

# Modbus イーサネットドライバー

© 2020 PTC Inc. All Rights Reserved.

# 目次

<b>Modbus イーサネットドライバ</b> .....	1
<b>目次</b> .....	2
Modbus イーサネットドライバ .....	5
<b>概要</b> .....	5
<b>サポートされるデバイスモデル</b> .....	5
<b>デバイスの設定</b> .....	6
デバイスのプロパティ - 一般 .....	7
デバイスのプロパティ - スキャンモード .....	8
デバイスのプロパティ - タイミング .....	9
デバイスのプロパティ - 自動格下げ .....	10
デバイスのプロパティ - タグ生成 .....	10
デバイスのプロパティ - 変数のインポート設定 .....	12
デバイスのプロパティ - 非送信請求 .....	12
Modbus マスターと Modbus 非送信請求の考慮事項 .....	14
デバイスのプロパティ - エラー処理 .....	14
デバイスのプロパティ - イーサネット .....	15
デバイスのプロパティ - 設定 .....	15
デバイスのプロパティ - ブロックサイズ .....	18
デバイスのプロパティ - 冗長 .....	20
チャンネルのプロパティ - 構成 API .....	20
デバイスのプロパティ - 構成 API .....	20
構成 API Modbus Ethernet の例 .....	22
<b>自動タグデータベース生成</b> .....	23
カスタムアプリケーションからのインポート .....	23
<b>Modbus イーサネット通信の最適化</b> .....	24
<b>データ型の説明</b> .....	25
<b>アドレスの説明</b> .....	25
ドライバのシステムタグのアドレス指定 .....	26
ファンクションコードの説明 .....	26
Applicom のサブモデルとアドレス指定 .....	27
ジェネリック Modbus のアドレス指定 .....	27
TSX Quantum .....	30
TSX Premium .....	33
CEG のアドレス指定 .....	35
Fluenta のアドレス指定 .....	35
Instromet のアドレス指定 .....	36
メールボックスのアドレス指定 .....	36
Modbus のアドレス指定 .....	37
Roxar のアドレス指定 .....	40
<b>イベントログメッセージ</b> .....	41

Winsock 通信を開始できませんでした。 .....	41
非送信請求通信を開始できませんでした。 .....	41
未定義のデバイスに対して非送信請求メールアドレスアクセスが行われました。ソケットを閉じています。  IP アドレス = '<アドレス>'。 .....	41
受信した要求は非送信請求メールアドレスでサポートされていません。  IP アドレス = '<アドレス>'。 .....	41
非送信請求メールアドレスのメモリ割り当てエラー。  IP アドレス = '<アドレス>'。 .....	42
ソケット接続を作成できません。 .....	42
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>'。 .....	42
不良配列。  配列範囲 = <開始> ~ <end>。 .....	42
ブロックに不良アドレスがあります。  ブロック範囲 = <アドレス> ~ <address>。 .....	42
ホストの解決に失敗しました。  ホスト名 = '<名前>'。 .....	43
指定された出カコイルブロックサイズは最大ブロックサイズを超えています。  指定されたブロックサイズ = <数 値> (コイル)、最大ブロックサイズ = <数値> (コイル)。 .....	43
指定された入カコイルブロックサイズは最大ブロックサイズを超えています。  指定されたブロックサイズ = <数 値> (コイル)、最大ブロックサイズ = <数値> (コイル)。 .....	43
指定された内部レジスタブロックサイズは最大ブロックサイズを超えています。  指定されたブロックサイズ = < 数値> (レジスタ)、最大ブロックサイズ = <数値> (レジスタ)。 .....	43
指定された保持レジスタブロックサイズは最大ブロックサイズを超えています。  指定されたブロックサイズ = < 数値> (レジスタ)、最大ブロックサイズ = <数値> (レジスタ)。 .....	43
ブロック要求で例外が返されました。  ブロック範囲 = <アドレス> ~ <address>、例外 = <コード>。 .....	43
ブロック要求で例外が返されました。  ブロック範囲 = <アドレス> ~ <address>、関数コード = <コード>、例 外 = <コード>。 .....	44
受信したブロックの長さは不適切です。  ブロック範囲 = <開始> ~ <end>。 .....	44
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。 .....	44
タグのインポート中にファイル例外が発生しました。 .....	44
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 = <field>。 .....	44
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>。 .....	45
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>'。 .....	45
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>'。 .....	45
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>。 .....	45
イーサネットマネージャが開始されました。 .....	46
イーサネットマネージャが停止しました。 .....	46
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>'。 .....	46
クライアントアプリケーションはシステムタグ _CEGExtension を介して CEG 拡張を変更しました。  拡張 = '< 拡張>'。 .....	46
非送信請求通信を開始しています。  プロトコル = '<名前>'、ポート = <数値>。 .....	46
スレーブデバイス用のメモリが作成されました。  スレーブデバイス ID = <デバイス>。 .....	46
すべてのチャンネルが仮想ネットワークを利用しているため、非送信請求通信を停止しています。 .....	46
チャンネルは仮想ネットワーク内にあるため、すべてのデバイスがデバイスにつき 1 つのソケットを使用する設定に 戻りました。 .....	46
接続されているクライアントでデバイス ID を 'マスター' から 'スレーブ' に変更できません。 .....	46
接続されているクライアントでデバイス ID を 'スレーブ' から 'マスター' に変更できません。 .....	47
チャンネルが仮想ネットワーク内にある場合、スレーブモードは許可されません。デバイス ID にループバックアド レスまたはローカル IP アドレスが含まれてはなりません。 .....	47
チャンネルが仮想ネットワーク内にある場合、メールアドレスモデルは許可されません。 .....	47

Modbus 例外コード .....	48
<b>索引</b> .....	<b>49</b>

## Modbus イーサネットドライバー

---

ヘルプバージョン [1.135](#)

### 目次

#### 概要

Modbus イーサネットドライバーとは

#### チャンネルとデバイスの設定

このドライバーを使用するためにデバイスを構成する方法

#### APIを使用した設定

構成 API を使用してチャンネルとデバイスを設定する方法

#### 自動タグデータベース生成

Modbus イーサネットドライバー用にタグを設定する方法

#### 通信の最適化

Modbus イーサネットドライバーから最高のパフォーマンスを得る方法

#### データ型の説明

Modbus イーサネットドライバーでサポートされるデータ型

#### アドレスの説明

Modbus イーサネットデバイスでデータ位置を参照する方法

#### イベントログメッセージ

Modbus イーサネットドライバーで生成されるメッセージ

### 概要

---

Modbus イーサネットドライバーは Modbus イーサネットデバイスが 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。このドライバーを使用するためには、ユーザーは TCP/IP を適切にインストールする必要があります。セットアップの詳細については、*Windows のドキュメント* を参照してください。

● **注記:** 処理中にエラーが発生した場合、このドライバーはメッセージを通知します。

### サポートされるデバイスモデル

---

#### Applicom

このモデルはジェネリック Modbus、TSX Premium、および TSX Quantum デバイス用に Applicom アドレス指定構文をサポートしています。

#### イーサネットから Modbus Plus へのブリッジ

このドライバーはイーサネットから Modbus Plus へのブリッジを介して Modbus Plus デバイスと通信できます。使用するデバイス ID は、ブリッジの IP アドレスと Modbus Plus ブリッジインデックスを組み合わせたものになります。たとえば、ブリッジ IP が 205.167.7.12、ブリッジインデックスが 5 の場合、デバイス ID は 205.167.7.12.5 となります。MBE から MBP へのブリッジを取得して設定する方法については、Modicon/Schneider Automation の販売代理店までお問い合わせください。

#### CEG

このモデルは CEG デバイスの拡張ブロックサイズをサポートしています。

#### Fluenta

このモデルは Fluenta FGM 100/130 フローコンピュータの非標準 Modbus マッピングをサポートしています。

#### Instromet

このモデルは Instromet デバイスの非標準 Modbus マッピングをサポートしています。

## メールボックス

このモデルは非送信請求要求の処理方法に影響を与えます。メールボックスデバイスを定義することによって、ドライバーはネットワーク上の PLC として機能しなくなります。代わりに、定義された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ようになります。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 IP アドレスを検出し、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。IP アドレ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。このタイプのデバイスを読み書きするクライアントアプリケーションは、物理デバイス内ではなくドライバー内のストレージ領域を読み書きします。

● **Modbus イーサネットドライバーに非送信請求要求を送信する方法については、Modicon のドキュメントで MSTR 命令のトピックを参照してください。**

● **注記:** Modbus メールボックスではファンクションコード 22 (0x16) はサポートされていません。0x10 (複数の保持レジスタへの書き込み) と 0x6 (単一の保持レジスタへの書き込み) はサポートされています。デバイスのプロパティで「保持レジスタのビット書き込み」を無効化すると、個々のビットに書き込むことが可能です。これにより、ビットに直接書き込む代わりに、読み取り/修正/書き込みシーケンスが使用されます。これが機能するためには、(メールボックスではなく) マスター Modbus デバイスの設定のみを変更する必要があります。

### ● [メールボックスデバイスモデルに対するメールボックスクライアントの権限](#)

## Modbus マスター

ほとんどのプロジェクトは Modbus マスターとして機能するように設定されています。このモードでは、ドライバーは物理デバイスにアクセスします (TSX Quantum や、Modbus オープンイーサネットと互換性があるその他のデバイスなど)。

