。
2. イーサネット I/O システムが独自の使用可能な I/O モジュールタイプの検出をサポートしている場合、通信ドライバーはイーサネット I/O ラックにプラグイン接続している I/O モジュールのタイプに基づいてサーバー内にタグを自動的に生成します。

注記: 自動タグデータベース生成の動作モードを詳細に設定できます。詳細については、以下のプロパティの説明を参照してください。



「プロパティ変更時」: デバイスが、特定のプロパティが変更された際の自動タグ生成をサポートする場合、**「プロパティ変更時」**オプションが表示されます。これはデフォルトで**「はい」**に設定されていますが、**「いいえ」**に設定してタグ生成を実行する時期を制御できます。この場合、タグ生成を実行するには**「タグを作成」**操作を手動で呼び出す必要があります。

「デバイス起動時」: OPC タグを自動的に生成するタイミングを指定します。オプションの説明は次のとおりです。

- **「起動時に生成しない」**: このオプションを選択した場合、ドライバーはOPC タグをサーバーのタグ空間に追加しません。これはデフォルトの設定です。
- **「起動時に常に生成」**: このオプションを選択した場合、ドライバーはデバイスのタグ情報を評価します。さらに、サーバーが起動するたびに、サーバーのタグ空間にタグを追加します。
- **「最初の起動時に生成」**: このオプションを選択した場合、そのプロジェクトが初めて実行されたときに、ドライバーがデバイスのタグ情報を評価します。さらに、必要に応じてOPC タグをサーバーのタグ空間に追加します。

注記: OPC タグを自動生成するオプションを選択した場合、サーバーのタグスペースに追加されたタグをプロジェクトとともに保存する必要があります。ユーザーは**「ツール」**|**「オプション」**メニューから、自動保存するようプロジェクトを設定できます。

「重複タグ」: 自動タグデータベース生成が有効になっている場合、サーバーが以前に追加したタグや、通信ドライバーが最初に作成した後で追加または修正されたタグを、サーバーがどのように処理するかを設定する必要があります。この設定では、自動生成されてプロジェクト内に現在存在するOPC タグをサーバーがどのように処理するかを制御します。これによって、自動生成されたタグがサーバーに累積することもなくなります。

たとえば、**「起動時に常に生成」**に設定されているサーバーのラックでI/O モジュールを変更した場合、通信ドライバーが新しいI/O モジュールを検出するたびに新しいタグがサーバーに追加されます。古いタグが削除されなかった場合、多数の未使用タグがサーバーのタグ空間内に累積することがあります。以下のオプションがあります。

- **「作成時に削除」**: このオプションを選択した場合、新しいタグが追加される前に、以前にタグ空間に追加されたタグがすべて削除されます。これはデフォルトの設定です。
- **「必要に応じて上書き」**: このオプションを選択した場合、サーバーは通信ドライバーが新しいタグに置き換えているタグだけ除去します。上書きされていないタグはすべてサーバーのタグ空間に残ります。
- **「上書きしない」**: このオプションを選択した場合、サーバーは以前に生成されたタグやサーバーにすでに存在するタグを除去しません。通信ドライバーは完全に新しいタグだけを追加できます。
- **「上書きしない、エラーを記録」**: このオプションには上記のオプションと同じ効果がありますが、タグの上書きが発生した場合にはサーバーのイベントログにエラーメッセージも書き込まれます。

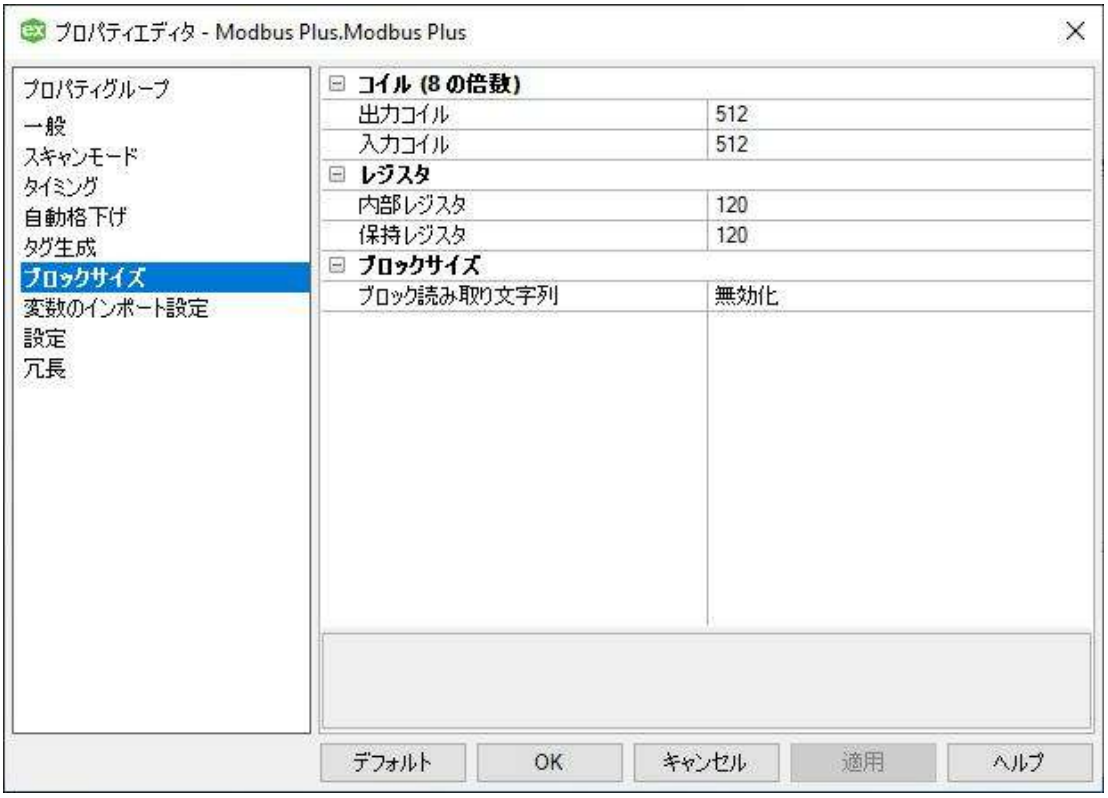
注記: OPC タグの除去は、通信ドライバーによって自動生成されたタグ、および生成されたタグと同じ名前を使用して追加されたタグに影響します。ドライバーによって自動生成されるタグと一致する可能性がある名前を使用してサーバーにタグを追加しないでください。

「親グループ」: このプロパティでは、自動生成されたタグに使用するグループを指定することで、自動生成されたタグと、手動で入力したタグを区別します。グループの名前は最大 256 文字です。この親グループは、自動生成されたすべてのタグが追加されるルートブランチとなります。

「自動生成されたサブグループを許可」: このプロパティでは、自動生成されたタグ用のサブグループをサーバーが自動的に作成するかどうかを制御します。これはデフォルトの設定です。無効になっている場合、サーバーはグループを作成しないで、デバイスのタグをフラットリスト内に生成します。サーバープロジェクトで、生成されたタグには名前としてアドレスの値が付きます。たとえば、生成プロセス中はタグ名は維持されません。
注記: サーバーがタグを生成しているときに、タグに既存のタグと同じ名前が割り当てられた場合、タグ名が重複しないようにするため、番号が自動的に1 つ増分します。たとえば、生成プロセスによってすでに存在する "AI22" という名前のタグが作成された場合、代わりに "AI23" としてタグが作成されます。