## Modbus 非送信請求

Modbus がモデルとして選択され、そのデバイス ID としてホストマシンの IP アドレスが設定されている場合、Modbus イーサネットドライバーはネットワーク上のデバイスとして機能します。ドライバーは受信したすべての非送信請求コマンドを受け入れて、別の PLC であるかのようにこれらの処理を試みます。ネットワーク上の Modbus マスターはいずれもその IP アドレスを使用してこのシミュレーション対象のデバイスと通信できます。

YYY.YYY.YYY.YYY.XXX としてスレーブデバイスのデバイス ID が指定されます。YYY には、ループバックアドレスか、そのドライバーを実行している PC のローカル IP アドレスを指定できます。XXX にはスレーブのステーション ID を 0 から 255 の範囲で指定できます。

複数のスレーブデバイスが同じステーション ID を持つことができます。その場合、ステーション ID を共有するすべてのデバイスが 1 つの共通のシミュレーション対象デバイスを指します。存在しないスレーブデバイス (ステーション ID) のデータをリモートマスターが要求した場合、その応答にはステーション 0 からのデータが含まれます。プロジェクト内にスレーブデバイスが作成されると、そのスレーブが有効になり、サーバーがシャットダウンするまで有効なままとなります。ステーション ID を変更すると新しいスレーブデバイスが有効になり、サーバーがシャットダウンするまで有効なままとなります。

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。非送信請求モードでは、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)。さらに、このドライバーにはループバック (ファンクションコード 08、サブコード 00) が実装されています。これらの位置には、ホスト PC からローカルに、スレーブデバイスに割り当てられたタグとしてアクセスできます。

● **注記:** 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。

## Roxar

このモデルは Roxar RFM 含水率メーターの非標準 Modbus マッピングをサポートしています。

## デバイスの設定

チャンネルあたりのデバイスの最大数は 8192 です。デバイスのプロパティのグループの詳細については以下をクリックしてください。

### [一般](#)

### [スキャンモード](#)

### [タイミング](#)

### [自動格下げ](#)

### [タグ生成](#)

変数のインポート設定非送信請求エラー処理サブモデルイーサネット設定ブロック冗長

●**注記:** すべてのグループがすべてのモデルで使用および適用可能なわけではありません。

●**関連項目:** 構成 API コマンドを使用したデバイスの設定、Modbus を使用した API の例

**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。

「チャンネル割り当て」: このデバイスが現在属しているチャン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。

● 特定のデバイスモデルの詳細については、サポートされるデバイスモデルを参照してください。

「モデル」: このデバイスのバージョン。

「ID」: デバイスの IP アドレスおよびイーサネットネットワーク上の Modbus ブリッジインデックスを指定します。デバイス ID は <ホスト>.XXX として指定し、ここで <ホスト> は標準 UNC/DNS 名または IP アドレスです。XXX はデバイスの

Modbus ブリッジインデックスを示し、0 から 255 の範囲になります。ブリッジが使用されていない場合、このインデックスを 0 に設定する必要があります。モデルとデバイス ID によっては、非送信請求デバイスまたはマスターデバイスとして機能するようデバイスを設定できます。

● 非送信請求モードの詳細については、[Modbus 非送信請求](#)を参照してください。

#### 例

1. IP アドレスが 205.167.7.19 である Modicon TSX Quantum デバイスからデータを要求する場合、デバイス ID を 205.167.7.19.0 として入力する必要があります。
2. ブリッジインデックスが 5 である Modbus イーサネットブリッジに接続している、IP アドレスが 205.167.7.50 である Modbus Plus デバイスからデータを要求する場合、デバイス ID を 205.167.7.50.5 として入力する必要があります。

#### 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、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。

● 関連項目: [構成 API コマンドを使用したデバイスの設定](#)、[Modbus を使用した API の例](#)

#### 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。

プロパティグループ	☐ スキャンモード	
一般	スキャンモード	クライアント固有のスキャン速度を適用 ▼
スキャンモード	キャッシュからの初回更新	無効化
タイミング		

「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、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回更新」: 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、初回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。電氣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。

プロパティグループ	通信タイムアウト	
一般	要求のタイムアウト (ミリ秒)	5000
スキャンモード	再試行回数	3
タイミング	タイミング	
自動格下げ	要求間遅延 (ミリ秒)	0

### 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」: このプロパティでは、ターゲットデバイスからの応答を待つのをいつやめるかを判断する際にすべてのドライバーが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 (167.6667 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」: このプロパティでは、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」: このプロパティでは、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、そのチャン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ン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ン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。

プロパティグループ	☑ <b>自動格下げ</b>	
一般	エラー時に格下げ	有効化
スキャンモード	格下げまでのタイムアウト回数	3
タイミング	格下げ期間 (ミリ秒)	10000
<b>自動格下げ</b>	格下げ時に要求を破棄	無効化

「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。

プロパティグループ	<input type="checkbox"/> タグ生成	
一般	デバイス起動時	起動時に生成しない
スキャンモード	重複タグ	作成時に削除
タイミグ	親グループ	
自動格下げ	自動生成されたサブグループを許可	有効化
タグ生成		

「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」: このプロパティでは、OPC タグを自動的に生成する場合を指定します。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、サーバーのタグ空間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。

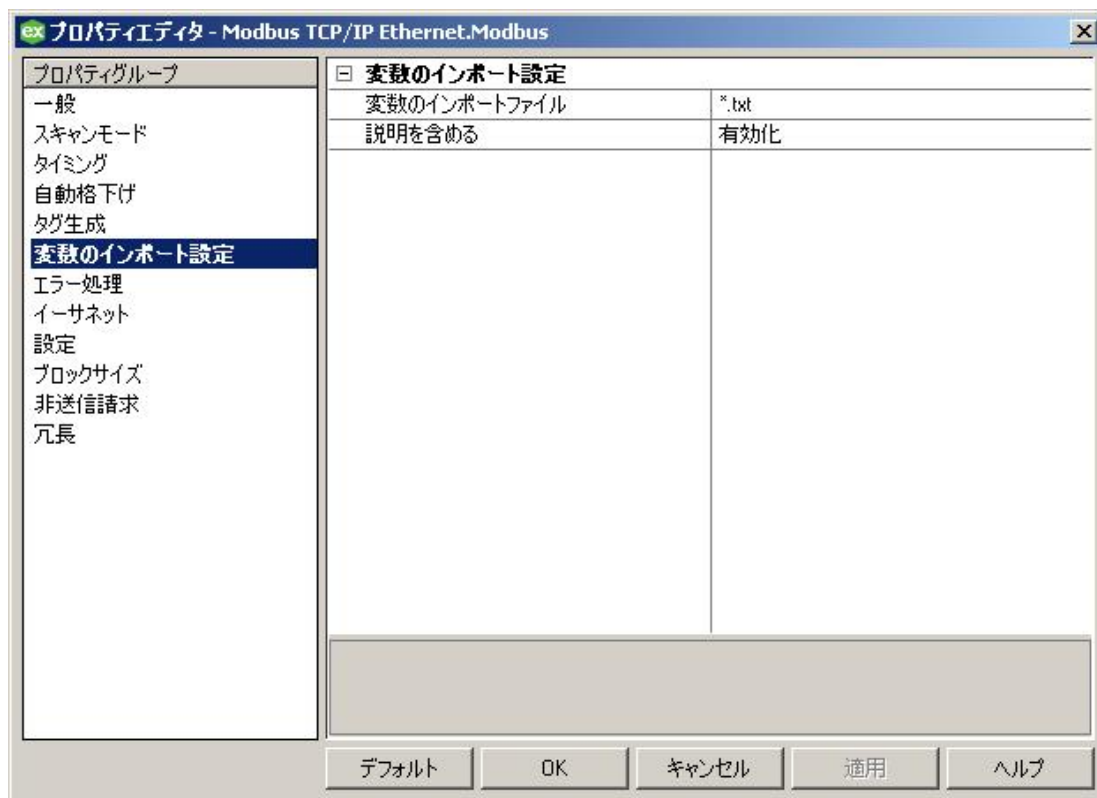
● 注記: サーバーがタグを生成しているときに、タグに既存のタグと同じ名前が割り当てられた場合、タグ名が重複しないようにするため、番号が自動的に 1 つ増分します。たとえば、生成プロセスによってすでに存在する "AI22" という名前のタグが作成された場合、代わりに "AI23" としてタグが作成されます。

「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 - 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。