「作成」: 自動生成 OPC タグの作成を開始します。**「タグを作成」**が有効な場合、デバイスの構成が修正されると、ドライバーはタグ変更の可能性についてデバイスを再評価します。システムタグからアクセスできるため、クライアントアプリケーションはタグデータベース作成を開始できます。
注記: 構成がプロジェクトをオフラインで編集する場合、**「タグを作成」**は無効になります。

デバイスのプロパティ - ブロックサイズ



コイル

「出力コイル」: 出力ブロックのサイズをビット数で指定します。コイルは 8 から 2000 ポイント (ビット) の範囲で一度に読み取ることができます。

「入力コイル」: 入力ブロックのサイズをビット数で指定します。コイルは 8 から 2000 ポイント (ビット) の範囲で一度に読み取ることができます。

注記:

1. コイルのサイズは 8 の倍数にする必要があります。
2. MBX、NETLIB、NONE の場合、デフォルトは 512 で最大は 2000 です。
3. OPC クライアントが接続している場合、このプロパティは無効になります。

「レジスタ」

「内部レジスタ」: 内部レジスタのブロックサイズをビット数で指定します。1 から 125 の標準 16 ビット Modbus レジスタを一度で読み取ることができます。

「保持レジスタ」: 保持レジスタのブロックサイズをビット数で指定します。1 から 125 の標準 16 ビット Modbus レジスタを一度で読み取ることができます。

注記:

1. MBX、NETLIB、NONE の場合、デフォルトは 120 で最大は 125 です。
2. OPC クライアントが接続している場合、このプロパティは無効になります。
3. TIO モジュールの場合、この設定を使用して、データ位置 400001 を読み取る際に返されるバイト数をドライバーに通知します。2 バイトを返すモジュールの場合、これを 1 に設定します。3 バイトを返すモジュールの場合、これを 2 に設定します。ドライバーはその他すべてのデータ位置に (この設定とは関係なく) 固定のブロック長を使用します。

4. デバイスでデフォルトサイズのブロック読み取り/書き込み操作がサポートされていないことがあります。小さいModicon PLC および Modicon 以外のデバイスでは、MBPlus ネットワークでサポートされているデータ転送の最大長がサポートされないことがあります。
5. 連続しないアドレスがデバイスに含まれていることがあります。その場合、未定義のメモリを含むデータブロックをドライバーが読み取ろうとすると、デバイスはおそらくその要求を却下します。

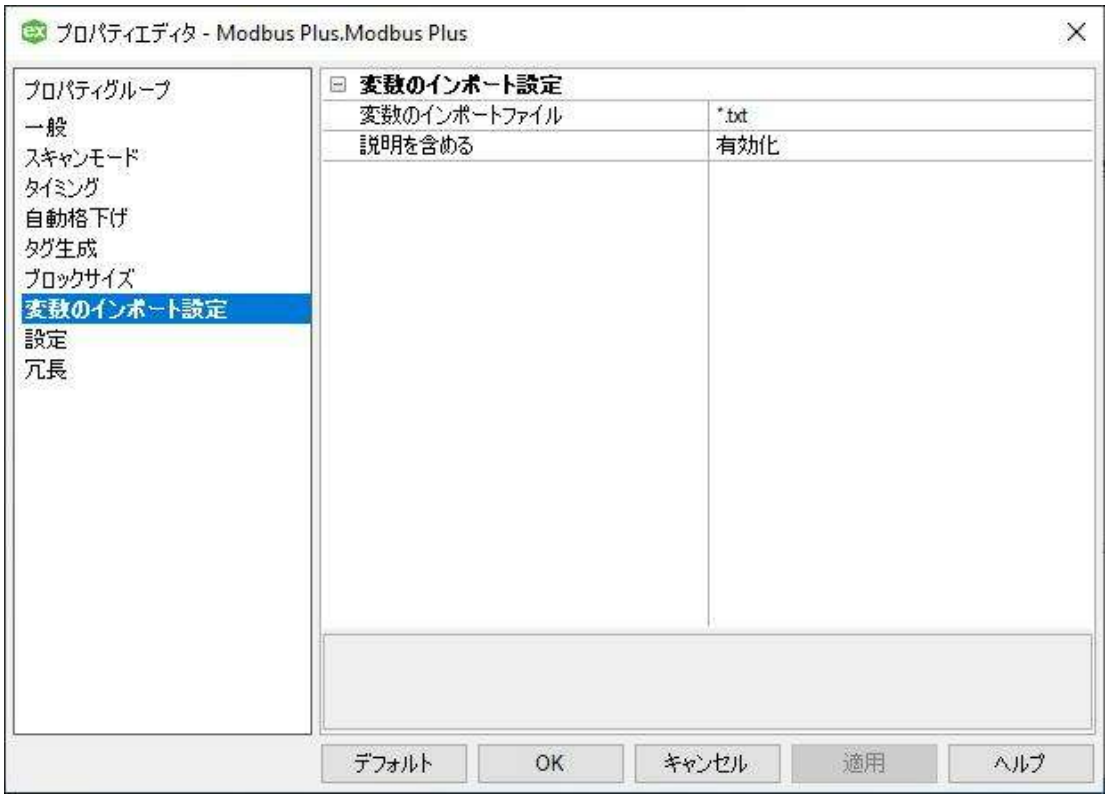
警告: ブロックサイズとして 120 より大きい値が設定され、任意のタグに 32 ビットまたは 64 ビットデータ型が使用されている場合、"ブロックに不良アドレスがあります" というエラーが発生することがあります。このエラーが発生しないようにするには、ブロックサイズの値を 120 に減らします。

ブロックサイズ

「ブロック読み取り文字列」: 通常は個別に読み取る文字列タグをグループで読み取ります。文字列タグはブロックサイズに応じてグループ化されます。ブロック読み取りは Modbus モデルの文字列タグに対してのみ実行できます。

デバイスのプロパティ - 変数のインポート設定

*Modbus ドライバー向け CSV ファイルの詳細については、**Modbus ドライバー向け CSV ファイルの作成**を参照してください。*



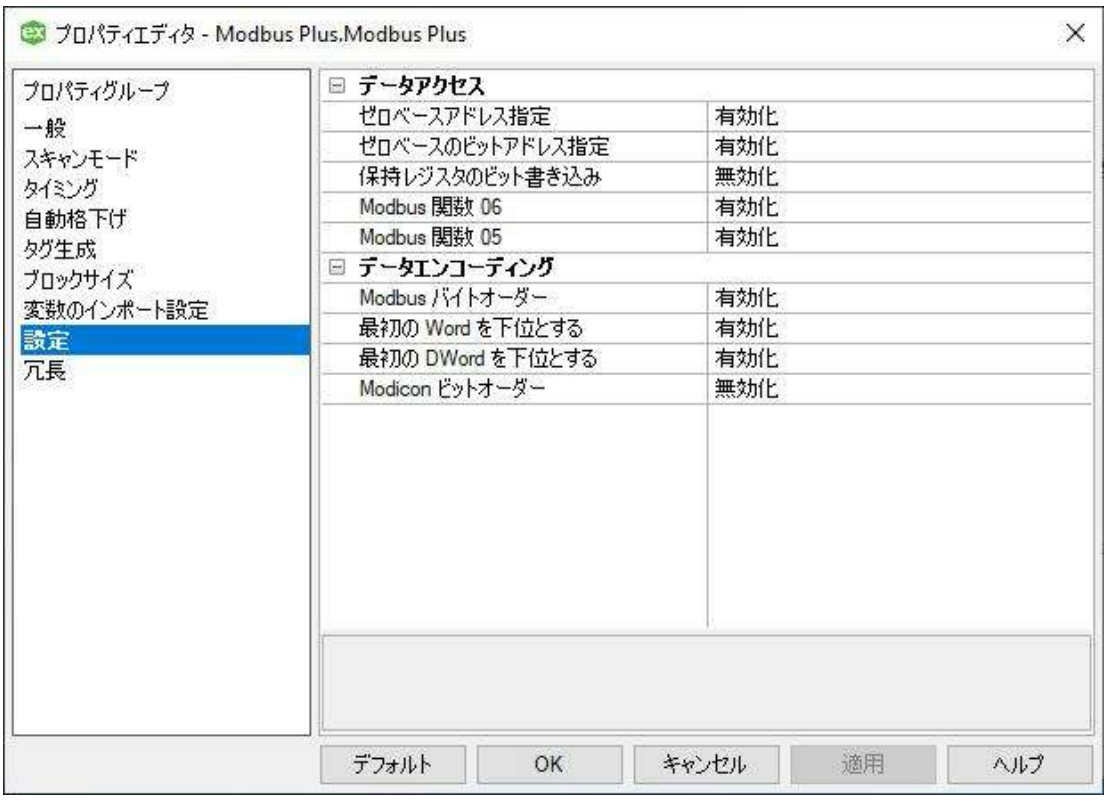
「変数のインポートファイル」: このデバイスで自動タグデータベース生成が有効になっている場合にドライバーが使用する変数インポートファイルの正確な場所を指定します。

「説明を含める」: インポートされたタグの説明 (ファイル内に存在する場合) を使用するには有効にします。

*自動タグデータベース生成機能の設定方法および変数インポートファイルの作成方法については、**自動タグデータベース生成**を参照してください。*

Concept および ProWORX からの変数インポートファイルの作成方法については、技術情報「Modbus ドライバー向け CSV ファイルの作成」を参照してください。

デバイスのプロパティ - 設定



データアクセス

「ゼロベースアドレス指定」: デバイスのアドレス番号付けの規則で番号がゼロではなく 1 で開始する場合、無効にします。デフォルトでは、Modbus デバイスと通信するためにフレームを構築する場合はユーザーが入力したアドレスから 1 が引かれます。デバイスがこの規則に従わない場合、**「ゼロベースアドレス指定」**を無効にしてください。デフォルトの動作は Modicon PLC の規則に従います。

「ゼロベースのビットアドレス指定」: Word 内のビットをブールとして参照可能なメモリタイプ。アドレス指定の表記は <アドレス>.<ビット> であり、ここで <ビット> は Word 内のビット番号を表します。レジスタ内のゼロベースのビットアドレス指定によって、ある Word 内の 1 ビットをゼロベースと 1 ベースの 2 つの方法によってアドレス指定できます。レジスタ内のゼロベースのビットアドレス指定は、最初のビットが 0 で始まることを意味します。1 ベースの場合、最初のビットは 1 で始まります。

「保持レジスタのビット書き込み」: 保持レジスタ内のビット位置に書き込む際、ドライバーは対象のビットのみを修正する必要があります。一部のデバイスはレジスタ内の 1 ビットを操作するコマンドをサポートしています (ファンクションコード 0x16 (16 進) または 22 (10 進))。デバイスがこの機能をサポートしていない場合、ドライバーは 1 ビットだけが変更されるように読み取り/修正/書き込み操作を実行する必要があります。デフォルトでは無効に設定されています。デバイスが保持レジスタのビットアクセスをサポートする場合は有効にします。その場合、ドライバーは「Modbus 関数 06」の設定に関係なくファンクションコード 0x16 を使用します。この設定が無効になっている場合、ドライバーは「Modbus 関数 06」プロパティの設定に応じて、単一レジスタへの書き込みにファンクションコード 0x06 または 0x10 を使用します。

注記:

1. 「Modbus バイトオーダー」が無効になっている場合、このコマンドで送信されるマスクのバイトオーダーは Intel バイトオーダーになります。

「Modbus 関数 06」: Modbus Plus ドライバーは、2 つの Modbus プロトコルファンクションを使用して保持レジスタのデータをターゲットデバイスに書き込むことができます。ほとんどの場合、このドライバーは書き込み対象のレジスタの数に基づいてこの 2 つのファンクションを切り替えます。単一の 16 ビットレジスタに書き込む場合、このドライバーは Modbus ファンクション 06 を使用します。32 ビット値を 2 つのレジスタに書き込む場合、このドライバーは Modbus ファンクション 16 を使用します。標準の Modicon PLC では、このどちらのファンクションを使用しても問題ありません。ただし、Modbus プロトコルを実装しているサードパーティデバイスが多数存在します。これらのデバイスの多くは、書き込み対象のレジスタの数に関係なく、保持レジスタへの書き込みに Modbus ファンクション 16 の使用のみをサポートしています。

「Modbus 関数 06」を使用することで、ドライバーにModbus ファンクション 16 のみを使用させることができます (必要な場合)。この選択はデフォルトで有効になっており、ドライバーは必要に応じて 06 と 16 を切り替えることができます。デバイスが Modbus ファンクション 16 のみを使用してすべての書き込みを行う必要がある場合、このオプションを無効にします。

注記: Word 内のビットの書き込みでは、**「保持レジスタのビット書き込み」**プロパティが**「Modbus 関数 06」**よりも優先されます。「保持レジスタのビット書き込み」が有効になっている場合、このプロパティの選択にかかわらず、ファンクションコード 0x16 が使用されます。無効になっている場合、Word 内のビットの書き込みには、このプロパティに応じてファンクションコード 0x06 または 0x10 が使用されます。

Modbus「Modbus 関数 05」: Modbus Plusドライバー は、2 つの Modbus プロトコルファンクションを使用して出力コイルデータをターゲットデバイスに書き込むことができます。ほとんどの場合、このドライバーは書き込み対象のコイルの数に基づいてこの 2 つのファンクションを切り替えます。単一のコイルに書き込む場合、このドライバーは Modbus ファンクション 05 を使用します。コイルの配列に書き込む場合、このドライバーは Modbus ファンクション 15 を使用します。標準の Modicon PLC では、このどちらのファンクションを使用しても問題ありません。ただし、Modbus プロトコルを実装しているサードパーティデバイスが多数存在します。これらのデバイスの多くは、書き込み対象のコイルの数に関係なく、出力コイルへの書き込みにModbus ファンクション 15 の使用のみをサポートしています。

「Modbus 関数 05」を使用することで、必要な場合にドライバーにModbus ファンクション 15 のみを使用させることができます。このプロパティはデフォルトで有効になっており、ドライバーは必要に応じて 05 と 15 を切り替えることができます。デバイスが Modbus ファンクション 15 のみを使用してすべての書き込みを行う必要がある場合、この選択を無効にします。

データエンコーディング

「Modbus バイトオーダー」: この選択を使用することで、Modbus Plusドライバー で使用されるバイトオーダーをデフォルトの Modbus バイトオーダーから Intel バイトオーダーに変更できます。この選択はデフォルトで有効になっており、Modbus 対応デバイスでは標準の設定です。デバイスが Intel バイトオーダーを使用する場合、このオプションを無効にすることで、ドライバーは Intel フォーマットのデータを適切に読み取ることができます。

「最初の Word を下位とする」: Modbus デバイスでは 32 ビットデータ型に 2 つの連続するレジスタアドレスが使用されます。ドライバーが最初の Word を 32 ビット値の下位 Word とするか上位 Word とするかを指定できます。デフォルト (「最初の Word を下位とする」) は、Modicon Modsoft プログラミングソフトウェアの規則に従います。