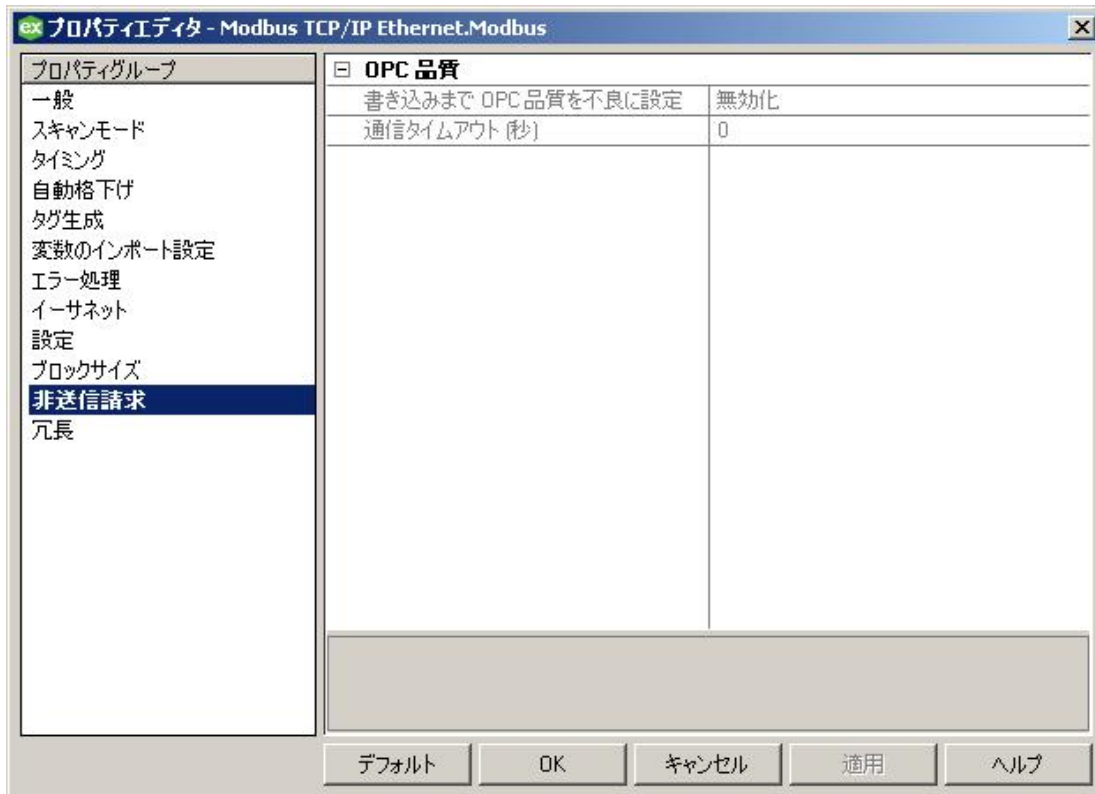


「変数のインポートファイル」: このパラメータでは、自動タグデータベース生成機能が有効になっている場合にこのドライバーが使用する変数インポートファイルの正確な場所を指定します。

「説明を含める」: 有効な場合、タグの説明がインポートされます (ファイル内に存在する場合)。

● 自動タグデータベース生成機能の設定 (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。

## デバイスのプロパティ - 非送信請求



## OPC 品質

「書き込みまで OPC 品質を不良に設定」: このドライバーに関連付けられるタグの初期 OPC 品質を制御します。無効にした場合、すべてのタグの初期値は 0 となり、OPC 品質は良好に設定されます。これがデフォルトの状態です。有効にした場合、すべてのタグの初期値は 0 となり、OPC 品質は不良に設定されます。タグが参照するすべてのコイルまたはレジスタが Modbus マスターまたはクライアントアプリケーションによって書き込まれるまで、タグの品質は不良のままとなります。たとえば、アドレスが 400001 でデータ型が DWord であるタグは 2 つの保持レジスタ 400001 および 400002 を参照します。両方の保持レジスタに書き込まれるまでタグの品質は良好になりません。

● **注記:** デバイスが非送信請求モードでない場合、このオプションは暗色表示されます。

「通信タイムアウト」: ドライバーが受信する要求を待機する時間 (秒) を設定します。この時間が経過すると、ドライバーはそのデバイスのタグの品質を不良に設定します。タイムアウトが発生した後、タイムアウトをリセットしてすべてのタグが通常どおり処理されるようにする唯一の方法としては、リモートマスターとの通信を再確立します。あるいは、これを 0 に設定することで通信タイムアウトを無効にします。有効にする場合、有効な範囲は 1 から 64,800 秒 (18 時間) です。

● **注記:**

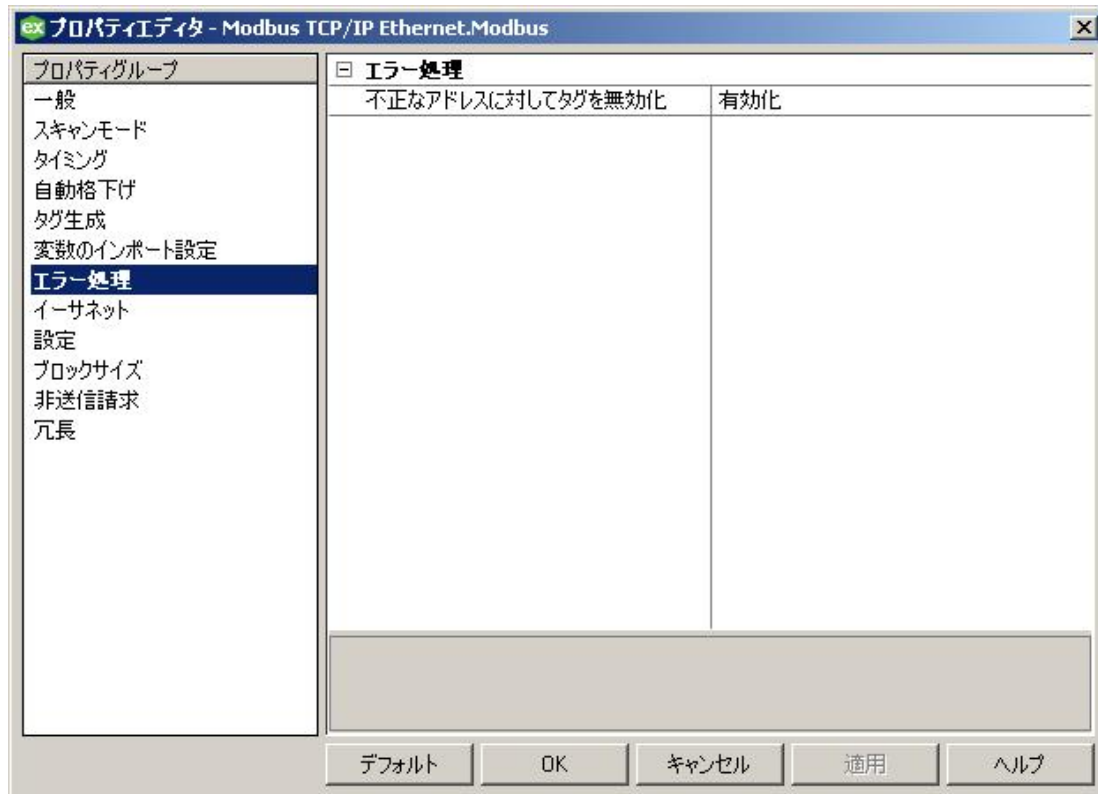
1. 存在しないスレーブデバイス (ステーション ID) への要求を受信した場合、その要求はステーション 0 に転送されます。その場合、タイムアウト期間中にリモート通信を明示的に受信しなかった場合でも、ステーション ID が 0 であるスレーブデバイスでタイムアウトは発生しません。
2. 非送信請求デバイスのモデルは Modbus、デバイス ID は *IP\_Address.yyy* (ここで *IP\_Address* はこのドライバーを実行している PC のローカル IP アドレス) である必要があります。たとえば、127.xxx.xxx.xxx (xxx = 0-255)、yyy (ステーション ID) = 0-255 です。
3. スレーブデバイスへの 1 つ目の非送信請求を受信すると、イベントログに "<日付> \_\_<時刻> \_\_<レベル> \_\_<ソース> \_\_<イベント>" という情報メッセージが表示されます。たとえば、"2/4/2011 4:53:10 PM 情報 Modbus TCP/IP イーサネット \_\_スレーブデバイス <スレーブ番号> 用のメモリが作成されました" と表示されます。
4. このドライバーでは、「スレーブ」と「非送信請求」という用語は同義で用いられています。

## Modbus マスターと Modbus 非送信請求の考慮事項

以下の注意事項は Modbus マスターデバイスと Modbus 非送信請求デバイスの両方に関連します。

- メールボックスデバイスと Modbus デバイスを同じマシンに配置することはお勧めしません。マスターは一度に一方のデバイスからのみデータを取得するので、どちらのデバイスからデータを取得するのか予測できません。
- 非送信請求デバイスの最適なタグ処理を実現するためには、マスターデバイスと非送信請求デバイスをサーバープロジェクト内の別々のチャンネルに配置することをお勧めします。
- クライアントが接続されている場合、デバイスのモードの変更 (マスターからスレーブまたはスレーブからマスター) が生じない場合にかぎりデバイス ID を変更できます。ループバックまたはローカル IP アドレスを別の IP アドレスに変更したかその逆の変更によってモードが変更されました。(ドライバーを実行している PC の) ループバックアドレスとローカル IP アドレスはスレーブ (非送信請求) モードを示し、その他の IP アドレスはデバイスのマスターモードを示しています。クライアントが接続されていない場合、モードを自由に変更できます (マスターからマスター、マスターからスレーブ、スレーブからスレーブ、スレーブからマスターなど)。
  - **注記:** 127.xxx.xxx.xxx というフォーマット (ここで xxx は 0-255 の範囲) のアドレスはループバックアドレスです。
- マスターデバイスとスレーブデバイスで「データエンコーディング」グループの設定が同じである必要があります。たとえば、Modbus マスターとして設定されているデバイスが、Modbus スレーブとして設定されているデバイスと通信している場合です。
- サーバープロジェクト全体では、一意のスレーブ ID ごとに1つずつ、最大255スレーブデバイスを使用できます。同じスレーブ ID を複数のチャンネルで使用することはできません。
- サーバーは、ループバックアドレス (127..x)、または localhost IP をそれ自体への参照として認識し、スレーブ ID に固有の共有メモリ空間を作成します。同じレジスタメモリを使用して、複数チャンネルの同じ ID が同じスレーブデバイスになります。
- 同じスレーブ ID をプロジェクトで複数回使用する必要がある場合は、同じスレーブデバイス id のほかのインスタンスと一致しないタグアドレス範囲を選択します。同じスレーブ ID エクスペリエンスで同じタグアドレス範囲を使用している複数のチャンネル/デバイスが、相互に通信し、データが破損しています。
- このドライバーでは、「スレーブ」と「非送信請求」という用語は同義で用いられています。

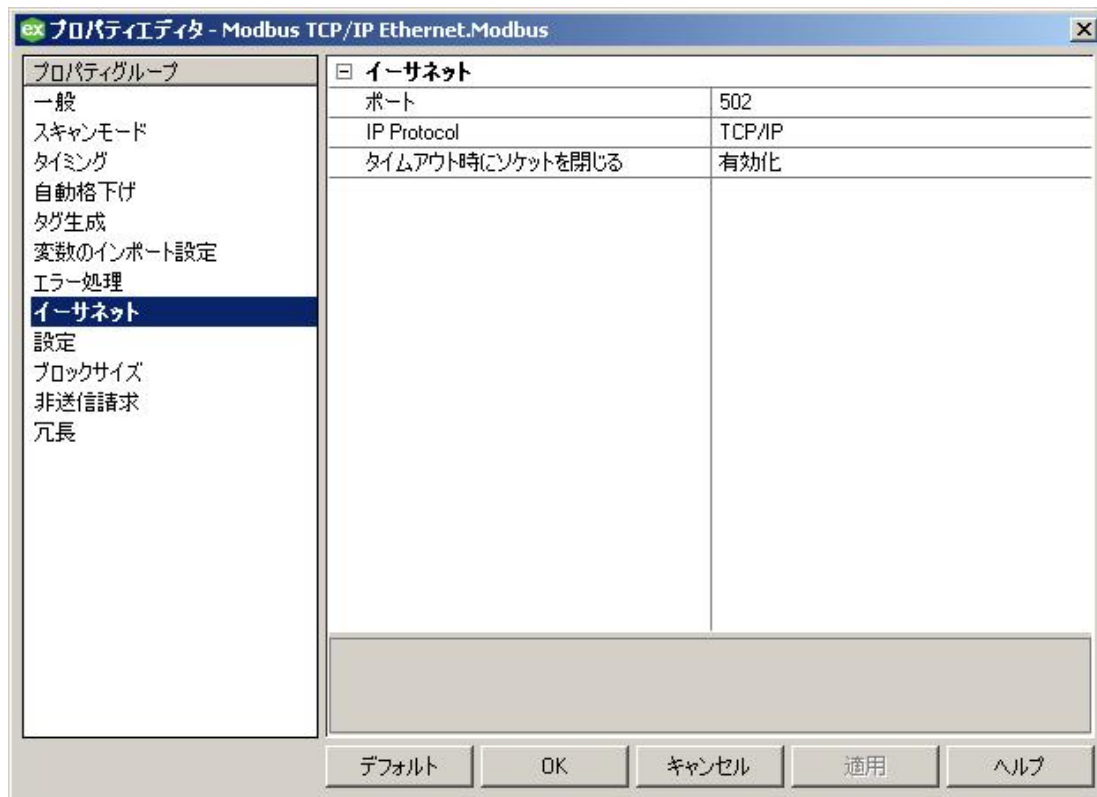
## デバイスのプロパティ - エラー処理



「不正なアドレスでタグを無効化」: デバイスがデータブロックの読み取りに応答して Modbus 例外コード 2 (不正なアドレス) または 3 (ポイント数などの不正なデータ) を返した場合にドライバーがそのブロックのポーリングを停止するには「有効」

化」を選択します。エラーの場合もドライバーがそのデータブロックを引き続きポーリングするには「無効化」を選択します。デフォルトで有効になっています。

## デバイスのプロパティ - イーサネット



「ポート」: リモートデバイスで使用するよう設定されているポート番号を指定します。有効な範囲は0から65535です。デフォルトは502です。このポート番号はデバイスに対して送信請求要求を行う際に使用されます。

● ポートのシステムタグが使用されている場合、ポート番号の設定が変更されます。詳細については、[ドライバーのシステムタグのアドレス](#)を参照してください。

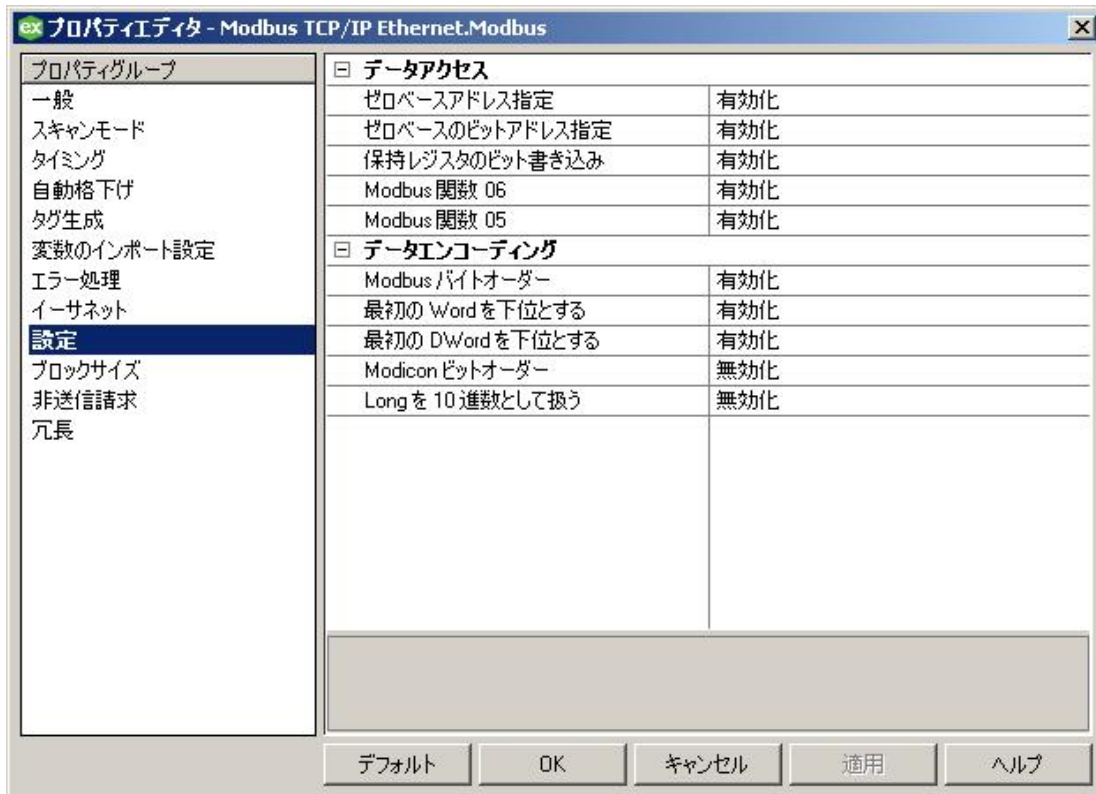
「IP プロトコル」: ドライバーがリモートデバイスに接続する際にユーザーデータグラムプロトコル (UDP) を使用するか伝送制御プロトコル (TCP/IP) を使用するかを指定します。マスターとスレーブで設定が一致する必要があります。たとえば、スレーブの IP プロトコル設定が TCP/IP である場合、そのデバイスでのマスターの IP プロトコル設定も TCP/IP である必要があります。

● 注記: このドライバーでは Winsock V1.1 以上が必要です。

「タイムアウト時にソケットを閉じる」: デバイスがタイムアウトの期限内に回答しなかった場合にドライバーが TCP ソケット接続を閉じるかどうかを指定します。有効にした場合 (デフォルト)、ドライバーはタイムアウトになるとソケット接続を閉じます。無効にした場合、エラーを受信するか、物理デバイスがソケットを閉じるか、ドライバーがシャットダウンするまで、ドライバーは同じ TCP ソケットを使用し続けます。

● 注記: Modbus イーサネットドライバーはソケットエラー時にソケット接続を閉じます。

## 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、デバイスのパラメータを定義する際にこの値を指定できます。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、「無効」を選択します。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**」: レジスタ内で、Word 内のビットをブールとして参照可能なメモリアイプ。アドレス指定の表記は <address> <bit> であり、ここで <bit> は Word 内のビット番号を表します。このオプションによって、ある Word 内の1ビットを2つの方法(ゼロベースまたは1ベース)によってアドレス指定できます。ゼロベースとは最初のビットが0で始まることを意味し(範囲 = 0-15)、1ベースとは最初のビットが1で始まることを意味します(範囲 = 1-16)。