「最初の DWord を下位とする」: Modbus デバイスでは 64 ビットデータ型に 4 つの連続するレジスタアドレスが使用されます。ドライバーが最初の DWord を 64 ビット値の下位 DWord とするか上位 DWord とするかを指定できます。デフォルト (「最初の DWord を下位とする」) は、32 ビットデータ型のデフォルトの規則に従います。

「Modicon ビットオーダー」: 有効な場合、ドライバーはレジスタに対する読み書きの際にビットオーダーを反転して Modicon Modsoft プログラミングソフトウェアの規則に従います。たとえば、このオプションが有効になっている場合、アドレス 40001.0/1 への書き込みはこのデバイスのビット 15/16 に影響します。このオプションはデフォルトで無効になっています。

注記: 次の例では、そのドライバーに設定されているレジスタ内の「ゼロベースのビットアドレス指定」に応じて、1 から 16 番目のビットは 0-15 ビットまたは 1-16 ビットを表します。

MSB = 最上位ビット
LSB = 最下位ビット

「Modicon ビットオーダー」が有効

MSB															LSB
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

「Modicon ビットオーダー」が無効

MSB															LSB
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

データエンコーディングオプションの詳細

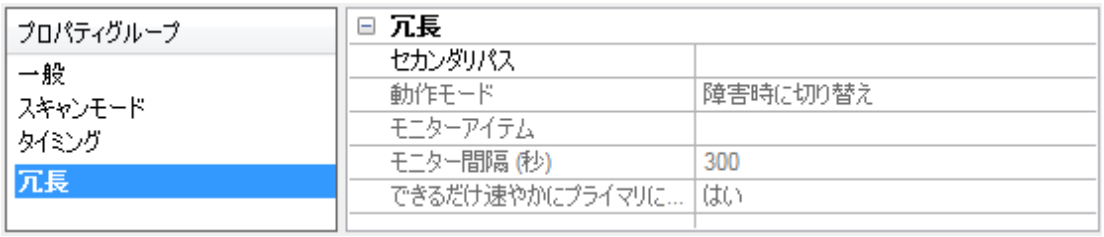
- 「Modbus バイトオーダー」オプションでは、各レジスタ/16 ビット値のデータエンコーディングを設定します。
- 「最初の Word を下位とする」では、各 32 ビット値と、64 ビット値の各 DWord のデータエンコーディングを設定します。
- 「最初の DWord を下位とする」では、各 64 ビット値のデータエンコーディングを設定します。

データ型	「Modbus バイトオーダー」	「最初の Word を下位とする」	「最初の DWord を下位とする」
Word、Short、BCD	適用可能	なし	なし
Float、DWord、Long、LBCD	適用可能	適用可能	なし
Double	適用可能	適用可能	適用可能

必要な場合、以下の情報と個々のデバイスのドキュメントを参照して、データエンコーディングオプションの正しい設定を調べてください。ほとんどの Modbus デバイスではデフォルト設定で問題ありません。

「Modbus バイトオーダー」が有効	上位バイト (15..8)	下位バイト (7..0)
「Modbus バイトオーダー」が無効	下位バイト (7..0)	上位バイト (15..8)
「最初の Word を下位とする」が無効	上位 Word (31..16) 64 ビットデータ型での DWord の上位 Word (63..48)	下位 Word (15..0) 64 ビットデータ型での DWord の下位 Word (47..32)
「最初の Word を下位とする」が有効	下位 Word (15..0) 64 ビットデータ型での DWord の下位 Word (47..32)	上位 Word (31..16) 64 ビットデータ型での DWord の上位 Word (63..48)
「最初の DWord を下位とする」が無効	上位 DWord (63..32)	下位 DWord (31..0)

デバイスのプロパティ - 冗長



冗長設定はメディアレベルの冗長プラグインで使用できます。
詳細については、Web サイトまたはユーザーマニュアルを参照するか、営業担当者までお問い合わせください。

自動タグデータベース生成

Modbus Plusドライバーでは自動タグデータベース生成が利用されるため、デバイスのラダープログラムによって使用されるデータポイントにアクセスするタグが自動的に作成されます。タグデータベースの構築に必要な情報をデバイスに対して照会可能な場合もありますが、このドライバーは代わりに変数インポートファイルを使用する必要があります。変数インポートファイルはConceptおよびProWORXデバイスプログラミングアプリケーションを使用して生成できます。

変数インポートファイルの作成

このインポートファイルは、Conceptデバイスプログラミングアプリケーションのデフォルトのエクスポートファイルフォーマットであるセミコロン区切りの.TXTフォーマットでなければなりません。ProWORXプログラミングアプリケーションはこのフォーマットで変数データをエクスポートできます。

Conceptおよび ProWORXからの変数インポートファイルの作成方法については、技術情報「Modbusドライバー向け CSVファイルの作成」を参照してください。

サーバー構成

アプリケーションのニーズに合わせて自動タグデータベース生成をカスタマイズできます。ウィザードを利用したデータベースの作成中またはデバイスのプロパティでプライマリ制御オプションを設定できます。

詳細については、サーバーのヘルプドキュメントを参照してください。

Modbus Plusドライバーは、自動タグデータベース生成をサポートするすべてのドライバーに共通する基本設定に加え、その他の設定を必要とします。このような特別な設定として変数インポートファイルの名前や場所を指定する必要があります。この情報はデバイスのプロパティの「変数のインポート設定」で指定できます。

*詳細については、**変数のインポート設定**を参照してください。*

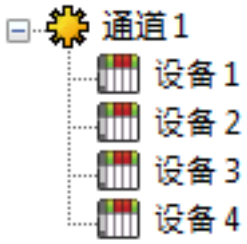
操作

個々の設定に応じて、タグ生成はサーバープロジェクトが開いたときに自動的に開始するか、後から手動で開始できます。「イベントログ」には、タグ生成プロセスの開始時刻、変数インポートファイルの処理中に発生したエラー、このプロセスの完了時刻が示されます。

通信の最適化

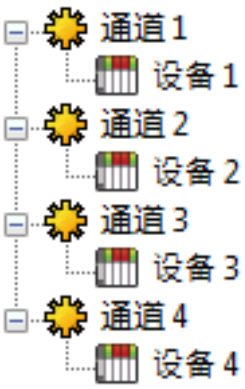
Modbus Plusドライバーはスループットを向上させてSA85カードをフルに活用できるように設計されています。これまで、Modbus Plusドライバーではサーバープロジェクトで設定するチャネルは1つに制限され、アクセス先のすべてのModbus Plusデバイスをこのチャネルの下で定義する必要がありました。つまり、ドライバーは要求を行う際にデバイス間を移動する必要がありました。OPCサーバーはすでに効率的な設計になっていたため、単一チャネルスキームでもほとんどのアプリケーションでは十分なパフォーマンスが得られました。しかし、実現技術としてのOPCの到来によって、プロジェクトのサイズは劇的に大きくなりました。高度なパフォーマンスを維持するため、Modbus Plusドライバーは高い効率とパフォーマンスで動作するよう設計されています。

注記: これらの変更を開始する前に、必要に応じて前の設定に戻せるようにサーバープロジェクトのディレクトリをバックアップしてください。



このプロジェクトでは、チャネルが1つだけ定義されています。アクセスする必要があるすべてのデバイスがこの1つのチャネルの下で定義されています。Modbus Plusドライバーは効果的な速度で情報を収集するために、できるだけ速やかにあるデバイスから次のデバイスに移動する必要があります。さらにデバイスが追加されたり、1つのデバイスからより多くの情報が要求されたりするにしたがい、更新レートが低下していきます。