「**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ビットを操作する特別な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。有効になっている場合、この単一レジスタ書き込みの設定に関係なく、レジスタへの書き込みにファンクションコード 0x16 を使用します。無効になっている場合、ドライバーは単一レジスタ書き込みの「Modbus 関数 06」の選択に応じて、ファンクションコード 0x06 または 0x10 を使用します。デフォルトでは無効になっています。

● **注記**: 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**」: このドライバーは、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 Modbus プロトコルファンクションをサポートしています。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ファンクション 06 と 16 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、このドライバーは通常、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 プロトコルを使用する多くのサードパーティデバイスとこれらのデバイスの多くは、保持レジスタへの書き込みに Modbus ファンクション 16 のみをサポートしています。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、この選択を無効にします。

● **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マスク」プロパティがこのオプションよりも優先されます。「保持レジスタのビットマスク」が有効になっている場合、このプロパティにかかわらず、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ファンクションコード 0x06 または 0x10 が使用されません。

**「Modbus 関数 05」:** このドライバーは、出力コイルのデータをターゲットデバイスに書き込む Modbus プロトコルファンクションをサポートしています。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 プロトコルを使用する多くのサードパーティデバイスとこれらのデバイスの多くは、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。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 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。

**「CEG 拡張」:** この Modbus ドライバーは、拡張ブロックサイズをサポートする CEG デバイスカ、CEG モデルで設定されている場合には Modbus デバイスと通信できます。このプロパティは CEG モデルでのみ使用できます。デフォルトは「有効」であり、そのデバイスは拡張ブロックサイズをサポートする CEG デバイスであることを示します。「無効」の場合、そのデバイスは拡張ブロックサイズをサポートしません。

● **注記:** アクティブなクライアント接続が存在する場合でもこのプロパティを修正できます。その場合、このオプションを無効にするとブロックサイズの範囲が変わります。ブロックサイズのプロパティのいずれかが最大値を超えている場合、それらは新しい最大値に自動的に調整されます。

**「メールボックスクライアントの権限」:** Modbus ドライバーは次のオプションでメールボックスクライアントと通信できます。

- **「メモリアップ読み取り専用」:** クライアントアプリケーションはメールボックスメモリアップから読み取りのみが可能です。
- **「メモリアップ読み書き」:** クライアントアプリケーションはメールボックスメモリアップとの間で読み書きが可能です。
- **「デバイス書き込み-メモリアップ読み取り」:** クライアントアプリケーションはデバイスへの書き込みのみが可能であり、読み取りはメモリアップから行います。

## データエンコーディング

**「Modbus バイトオーダー」:** 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。この選択を使用することで、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。デフォルトでは有効になっており、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、このプロパティを無効にすることで Intel フォーマットのデータを読み取ります。

**「最初の Word を下位とする」:** 32 ビット値と、64 ビット値の DWord のデータエンコーディングを設定します。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。ドライバーはこのオプションに基づいて、最初の Word を 32 ビット値の下位 Word または上位 Word として読み取ることができます。デフォルトは「有効」で最初の Word が下位となり、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」:** 64 ビット値のデータエンコーディングを設定します。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。ドライバーは最初の DWord を 64 ビット値の下位 DWord または上位 DWord として読み取ることができます。デフォルトは「有効」で最初の DWord が下位となり、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。

次の例では、そのドライバーが使用しているレジスタ内のビットアドレス指定がゼロベースか 1 ベースかに応じて、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

「Long を 10 進数として扱う」: 有効な場合、ドライバーは倍精度符号なしの Long データ型と DWord データ型を 0 から 99999999 の範囲の値としてエンコード/デコードします。このフォーマットでは各 Word が 0 から 9999 の値を表します。この範囲を超える読み取られた値はクランプされませんが、動作は定義されていません。読み取られた値はすべて [読み取られた値] = 上位 Word \* 10000 + 下位 Word、という式を使用してデコードされます。99999999 より大きい書き込まれた値は最大値にクランプされます。書き込まれた値はすべて、生データ = [書き込まれた値] / 10000 + [書き込まれた値] % 10000 という式を使用してエンコードされます。

### ●設定に関するヒント

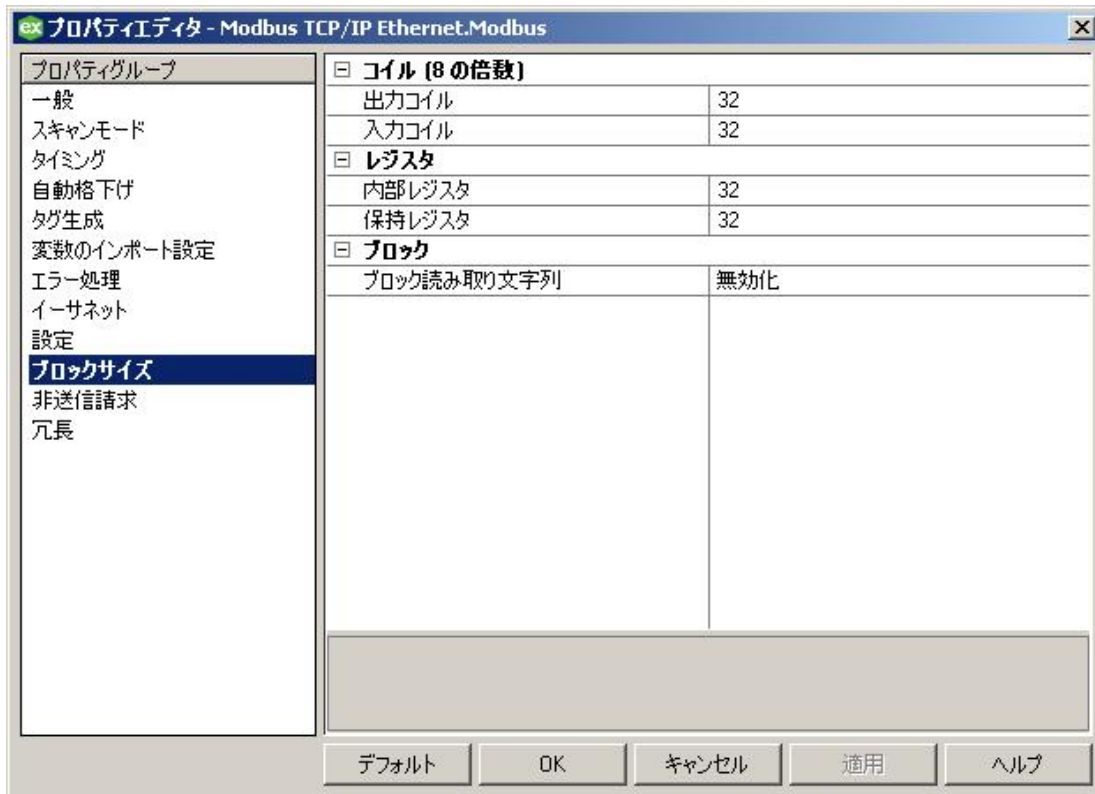
データ型	Modbus バイトオーダー	最初の Word を下位とする	最初の DWord を下位とする
Word、Short、BCD	適用可能	なし	なし
Float、DWord、Long、LBCD	適用可能	適用可能	なし
Double	適用可能	適用可能	適用可能

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。

データエンコーディングのオプション	データエンコーディング	
Modbus バイトオーダー	上位バイト (15..8)	下位バイト (7..0)
Modbus バイトオーダー	下位バイト (7..0)	上位バイト (15..8)
最初の Word を下位とする	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	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	上位 DWord (63..32)	下位 DWord (31..0)
最初の DWord を下位とする	下位 DWord (31..0)	上位 DWord (63..32)

### 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。デフォルトは 32 です。

「**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。デフォルトは 32 です。

## 「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 120 の標準 16 ビット Modbus レジスタを一度に読み取ることができます。デフォルトは 32 です。

「**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 120 の標準 16 ビット Modbus レジスタを一度に読み取ることができます。デフォルトは 32 です。

## 「ブロック」

「**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ブロックで読み取ります。文字列タグは選択した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。

### ● 注記:

1. Instromet、Roxar、および Fluenta モデル (32 ビットおよび 64 ビットレジスタをサポート) では特別な注意が必要です。Modbus プロトコルではブロックサイズが 256 バイト以下に制限されます。これにより、これらのモデルでは最大ブロックサイズが 64 (32 ビットレジスタ) または 32 (64 ビットレジスタ) になります。
2. CEG モデルでサポートされるコイルのブロックサイズは 8 から 8000 の範囲の 8 の倍数であり、レジスタのブロックサイズは 1 から 500 の範囲です。このモデルは必ず CEG デバイスとともに使用する必要があります。
3. レジスタの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。これを防止す

るには、ブロックサイズの値を 120 に減らしてください。

4. 一部のデバイスではデフォルトサイズのブロック読み取り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、Modbus イーサネット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. 一部のデバイスには連続しないアドレスが含まれていることがあります。この場合、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、その要求は却下されることがあります。

## デバイスのプロパティ - 冗長

プロパティグループ	☐ 冗長	
一般	セカンダリパス	
スキャンモード	動作モード	障害時に切り替え
タイミング	モニターアイテム	
冗長	モニター間隔 (秒)	300
	できるだけ速やかにプライマリに...	(はい)

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。

## チャンネルのプロパティ - 構成 API

以下のプロパティは、構成 API サービスを使用してチャンネルを定義します。

### 一般プロパティ

共通。ALLTYPES\_NAME \* 必須パラメータ。

● 注記: このプロパティを変更すると、API エンドポイント URL が変更されます。

common.ALLTYPES\_DESCRIPTION

サーバーメイン。MULTIPLE\_TYPES\_DEVICE\_DRIVER \* 必須パラメータ

サーバーメイン。CHANNEL\_DIAGNOSTICS\_CAPTURE

### イーサネット通信プロパティ

サーバーメイン。CHANNEL\_ETHERNET\_COMMUNICATIONS\_NETWORK\_ADAPTER\_STRING

### 詳細プロパティ

サーバーメイン。CHANNEL\_NON\_NORMALIZED\_FLOATING\_POINT\_HANDLING \* 必須パラメータ

### 書き込み最適化

サーバーメイン。CHANNEL\_WRITE\_OPTIMIZATIONS\_METHOD

サーバーメイン。CHANNEL\_WRITE\_OPTIMIZATIONS\_DUTY\_CYCLE

● 関連項目: サーバーヘルプシステムの構成 API サービスのセクション

## デバイスのプロパティ - 構成 API

以下のプロパティは構成 API サービスを使用してチャンネルを定義します。

### 一般プロパティ

```
common.ALLTYPES_NAME
common.ALLTYPES_DESCRIPTION
servermain.DEVICE_CHANNEL_ASSIGNMENT
servermain.MULTIPLE_TYPES_DEVICE_DRIVER
servermain.DEVICE_MODEL
servermain.DEVICE_ID_STRING
servermain.DEVICE_DATA_COLLECTION
servermain.DEVICE_SIMULATED
```

### スキャンモード

```
servermain.DEVICE_SCAN_MODE * 必須パラメータ
servermain.DEVICE_SCAN_MODE_RATE_MS
servermain.DEVICE_SCAN_MODE_RATE_MS
servermain.DEVICE_SCAN_MODE_PROVIDE_INITIAL_UPDATES_FROM_CACHE
```

### 自動格下げ

```
servermain.DEVICE_AUTO_DEMOTION_ENABLE_ON_COMMUNICATIONS_FAILURES
servermain.DEVICE_AUTO_DEMOTION_DEMOTE_AFTER_SUCESSIVE_TIMEOUTS
servermain.DEVICE_AUTO_DEMOTION_PERIOD_MS
servermain.DEVICE_AUTO_DEMOTION_DISCARD_WRITES
```

### タグ生成

```
servermain.DEVICE_TAG_GENERATION_ON_STARTUP * 必須パラメータ
servermain.DEVICE_TAG_GENERATION_DUPLICATE_HANDLING * 必須パラメータ
servermain.DEVICE_TAG_GENERATION_GROUP
servermain.DEVICE_TAG_GENERATION_ALLOW_SUB_GROUPS
```

● **ヒント:** 自動タグ生成を起動するには、空のボディを持つ PUT をデバイス上の taggeneration サービスエンドポイントに送信します。

● **関連項目:** 詳細については、サービスのヘルプを参照してください。

### タイミング

```
servermain.DEVICE_CONNECTION_TIMEOUT_SECONDS
servermain.DEVICE_REQUEST_TIMEOUT_MILLISECONDS
servermain.DEVICE_RETRY_ATTEMPTS
servermain.DEVICE_INTER_REQUEST_DELAY_MILLISECONDS
```

● **関連項目:** サーバーヘルプシステムの構成 API サービスのセクション

## 構成 API Modbus Ethernet の例

チャンネルとデバイスの定義および列挙のリストについては、REST クライアントを使用して次のエンドポイントにアクセスしてください。

### チャンネル定義

エンドポイント (GET):

```
https://<ホスト名または IP>:<ポート>/config/v1/doc/drivers/Modbus%20TCP%20Ethernet/channels
```

### デバイス定義

エンドポイント (GET):

```
https://<ホスト名または IP>:<ポート>/config/v1/doc/drivers/Modbus%20TCP%20Ethernet/devices
```

### Modbus チャンネルの作成

エンドポイント (POST):

```
https://<ホスト名または IP>:<ポート>/config/v1/project/channels
```

ボディ:

```
{ "common.ALLTYPES_NAME": "MyChannel", "servermain.MULTIPLE_TYPES_DEVICE_DRIVER": "Modbus TCP/IP Ethernet" }
```

関連項目: チャンネルのプロパティのリストについては、付録 A を参照してください。

### Modbus デバイスの作成

エンドポイント (POST):

```
https://<ホスト名または IP>:<ポート>/config/v1/project/channels/MyChannel/devices
```

ボディ:

```
{ "common.ALLTYPES_NAME": "MyDevice", "servermain.DEVICE_ID_STRING": "<192.160.0.1>.0", "servermain.MULTIPLE_TYPES_DEVICE_DRIVER": "Modbus TCP/IP Ethernet" }
```

関連項目: デバイスのプロパティのリストについては、付録 B を参照してください。

### デバイス ID の更新

REST クライアントから "PUT" コマンドを使用して、デバイス ID を更新します。

以下のエンドポイントの例は、"ModbusTCPIP" チャンネル名と "ModbusDevice" デバイス名を持つ "demo-project.json" プロジェクト構成を参照しています。

### デバイス ID の例

エンドポイント (PUT):

```
https://<ホスト名または IP>:<ポート>/config/v1/project/channels/ModbusTCPIP/devices/ModbusDevice
```

ボディ:

```
{ "project_id": <GET で取得したプロジェクト ID>, "servermain.DEVICE_ID_STRING": "<IP アドレス>" }
```

## Modbus タグの作成

エンドポイント (POST):

```
https://<ホスト名または IP>:<ポート>/config/v1/project/channels/MyChannel/devices/MyDevice/tags
```

ボディ:

```
[ { "common.ALLTYPES_NAME": "MyTag1", "servermain.TAG_ADDRESS": "40001" } { "common.ALLTYPES_NAME": "MyTag2", "servermain.TAG_ADDRESS": "40002" } ]
```

関連項目: タグのプロパティのリストについては、付録 C を参照してください。

構成 API を使用したプロジェクト設定の詳細については、サーバーのヘルプを参照してください。

## 自動タグデータベース生成

このドライバーは自動タグデータベース生成をサポートしているため、ドライバーは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を自動的に作成できます。構成に応じて、タグ生成はサーバープロジェクトが開始し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表示されます。

● 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。

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 や ProWORX などの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。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。

● 関連項目: [カスタムアプリケーションからのインポート](#)

● 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。

## カスタムアプリケーションからのインポート

次の CSV ファイルフォーマットを使用してカスタムタグをインポートできます。

[レコードタイプ]; [変数名]; [データ型]; [アドレス]; [設定値]; [コメント]

- **レコードタイプ:** これはタグをインポートするもう 1 つの手段である Concept ソフトウェアで使用されているフラグです。N または E を指定できます。どちらのフラグも同様に処理されます。
- **変数名:** これはサーバー内の静的タグの名前です。長さは最大 256 文字です。
- **データ型:** これはタグのデータ型です。サポートされるデータ型は次のとおりです。
  - BOOL
  - DINT
  - INT
  - REAL (32 ビット Float)
  - UDINT
  - UINT
  - WORD
  - BYTE
  - TIME (DWord として処理)
  - STRING

- **アドレス:** これはタグの Modbus アドレスです。長さは最大 16 文字です。
- **設定値:** これは無視され、空白のままにする必要があります。
- **コメント:** これはサーバー内のタグの説明です。長さは最大 255 文字です。

## 例

- N;Amps;WORD;40001;;Current in
- N;Volts;WORD;40003;;Volts in
- N;Temperature;REAL;40068;;Tank temp

## Modbus イーサネット通信の最適化

Modbus イーサネットドライバーは、システム全体のパフォーマンスへの影響を最小限に抑えながら最大のパフォーマンスが得られるように設計されています。このドライバーは高速ですが、このアプリケーションを制御および最適化して最大のパフォーマンスを得るために参考となるいくつかのガイドラインがあります。

このサーバーでは、Modbus イーサネットなどの通信プロトコルのことをチャンネルと呼びます。アプリケーションで定義されている各チャンネルは、サーバーでの個々の実行パスを表します。チャンネルが定義された後、そのチャンネルの下に一連のデバイスを定義する必要があります。これらのデバイスそれぞれが、データの収集元となる単一の Modbus コントローラを表します。このアプローチに従ってアプリケーションを定義することで高いパフォーマンスが得られますが、ドライバーやネットワークがフルに利用されるわけではありません。単一のチャンネルを使用して構成されているアプリケーションの表示例を次に示します。



デバイスそれぞれが単一の Modbus イーサネットチャンネルの下に表示されます。この構成では、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せよ、全体的な更新レートが低下していきます。