最新バージョンのModbus Plusドライバーでは、アプリケーションのパフォーマンスを向上させるため、複数チャネル定義が使用されています。この構成では、サーバー内の各チャネルが個々の実行パスを表します。新しいチャネルを追加することにより、アプリケーションのワークロードがそれらのチャネルに分散されます。これによって独立して実行する複数の実行パスが作成され、パフォーマンスが大幅に向上します。以下の図では、同じアプリケーションが複数のチャネルを使用するように構成が変更されています。



各デバイスを、そのデバイス専用のチャネルの下で定義することができます。この構成では、各デバイスに専用のチャネルが割り当てられているため、サーバーは 1 台のデバイスからデータを収集するタスクに対して、1 つの実行パスを割り当てることができます。アプリケーションのデバイス数が少ない場合は、ここで説明した方法で最適化することができます。

アプリケーションのデバイス数が多い場合であっても、パフォーマンスは向上します。デバイスの数は少ないことが理想的ですが、そうでない場合でもアプリケーションは追加のチャネルから恩恵を受けます。それぞれのチャネル内でサーバーはデバイス間を移動する必要がありますが、1 つのパスで処理するデバイスの数は少なくなります。

注記: チャネル数の制限は、SA85 カードの製造メーカーによって設定されている複数パスの制限に対応しています。

各デバイスの下で多数のタグが定義されている場合でも複数のチャネルをサポートするようにアプリケーションを設計変更できます。詳細については、次の手順に従います。

1. 単一チャネルベースの既存のプロジェクトで、**「接続性」**|**「新しいチャネル」**の順にクリックし、任意の名前を付けます。
2. **ModbusPlus** チャネルから **PLC2** を切り取ります。
3. これを新しいチャネルの下に貼り付けます。切り取り/貼り付け機能を使用することで、新しい Modbus Plus ドライバーの利点を活用できるようにアプリケーションを簡単に修正できます。

この例では、Modbus Plus ドライバーで可能な最もわかりやすい最適化に注目しています。そのほかにも、1 つのチャネルをグローバルデータだけに割り当てるという最適化の方法があります。これには、その新しいチャネルでアクセスする、グローバルデータを持つ各デバイスに一連の新しいデバイス名を定義します。グローバルデータには、これらの新たに定義したデバイス名からのみアクセスすることを忘れないでください。

データ型の説明

データ型	説明
Boolean	1 ビット
Word	符号なし 16 ビット値 ビット 0 が下位ビット ビット 15 が上位ビット
Short	符号付き 16 ビット値 ビット 0 が下位ビット ビット 14 が上位ビット ビット 15 が符号ビット
DWord	符号なし 32 ビット値 ビット 0 が下位ビット ビット 31 が上位ビット
Long	符号付き 32 ビット値 ビット 0 が下位ビット ビット 30 が上位ビット ビット 31 が符号ビット
BCD	2 バイトパックされた BCD 値の範囲は 0-9999 です。この範囲外の値には動作が定義されていません。
LBCD	4 バイトパックされた BCD 値の範囲は 0-99999999 です。この範囲外の値には動作が定義されていません。
String	Null 終端 ASCII 文字列 Modbus モデルでサポートされ、バイトオーダーを HiLo/LoHi から選択できます。
Double*	64 ビット浮動小数点値 ドライバーは最後の 2 つのレジスタを上位 DWord、最初の 2 つのレジスタを下位 DWord とすることで、連続する 4 つのレジスタを倍精度値として解釈します。
Double の例	レジスタ 40001 が Double として指定されている場合、レジスタ 40001 のビット 0 は 64 ビットデータ型のビット 0 になり、レジスタ 40004 のビット 15 は 64 ビットデータ型のビット 63 になります。
Float*	32 ビット浮動小数点値 ドライバーは最後のレジスタを上位 Word、最初のレジスタを下位 Word とすることで、連続する 2 つのレジスタを単精度値として解釈します。
Float の例	レジスタ 40001 が Float として指定されている場合、レジスタ 40001 のビット 0 は 32 ビットデータ型のビット 0 になり、レジスタ 40002 のビット 15 は 32 ビットデータ型のビット 31 になります。

*この説明は、64 ビットデータ型では最初の DWord を下位とし、32 ビットデータ型では最初の Word を下位とするデフォルトのデータ処理を前提としています。

アドレスの説明

アドレスの仕様は使用されているモデルによって異なります。対象のモデルのアドレス情報を取得するには、次のリストからリンクを選択してください。

Modbus のアドレス指定
TIO モジュールのアドレス指定

Modbus のアドレス指定

このドライバーでは、「サーバー」と「非送信請求」という用語は同じ意味になります。

5 桁のアドレス指定と 6 桁のアドレス指定

Modbus のアドレス指定では、アドレスの最初の桁はプライマリテーブルを示します。以降の桁はデバイスのデータアイテムを表します。最大値は 2 バイトの符号なし整数 (65,535) です。アドレステーブルとアイテム全体を表すのに 6 桁が必要です。このため、デバイスのマニュアルで 0xxxx、1xxxx、3xxxx、4xxxx として指定されているアドレスは、Modbus タグのアドレスフィールドに適用されると追加のゼロが 1 つパディングされます。

プライマリテーブル	説明
0	出力コイル
1	入力コイル
3	内部レジスタ
4	保持レジスタ

Modbus のアドレス指定 (10 進フォーマット)

参加者の ファンクションコードは 10 進数で表示されます。詳細については、ファンクションコードの説明を参照してください。

アドレスタイプ	範囲	データ型	アクセス	ファンクションコード
出力コイル	000001-065536	Boolean	読み取り/書き込み	01, 05, 15
入力コイル	100001-165536	Boolean	読み取り専用	02
内部レジスタ	300001-365536 300001-365535 300001-365533 3xxxxx.0/1-3xxxxx.15/16** 300001.2H-365536.240H*** 300001.2L-365536.240L***	**Word**、Short、BCD Float、DWord、Long、 LBCD Double **Boolean** String String	読み取り専用* 読み取り専用* 読み取り専用* 読み取り専用* 読み取り専用 読み取り専用	04 04 04 04 04 04
保持レジスタ	400001-465536 400001-465535 400001-465533 4xxxxx.0/1-4xxxxx.15/16** 400001.2H-465536.240H*** 400001.2L-465536.240L***	**Word**、Short、BCD Float、DWord、Long、 LBCD Double **Boolean** String String	読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り/書き込み	03, 06, 16 03, 06, 16 03, 06, 16 03, 06, 16, 22 03, 16 03, 16

アドレスタイプ	範囲	データ型	アクセス	ファンクションコード
			読み取り/書き込み	
グローバルデータ	G01-G32 G01-G31 G01-G29 Gxx.0/1-Gxx.15/16**	**Word**、Short、BCD Float、DWord、Long、LBCD Double **Boolean**	読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り専用	該当なし なし なし なし

*Modbus サーバーデバイスの場合、これらの位置は読み取り/書き込みが可能です。
**詳細については、設定の「ゼロベースと1ベースのアドレス指定」を参照してください。
***ピリオドの後ろのビット番号は2から240バイトの範囲の文字列長を示します。

Modbus のアドレス指定 (16 進フォーマット)