Modbus イーサネットドライバーがチャンネルを 1 つだけ定義可能な場合、上に示した例が唯一可能なオプションとなりますが、このドライバーは最大 256 チャンネルまで定義できます。複数のチャンネルを使用して複数の要求をネットワークに同時に発行することで、データ収集のワークロードが分散されます。パフォーマンスを改善するために同じアプリケーションを複数のチャンネルを使用して構成した場合の例を次に示します。



ここではそれぞれのデバイスが各自のチャンネルの下に定義されています。この新しい構成では、各デバイスからのデータ収集タスクごとに 1 つの実行パスが割り当てられます。アプリケーションのデバイスの数が 256 以下である場合、まさにここで示したように最適化できます。

アプリケーションのデバイスの数が 256 を超える場合でもパフォーマンスは改善されます。デバイスの数は 256 以下であるのが理想的ですが、そうでない場合でもアプリケーションは追加のチャンネルから恩恵を受けます。デバイスの負荷を 256 個のチャンネルすべてに分散してもサーバーはデバイスを切り替えますが、単一のチャンネルで処理するデバイスの数ははるかに少なくなります。

## ブロックサイズ

ブロックサイズは、Modbus イーサネットドライバーのパフォーマンスに影響を与えることがあるもう 1 つのパラメータです。ブロックサイズパラメータはデバイスごとに用意されており、デバイスのプロパティのブロックサイズの設定で定義します。ブロックサイズには、デバイスから一度に要求可能なレジスタまたはビットの数を指定します。ブロックサイズを 1 から 120 レジスタまたは 8 から 2000 ビットに設定することでドライバーのパフォーマンスを微調整できます。

### ヒント:

「タイムアウト時にソケットを閉じる」プロパティを有効にすることで、さらなるパフォーマンスゲインを実現できます。タイムアウトとタイミングのプロパティを調整することによっても、さらなるパフォーマンスゲインを実現できます。



● 詳細については、[イーサネットのプロパティ](#)、[通信タイムアウト](#)、[タイミング](#)を参照してください。

## 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設定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。

### [Applicom のアドレス指定](#)

[CEG のアドレス指定](#)[Fluenta のアドレス指定](#)[Instromet のアドレス指定](#)[メールボックスのアドレス指定](#)[Modbus のアドレス指定](#)[Roxar のアドレス指定](#)

## ドライバーのシステムタグのアドレス指定

### 内部タグ

タグ	説明	データ型	アクセス
ポート	ポートのシステムタグにより、クライアントアプリケーションはポート番号の設定を読み書きできます。このタグへの書き込みが行われると、ドライバーはデバイスから切断し、指定されたポートへの再接続を試みます。	Word、Short、DWord、Long	読み取り/書き込み

#### ● 注記:

- デバイスのポート設定はドライバーによる非送信請求通信には使用されません。
- このドライバーでは、「スレーブ」と「非送信請求」という用語は同義で用いられています。
- このタグに対する変更によってプロジェクトが修正され、サーバーはシャットダウン時にプロジェクトの保存を確認するプロンプトを表示します。

### システムタグ

タグ	説明	データ型	アクセス
_CEGExtension	このタグは CEG モデルのデバイスのみで使用されます。このタグを使用することで、 <b>CEG 拡張</b> のデバイスのプロパティをクライアントアプリケーションから変更できます。	Boolean	読み取り/書き込み
_InputCoilBlockSize	このタグを使用することで、「入力コイル」ブロックサイズのプロパティをクライアントアプリケーションから変更できます。	DWord	読み取り/書き込み
_OutputCoilBlockSize	このタグを使用することで、「出力コイル」ブロックサイズのプロパティをクライアントアプリケーションから変更できます。	DWord	読み取り/書き込み
_InternalRegisterBlockSize	このタグを使用することで、「内部レジスタ」ブロックサイズのプロパティをクライアントアプリケーションから変更できます。	DWord	読み取り/書き込み
_HoldingRegisterBlockSize	このタグを使用することで、「保持レジスタ」ブロックサイズのプロパティをクライアントアプリケーションから変更できます。	DWord	読み取り/書き込み

● 注記: これらのタグに対する変更によってプロジェクトが修正され、サーバーはシャットダウン時にプロジェクトの保存を確認するプロンプトを表示します。

● 関連項目: [イーサネット](#)

### ファンクションコードの説明

次の表に示すファンクションコードは Modbus モデルと Applicom モデルのデバイスによってサポートされています。

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

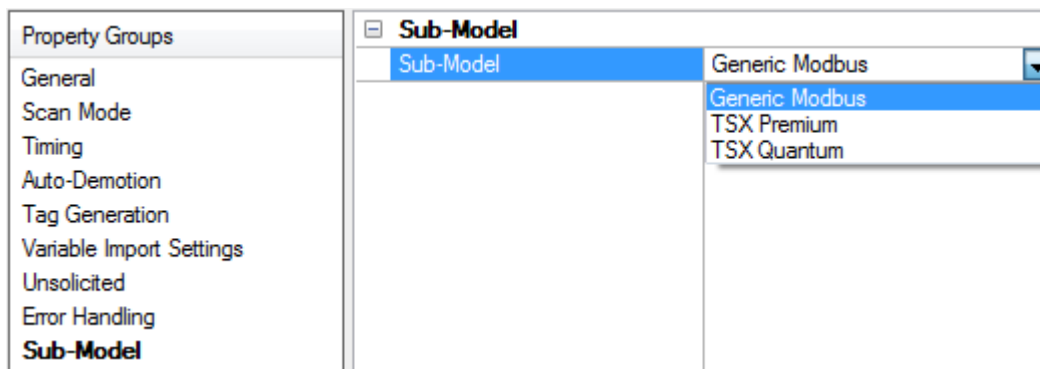
## Applicom のサブモデルとアドレス指定

Applicom デバイスは 3 つの Applicom サブモデルをサポートしています。接続するデバイスに適したサブモデルを選択します。アドレス情報については、以下のサブモデルのリンクをクリックしてください。

[ジェネリック Modbus](#)

[TSX Premium](#)

[TSX Quantum](#)



## ジェネリック Modbus のアドレス指定

すべてのファンクションコードが 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。

### 出力コイル

アドレス	範囲	データ型	アクセス	ファンクションコード
Bxxxxx	0-65535	Boolean	読み取り/書き込み	01, 05, 15

#### 配列のサポート

出力コイルのアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。

Bxxxxx\_列数 (行数は 1 であるものと見なされます)。

Bxxxxx\_行数\_列数。

ベースアドレス + (行数 \* 列数) が 65535 を超えてはなりません。要求されるコイルの総数が、このデバイスに指定された出力コイルのブロックサイズを超えてはなりません。

### 入力コイル

アドレス	範囲	データ型	アクセス	ファンクションコード
Blxxxxx	0-65535	Boolean	読み取り専用	02

#### 配列のサポート

入力コイルのアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。

Blxxxxx\_列数 (行数は 1 であるものと見なされます)。

Blxxxx\_行数\_列数。

ベースアドレス + (行数 \* 列数) が 65535 を超えてはなりません。要求されるコイルの総数が、このデバイスに指定された入力コイルのブロックサイズを超えてはなりません。

## 内部レジスタ

デフォルトのデータ型を太字で示しています。

内部レジスタの位置では Boolean と String 以外のすべてのデータ型で配列がサポートされています。

● **注記:** スレーブデバイスの場合、読み取り専用の位置は読み取り/書き込みが可能です。

アドレス	範囲	データ型	アクセス	ファンクションコード
Wlxxxx	0-65535 0-65534 0-65532	<b>Word</b> , Short, BCD Float, DWord, Long, LBCD Double	読み取り専用	04
Wlxxxx.bb	xxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り専用	04
Wlxxxx:Xbb	xxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り専用	04
Dlxxxx	0-65534	<b>DWord</b>	読み取り専用	04
Flxxxx	0-65534	<b>Float</b>	読み取り専用	04
Wlxxxx_S	0-65535	<b>Short</b>	読み取り専用	04
Wlxxxx_B	0-65535	<b>BCD</b>	読み取り専用	04
Wlxxxx_A**	0-65535	<b>String</b>	読み取り専用	04
Wlxxxx_X<1, 2, 3>***	0-65535 0-65534	<b>Word</b> , Short, BCD Float, DWord, Long, LBCD	読み取り専用	04
Dlxxxx_S	0-65534	<b>Long</b>	読み取り専用	04
Dlxxxx_B	0-65534	<b>LBCD</b>	読み取り専用	04
Dlxxxx_X<1, 2, 3>***	0-65534	<b>DWord</b>	読み取り専用	04
Flxxxx_X<1, 2, 3>***	0-65534	<b>Float</b>	読み取り専用	04
M_Wlxxxx_n(H) (String)、HiLo バイトオーダー (H はオプション)	xxxx=0-65535 n は文字列長 範囲は 1 から 120 Word	<b>String</b>	読み取り専用	04
M_Wlxxxx_nL (String)、LoHi バイトオーダー	xxxx=0-65535 n は文字列長 範囲は 1 から 120 Word	<b>String</b>	読み取り専用	04

\*詳細については、[設定](#)の「ゼロベースのビットアドレス指定」を参照してください。

\*\*文字列の長さは 2 バイトです。

\*\*\*詳細については、[バイト交換のサフィックス](#)を参照してください。

## 配列のサポート

内部レジスタアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。  
 Wxxxxx\_列数 (行数は1であるものと見なされます)。  
 Wxxxxx\_行数\_列数。