アドレスタイプ	10 進範囲	データ型	アクセス
出力コイル	H000001-H010000	Boolean	読み取り/書き込み
入力コイル	H100001-H110000	Boolean	読み取り専用
内部レジスタ	H300001-H310000 H300001-H30FFFF H300001-H30FFFD H3yyyyy.0/1-H3yyyyy.F/10 H300001.2H-H3FFFF.240H H300001.2L-H3FFFF.240L	**Word**、Short、BCD Float、DWord、Long、LBCD Double **Boolean** **String** **String**	読み取り専用* 読み取り専用* 読み取り専用* 読み取り専用* 読み取り専用 読み取り専用
保持レジスタ	H400001-H410000 H400001-H40FFFF H400001-H40FFFD H4yyyyy.0/1-H4yyyyy.F/10 H400001.2H-H4FFFF.240H H400001.2L-H4FFFF.240L	**Word**、Short、BCD Float、DWord、Long、LBCD Double **Boolean** String String	読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り/書き込み
グローバルデータ	HG01-HG20 HG01-HG1F HG01-HG1D HGyy.0/1-HGyy.F/10	**Word**、Short、BCD Float、DWord、Long、LBCD Double **Boolean**	読み取り/書き込み 読み取り/書き込み 読み取り/書き込み 読み取り専用

*Modbus サーバーデバイスの場合、これらの位置は読み取り/書き込みが可能です。
**ピリオドの後ろのビット番号は2から240バイトの範囲の文字列長を示します。

パックコイル

パックタイプのコイルアドレスでは、複数の連続するコイルにアナログ値としてアクセスできます。この機能は、Modbus モデルが Modbus クライアントモードの場合にのみ使用することができます。構文は以下のとおりです。

出力コイル: 0*xxxxx#nn*
入力コイル: 1*xxxxx#nn*

ここで

- *xxxxx* は 1 つ目のコイルのアドレスです。10 進数値と16 進数値の両方を使用できます。
- *nn* はアナログ値にパックされるコイルの数です。有効な範囲は 1-16 であり、10 進数値のみを使用できます。

注記: 有効な唯一のデータ型が Word です。出力コイルでは読み取り/書き込みのアクセスが可能であり、入力コイルでは読み取り専用のアクセスが可能です。開始アドレスがアナログ値の最下位ビット (LSB) となるビットオーダーになります。

書き込み専用アクセス

("W40001" などのように) アドレスの先頭に "W" を付けることによって、すべての読み取り/書き込み可能アドレスを書き込み専用として設定でき、これによってドライバーはレジスタの指定したアドレスを読み取れなくなります。クライアントが書き込み専用タグを読み取ろうとすると、指定したアドレスへの最後に成功した書き込みの値が取得されます。成功した書き込みがない場合、クライアントは数値/文字列値の初期値である 0/NULL を受信します。

警告: 書き込み専用タグのクライアントアクセス権限を読み取り専用に設定した場合、これらのタグへの書き込みは失敗し、クライアントは数値/文字列値として必ず 0/NULL を受信します。

メールボックスモード

メールボックスモードでは保持レジスタのみがサポートされます。クライアントから読み取る場合、データは物理デバイスからではなく、キャッシュからローカルに読み取られます。クライアントから書き込む場合、データはデバイス ID のルーティングパスによって指定されている物理デバイスとローカルキャッシュの両方に書き込まれます。詳細については、**メールボックスモード**を参照してください。

注記: Double データ型はサポートされません。

文字列のサポート

Modbus モデルでは保持レジスタメモリを ASCII 文字列として読み書きできます。文字列データに保持レジスタを使用している場合、各レジスタに 2 バイトの ASCII データが格納されます。文字列を定義する際に、そのレジスタにおける ASCII データの順序を選択できます。文字列の長さは 2 から 240 バイトの範囲で指定でき、ビット番号の位置に入力します。この長さは偶数として入力する必要があります。"H" または "L" をアドレスに追加することでバイトオーダーが指定されます。

例

- 40200 で開始し、長さが 100 バイト、HiLo バイトオーダーの文字列をアドレス指定するには、"40200.100H" と入力します。
- 40500 で開始し、長さが 78 バイト、LoHi バイトオーダーの文字列をアドレス指定するには、"40500.78L" と入力します。

注記: デバイスで許可される書き込み要求の最大サイズによって文字列の長さが制限されることがあります。文字列タグを使用しているときに、サーバーのイベントウィンドウで「デバイス <デバイス> のアドレス <アドレス> に書き込めません: デバイスは例外コード 3 を返しました」というエラーメッセージを受信した場合、文字列の長さがそのデバイスに適していませんでした。可能な場合には文字列を短くする必要があります。

配列のサポート

内部レジスタと保持レジスタの位置 (Boolean と String を除くすべてのデータ型) および入力コイルと出力コイル (Boolean データ型) では配列がサポートされています。配列のアドレスを指定するには 2 つの方法があります。次の例は保持レジスタに当てはまります。

4*xxxx* [行数] [列数]
4*xxxx* [列数] (行数は 1 であるものと見なされます)。

Word、Short、BCD 配列の場合、ベースアドレス + (行数 * 列数) が 65536 を超えてはなりません。Float、DWord、Long、および Long BCD 配列の場合、ベースアドレス + (行数 * 列数 * 2) が 65535 を超えてはなりません。すべての配列で、要求されるレジスタの総数が、このデバイスに指定された保持レジスタのブロックサイズを超えてはなりません。

注記: グローバルデータのベースアドレスが 32 を超えてはなりません。

ファンクションコードの説明

10 進	16 進	説明
01	0x01	コイルのステータスを読み取り
02	0x02	入力ステータスを読み取り
03	0x03	保持レジスタを読み取り
04	0x04	内部レジスタを読み取り
05	0x05	単一コイルを適用
06	0x06	単一レジスタをプリセット
15	0x0F	複数コイルを適用
16	0x10	複数レジスタをプリセット
22	0x16	レジスタへのマスク書き込み

グローバルデータ通信のためのデバイスの設定

グローバルデータはSA85 インタフェースカードによってサポートされています。これには単一のネットワークからのみアクセスできます。たとえば、"7.0.0.0.0" はグローバルデータにアクセスできますが、"7.1.0.0.0" はアクセスできません。

注記: 非送信請求モードではグローバルデータはサポートされません。

デバイスへのグローバルデータの書き込み

PLC から見たホスト PC のアドレスは2.0.0.0.0 です。ホスト PC から見たPLC のアドレスは9.0.0.0.0 です。これはデバイス ID のパスです。ユーザーはデバイスが読み書き可能なアドレスをプログラミングソフトウェアで設定する必要があります。

制御ブロック

レジスタ	コンテンツ	説明
Control [1]	5	グローバルデータを書き込むためのファンクションコード
Control [2]	- 0 = エラーなし	エラーコード。これは変更できません。
Control [3]	32	ソリッドステート RAM からグローバルメモリに書き込む Word の数。最大 32 ビット。
Control [4]	-	予約済み*
Control [5]	2	データの送信先となる Modbus Plus のノードアドレス
Control [6]	0	ホスト PC へのパス
Control [7]	0	ホスト PC へのパス
Control [8]	0	ホスト PC へのパス
Control [9]	0	ホスト PC へのパス

*このレジスタはアプリケーション固有です。

データ領域

レジスタ	コンテンツ	説明
DataField [1]-DataField [32]	データ	なし

デバイスからのグローバルデータの読み取り

PLC から見たホスト PC のアドレスは2.0.0.0.0 です。ホスト PC から見たPLC のアドレスは9.0.0.0.0 です。これはデバイス ID のパスです。

制御ブロック

レジスタ	コンテンツ	説明
Control [1]	6	グローバルデータを読み取るためのファンクションコード
Control [2]	- 0 = エラーなし	エラーコード。これは変更できません
Control [3]	32	ソリッドステート RAM からグローバルメモリに書き込む Word の数。最大 32 ビット。
Control [4]	-	予約済み*
Control [5]	2	データの読み取り元である Modbus Plus のノードアドレス
Control [6]	0	ホスト PC へのパス
Control [7]	0	ホスト PC へのパス
Control [8]	0	ホスト PC へのパス
Control [9]	0	ホスト PC へのパス

*このレジスタはアプリケーション固有です。

データ領域

レジスタ	コンテンツ	説明
DataField [1]-DataField [32]	データ	なし

TIO モジュールのアドレス指定

このモデルではメールボックスモードはサポートされていません。

10 進での TIO モジュールのアドレス指定

アドレスタイプ	範囲	データ型	アクセス
データ I/O*	400001 400001.0/1-400001.15/16**	**Word**、Short **Boolean**	読み取り/書き込み 読み取り/書き込み
データ入力 - ラッチ	400257 400257.0/1-400257.15/16**	**Word**、Short **Boolean**	読み取り専用 読み取り専用
モジュールのタイムアウト	461441 461441.0/1-461441.15/16**	**Word**、Short **Boolean**	読み取り/書き込み 読み取り/書き込み
モジュールのステータス	463489-463497 4xxxxx.0/1-4xxxxx.15/16**	**Word**、Short **Boolean**	読み取り専用 読み取り専用
モジュールの ASCII ヘッダー	464513	String	読み取り専用

*データ I/O 位置から読み取られる値はモジュールの入力レジスタからのものです。この位置に書き込む際は、送信された値によってモジュールの出力レジスタが修正されます。このため、この位置で読み取られた値は、この位置に以前に書き込まれた値と一致しません。
***詳細については、**設定**の「ゼロベースと 1 ベースのアドレス指定」を参照してください。*

16 進での TIO モジュールのアドレス指定

アドレスタイプ	範囲	データ型	アクセス
データ I/O*	H40001 H40001.0/1-H40001.F/10	**Word**、Short **Boolean**	読み取り/書き込み 読み取り/書き込み

アドレスタイプ	範囲	データ型	アクセス
データ入力 - ラッチ	H40101 H40101.0/1-40101.F/10	**Word**、Short **Boolean**	読み取り専用 読み取り専用
モジュールのタイムアウト	H4F001 H4F001.0/1-H4F001.F/10	**Word**、Short **Boolean**	読み取り/書き込み 読み取り/書き込み
モジュールのステータス	H4F801-H4F809 H4yyyy.0/1-H4yyyy.F/10	**Word**、Short **Boolean**	読み取り専用 読み取り専用
モジュールの ASCII ヘッダー	H4FC01	String	読み取り専用

*データ I/O 位置から読み取られる値はモジュールの入力レジスタからのものです。この位置に書き込む際は、送信された値によってモジュールの出力レジスタが修正されます。このため、この位置で読み取られた値は、この位置に以前に書き込まれた値と一致しません。

イベントログメッセージ

次の情報は、メインユーザーインタフェースの「イベントログ」枠に記録されたメッセージに関するものです。「イベントログ」詳細ビューのフィルタリングとソートについては、OPC サーバーのヘルプを参照してください。サーバーのヘルプには共通メッセージが多数含まれているので、これらも参照してください。通常は、可能な場合、メッセージのタイプ (情報、警告) とトラブルシューティングに関する情報が提供されています。

ブロックに不良アドレスがあります。| ブロック範囲 = <開始> から <終了>。

エラータイプ:
エラー

考えられる原因:
指定されたデバイスに存在しない位置を参照しようとしました。

解決策:
デバイスに割り当てられているすべてのタグのアドレスを確認し、無効な位置を参照するタグを削除してください。

ブロックに不良アドレスがあります。| ブロック範囲 = H<開始> ～ H<終了>。

エラータイプ:
エラー

考えられる原因:
指定されたデバイスに存在しない位置を参照しようとしました。

解決策:
デバイスに割り当てられているすべてのタグのアドレスを確認し、無効な位置を参照するタグを削除してください。

MBPLUS.SYS デバイスを開始できません。

エラータイプ:
エラー

考えられる原因:
MBPLUS.SYSドライバーが適切に設定されていません。

解決策:
「コントロールパネル」|「デバイス」アプレットを使用してMBPLUS デバイスを手動で開始および停止できることを確認してください。MBPLUS.SYSドライバーを手動で開始できる場合、modbus_unsolicited.dllドライバーでもこのドライバーを開始できます。

カードを検出できないかModbus Plus サービスを開始できません。カードとMBP *.sysドライバーが適切にインストールされていることを確認してください。

エラータイプ:
エラー

このドライバーの実行に必要なシステムリソースを作成できません。

エラータイプ:
エラー

チャネルを初期化できません。

エラータイプ:
エラー

不良配列。| 配列範囲 = <開始> ～ <終了>。

エラータイプ:
エラー

考えられる原因:
アドレスの配列がアドレス空間の末端を超えています。

解決策:
デバイスのメモリ空間のサイズを確認し、配列長を適切に再定義してください。

チャネルをロードできません。1 つの Hilscher アダプタにつき 1 つのチャネルのみが許可されます。各チャネルが独自のアダプタを持つようにプロジェクトを修正してから再ロードしてください。

エラータイプ:
エラー

タグデータベースのインポート用のファイルを開くときにエラーが発生しました。| OS エラー = '<エラー>'。

エラータイプ:
エラー

MBPLUS パスを開いているときにエラーが発生しました。| パス = '<パス>'。

エラータイプ:
警告

考えられる原因:

1. MBPLUS.SYS ドライバーが適切に設定されていません。
2. ドライバーは指定されたアダプタでパスを開くことができません。

解決策:

1. MBPLUS ドライバーのインストールと設定に関する手順に従ってください。
2. 同じアダプタ番号に割り当てられているチャネルが 8 つ以下であることを確認してください。

受信したブロック長が予想ブロック長と一致しません。| 受信した長さ = <数値> (バイト)、予想される長さ = <数値> (バイト)。

エラータイプ:
警告

デバイスからグローバルデータを取得できません。

エラータイプ:
警告

デバイスからグローバルデータを読み取り中にエラーが発生しました。

エラータイプ:
警告

デバイスに対するブロック要求で例外が返されました。| ブロック範囲 = <開始> から <終了>、例外 = <コード>。

エラータイプ:
警告

考えられる原因:
要求されたノードが応答しませんでした。

解決策:
ケーブル接続、配線、ピンを確認してください。

関連項目:
Hilscher CIF 例外コード

デバイスのアドレスに書き込めません。デバイスは例外を返しました。| アドレス = '<アドレス>'、例外 = <コード>。

エラータイプ:
警告

考えられる原因:
例外コードの説明については、「Modbus 例外コード」を参照してください。

解決策:
「Modbus 例外コード」を参照してください。

関連項目:
Modbus 例外コード

デバイスのアドレスから読み取れません。デバイスは例外を返しました。| アドレス = '<アドレス>'、例外 = <コード>。

エラータイプ:
警告

考えられる原因:
例外コードの説明については、「Modbus 例外コード」を参照してください。

解決策:
「Modbus 例外コード」を参照してください。

関連項目:
Modbus 例外コード

ブロックアドレスの要求で例外が返されました。| ブロック範囲 = H<開始> ～ H<終了>、例外 = <コード>。

エラータイプ:
警告

考えられる原因:
要求されたノードが応答しませんでした。

解決策:
ケーブル接続、配線、ピンを確認してください。

関連項目:

Hilscher CIF 例外コード

警告: グローバルデータが無効です。アクセスするにはModiconの4.0低レベルシステムドライバーが必要です。

エラータイプ:
警告

アダプタを開くことができません。| アダプタ = <名前>。

エラータイプ:
警告

メモリリソース量の低下によりタグインポートが失敗しました。

エラータイプ:
警告

考えられる原因:
ドライバーは変数インポートファイルの処理に必要なメモリを割り当てることができません。

解決策:
不要なアプリケーションをすべて終了してから、もう一度試してください。

タグのインポート中にファイル例外が発生しました。

エラータイプ:
警告

考えられる原因:
変数インポートファイルを読み取れませんでした。

解決策:
変数インポートファイルを再生成してください。

インポートファイルのレコードの解析でエラーが発生しました。| レコード番号 = <数値>、フィールド = <数値>。

エラータイプ:
警告

考えられる原因:
変数インポートファイルの指定されたフィールドが予想より長いか無効なため、解析できませんでした。

解決策:
変数インポートファイルを編集して、問題のあるフィールドを変更してください。

インポートファイルのレコードの説明が切り詰められました。| レコード番号 = <数値>。

エラータイプ:
警告

考えられる原因:
指定されたレコード内のタグの説明が長すぎます。

解決策:
ドライバーは必要に応じて説明を切り詰めます。このエラーを防止するには、変数インポートファイルを編集して、説明を短くしてください。

インポートされたタグ名が無効のため変更されました。| タグ名 = '<タグ>'、変更後のタグ名 = '<タグ>'。

エラータイプ:
警告

考えられる原因:
変数インポートファイル内のタグ名に無効な文字が含まれていました。

解決策:
ドライバーは変数インポートファイルに基づいて有効な名前を構築します。このエラーを防止し、名前の一貫性を維持するには、エクスポートされた変数の名前を変更してください。

データ型がサポートされていないため、タグをインポートできませんでした。| タグ名 = '<タグ>'、サポートされていないデータ型 = '<タイプ>'。

エラータイプ:
警告

考えられる原因:
変数インポートファイルで指定されたデータ型は、このドライバーでサポートされている型ではありません。

解決策:
変数インポートファイルで指定されているデータ型を、サポートされているいずれかの型に変更してください。構造体の変数である場合、ファイルを手動で編集して構造体に必要な各タグを定義するか、サーバーで必要なタグを手動で設定してください。

関連項目:
Concept からの変数のエクスポート

デバイスのアドレスに書き込めません。ボードは例外を返しました。| アドレス = '<アドレス>'、例外 = <コード>。

エラータイプ:
警告

考えられる原因:

1. アダプタが存在しない可能性があります。
2. 返されたエラーコードによって異なります。

解決策:
チャネルプロパティで適切なアダプタ番号が選択されていることを確認してください。アダプタオーダリングを指定するには SyCon を使用します。

注記:
SA85 カードには適用されません。Hilscher CIF カードの場合はコード -1、-33 を使用します。

関連項目:
SyCon ユーザーマニュアル

デバイスのアドレスから読み取れません。ボードは例外を返しました。| アドレス = '<アドレス>'、例外 = <コード>。

エラータイプ:
警告

考えられる原因:

1. アダプタが存在しない可能性があります。
2. 返されたエラーコードによって異なります。

解決策:
チャネルプロパティで適切なアダプタ番号が選択されていることを確認してください。アダプタオーダリングを指定するには SyCon を使用します。

注記:
SA85 カードには適用されません。Hilscher CIF カードの場合はコード -1、-33 を使用します。

関連項目:
SyCon ユーザーマニュアル

MBPLUS.SYS デバイスを開始

エラータイプ:
情報

タグデータベースをインポートしています。| ソースファイル = '<ファイル名 >'

エラータイプ:
情報

Modbus 例外コード

以下のデータは Modbus Application Protocol Specifications ドキュメントからのものです。

コード 10 進 /16 進	名前	意味
01/0x01	ILLEGAL FUNCTION (不正なファンクション)	クエリーで受信したファンクションコードを、サーバーに対する操作として実行することはできません。このファンクションコードは新しいデバイスにだけ適用できるか、選択したユニットに実装されていないことが原因である可能性があります。また、サーバーが、このタイプの要求を処理する状態になっていない可能性もあります (レジスタ値を返す必要があるにもかかわらず、サーバーがそのように設定されていない場合など)。
02/0x02	ILLEGAL DATA ADDRESS (不正なデータアドレス)	クエリーで受信したデータアドレスを、サーバーに対するアドレスとして使用することはできません。具体的には、参照番号と転送長さの組み合わせが無効です。レジスタが 100 個あるコントローラの場合、オフセット 96 と長さ 4 の要求では成功します。オフセット 96 と長さ 5 の要求では例外 02 が生成されます。
03/0x03	ILLEGAL DATA VALUE (不正なデータ値)	クエリーデータフィールドに含まれている値を、サーバーに対する値として使用することはできません。これは、示された長さが正しくないなど、複合型要求の残りの構造体に誤りがあることを示しています。Modbus プロトコルでは個々のレジスタのそれぞれの値の有意性は認識されないので、これはレジスタのストレージにサブミットされたデータアイテムの値がアプリケーションプログラムでの予想の範囲外であることを必ずしも意味しません。
04/0x04	SERVER DEVICE FAILURE (サーバーデバイスの障害)	サーバーが要求された操作を実行しようとしたときに回復不可能なエラーが発生しました。
05/0x05	ACKNOWLEDGE	サーバーは要求を受け入れて処理していますが、処理が完了するまで時間がかかります。クライアントでタイムアウトエラーが発生しないようにするために、この応答が返されます。クライアントは次にプログラム完了ポーリングメッセージを送信して、処理が完了したかどうかを判断します。
06/0x06	SERVER DEVICE BUSY (サーバーデバイスがビジー状態)	サーバーは、時間がかかるプログラムコマンドを処理しています。クライアントは、サーバーによる処理の完了後にメッセージを再送信する必要があります。
07/0x07	NEGATIVE	サーバーは、クエリーで受信したプログラムファンクションを実行できません。このコー

コード 10 進 /16 進	名前	意味
	ACKNOWLEDGE (否定応答)	ドはファンクションコード 13 または 14 (10 進) を使用したプログラミング要求が成功しなかった場合に返されます。クライアントは、サーバーに対して診断情報またはエラー情報を要求する必要があります。
08/0x08	MEMORY PARITY ERROR (メモリパリティエラー)	サーバーが拡張メモリを読み取ろうとしたときに、メモリ内でパリティエラーが検出されました。クライアントはこの要求を再試行できますが、サーバーデバイス上でサービスが必要になる場合があります。
10/0x0A	GATEWAY PATH UNAVAILABLE (ゲートウェイパスを使用できません)	ゲートウェイが使用されている場合、ゲートウェイが要求を処理するために入力ポートから出力ポートへの内部通信パスを割り当てることができなかったことを示します。これは通常、ゲートウェイの設定に誤りがあるかオーバーロードされていることを意味します。
11/0x0B	GATEWAY TARGET DEVICE FAILED TO RESPOND (ゲートウェイのターゲットデバイスが応答しませんでした)	ゲートウェイが使用されている場合、ターゲットデバイスから応答がなかったことを示します。これは通常、デバイスがネットワーク上に存在しないことを意味します。

注記: このドライバーでは、「サーバー」と「非送信請求」という用語は同じ意味になります。

索引

1

B

C

D

F

I

L

M

P

S

T

W

あ

い

え

か

き

く

こ

さ

し

す

せ

た

ち

て

と

は

ふ

ほ

め

も

ら

れ

漢字