Word、Short、BCD 配列の場合、ベースアドレス + (行数 \* 列数) が65535 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65534 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内部レジスタのブロックサイズを超えてはなりません。

## 保持レジスタ

デフォルトのデータ型を太字で示しています。

保持レジスタの位置では Boolean と String 以外のすべてのデータ型で配列がサポートされています。

● **注記:** スレーブデバイスの場合、読み取り専用の位置は読み取り書き込みが可能です。

アドレス	範囲	データ型	アクセス	ファンクションコード
Wxxxxx	0-65535 0-65534 0-65532	<b>Word</b> 、Short、BCD Float、DWord、 Long、LBCD Double	読み取り/書き込み	03, 06, 16
Wxxxxx.bb	xxxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り/書き込み	03, 06, 16, 22
Wxxxxx:Xbb	xxxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り/書き込み	03, 06, 16, 22
Dxxxxx	0-65534	<b>DWord</b>	読み取り/書き込み	03, 06, 16
Fxxxxx	0-65534	<b>Float</b>	読み取り/書き込み	03, 06, 16
Wxxxxx_S	0-65535	<b>Short</b>	読み取り/書き込み	03, 06, 16
Wxxxxx_B	0-65535	<b>BCD</b>	読み取り/書き込み	03, 06, 16
Wxxxxx_A**	0-65535	<b>String</b>	読み取り専用	03, 16
Wxxxxx_X<1, 2, 3>***	0-65535 0-65534	<b>Word</b> 、Short、BCD Float、DWord、 Long、LBCD	読み取り/書き込み	03, 06, 16
Dxxxxx_S	0-65534	<b>Long</b>	読み取り/書き込み	03, 06, 16
Dxxxxx_B	0-65534	<b>LBCD</b>	読み取り/書き込み	03, 06, 16
Dxxxxx_X<1, 2, 3>***	0-65534	<b>DWord</b>	読み取り/書き込み	03, 06, 16
Fxxxxx_X<1, 2, 3>***	0-65534	<b>Float</b>	読み取り/書き込み	03, 06, 16
M_Wxxxxx_n(H) (String)、HiLo バイトオーダー (H はオプション)	xxxxx=0-65535 n は文字列長 範囲は 1 から 120 Word	<b>String</b>	読み取り/書き込み	03, 16
M_Wxxxxx_nL (String)、LoHi バイトオーダー	xxxxx=0-65535 n は文字列長 範囲は 1 から 120 Word	<b>String</b>	読み取り/書き込み	03, 16

\*詳細については、[設定](#)の「ゼロベースのビットアドレス指定」を参照してください。

\*\*文字列の長さは2バイトです。

\*\*\*詳細については、[バイト交換のサフィックス](#)を参照してください。

#### 配列のサポート

保持レジスタアドレスでは配列がサポートされています。10進アドレス指定を使用して配列を宣言する構文を次に示します。

Wxxxxx\_列数 (行数は1であるものと見なされます)。

Wxxxxx\_行数\_列数。

Word、Short、BCD 配列の場合、ベースアドレス + (行数 \* 列数) が 65535 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 65534 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。

#### 文字列のサポート

Applicom 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。文字列の長さは1から120 Word の範囲です。文字列タグに対してブロック読み取りを実行する方法については、[ブロックサイズ](#)を参照してください。

● **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。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、そのデバイスはこの文字列長をサポートしていません。これを解決するには、サポートされている長さまで文字列を短くしてください。

#### バイト交換のサフィックス

これらのサフィックスは16ビット Word、32ビット DWord、または32ビット Float のデータを構成するバイトの交換に使用されます。バイト交換は、デバイスレベルの設定である「Modbus バイトオーダー」と「最初の Word を下位とする」が適用された後で適用されます。詳細については、[設定](#)を参照してください。

バイト交換のサフィックスは内部レジスタおよび保持レジスタでのみ使用できます。アイテムのサフィックスとデータ型に依存する各種交換については、次の表を参照してください。

サフィックス	16ビットデータ型 (Word、Short、BCD)	32ビットデータ型 (DWord、Long、LBCD、Float)
_X1	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 04 03 02 01 (バイト交換)
_X2	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 03 04 01 02 (Word 交換)
_X3	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 02 01 04 03 (Word 内のバイト交換)

### TSX Quantum

すべてのファンクションコードが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。

#### 出カコイル

アドレス	範囲	データ型	アクセス	ファンクションコード
0xxxxx	1-65536	Boolean	読み取り/書き込み	01, 05, 15

#### 配列のサポート

出カコイルのアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。

0xxxxx\_列数 (行数は1であるものと見なされます)。

0xxxxx\_行数\_列数。

ベースアドレス + (行数 \* 列数) が 65536 を超えてはなりません。要求されるコイルの総数が、このデバイスに指定された出カコイルのブロックサイズを超えてはなりません。

#### 入カコイル

アドレス	範囲	データ型	アクセス	ファンクションコード
1xxxxx	1-65536	Boolean	読み取り専用	02

#### 配列のサポート

入カコイルのアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。

1xxxxx\_列数 (行数は1であるものと見なされます)。

1xxxxx\_行数\_列数。

ベースアドレス + (行数 \* 列数) が 65536 を超えてはなりません。要求されるコイルの総数が、このデバイスに指定された入カコイルのブロックサイズを超えてはなりません。

#### 内部レジスタ

デフォルトのデータ型を太字で示しています。

内部レジスタの位置では Boolean と String 以外のすべてのデータ型で配列がサポートされています。

● **注記:** スレーブデバイスの場合、読み取り専用の位置は読み取り書き込みが可能です。

アドレス	範囲	データ型	アクセス	ファンクションコード
3xxxxx	1-65536 1-65535 1-65533	<b>Word</b> 、Short、BCD Float、DWord、Long、 LBCD Double	読み取り専用	04
3xxxxx.bb	xxxxx=1-65536 bb=0/1-15/16*	<b>Boolean</b>	読み取り専用	04
3xxxxx:Xbb	xxxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り専用	04
D3xxxxx	1-65535	<b>DWord</b>	読み取り専用	04
F3xxxxx	1-65535	<b>Float</b>	読み取り専用	04
3xxxxx_S	1-65536	<b>Short</b>	読み取り専用	04
3xxxxx_B	1-65536	<b>BCD</b>	読み取り専用	04
3xxxxx_A**	1-65536	<b>String</b>	読み取り専用	04
3xxxxx_X<1, 2, 3>***	1-65536 1-65535	<b>Word</b> 、Short、BCD Float、DWord、Long、 LBCD	読み取り専用	04
D3xxxxx_S	1-65535	<b>Long</b>	読み取り専用	04
D3xxxxx_B	1-65535	<b>LBCD</b>	読み取り専用	04
D3xxxxx_X<1, 2, 3>***	1-65535	<b>DWord</b>	読み取り専用	04
F3xxxxx_X<1, 2, 3>***	1-65535	<b>Float</b>	読み取り専用	04
M_3xxxxx_n(H) (String)、HiLo バイトオーダー (H はオプション)	xxxxx=1-65536 n は文字列長 範囲は 1 から 120 Word	<b>String</b>	読み取り専用	04
M_3xxxxx_nL (String)、LoHi バイトオーダー	xxxxx=1-65536 n は文字列長 範囲は 1 から 120	<b>String</b>	読み取り専用	04

アドレス	範囲	データ型	アクセス	ファンクションコード
	Word			

\*詳細については、[設定](#)の「ゼロベースのビットアドレス指定」を参照してください。

\*\*文字列の長さは2バイトです。

\*\*\*詳細については、[バイト交換のサフィックス](#)を参照してください。

### 配列のサポート

内部レジスタアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。

3xxxxx\_列数 (行数は1であるものと見なされます)。

3xxxxx\_行数\_列数。

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内部レジスタのブロックサイズを超えてはなりません。

### 保持レジスタ

デフォルトのデータ型を太字で示しています。

保持レジスタの位置では Boolean と String 以外のすべてのデータ型で配列がサポートされています。

● **注記:** スレーブデバイスの場合、読み取り専用の位置は読み取り/書き込みが可能です。

アドレス	範囲	データ型	アクセス	ファンクションコード
4xxxxx	1-65536 1-65535 1-65533	<b>Word</b> 、Short、BCD Float、DWord、 Long、LBCD Double	読み取り/書き込み	03, 06, 16
4xxxxx.bb	xxxxx=1-65536 bb=0/1-15/16*	<b>Boolean</b>	読み取り/書き込み	03, 06, 16, 22
4xxxxx.Xbb	xxxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り/書き込み	03, 06, 16, 22
D4xxxxx	1-65535	<b>DWord</b>	読み取り/書き込み	03, 06, 16
F4xxxxx	1-65535	<b>Float</b>	読み取り/書き込み	03, 06, 16
4xxxxx_S	1-65536	<b>Short</b>	読み取り/書き込み	03, 06, 16
4xxxxx_B	1-65536	<b>BCD</b>	読み取り/書き込み	03, 06, 16
4xxxxx_A**	1-65536	<b>String</b>	読み取り専用	03, 16
4xxxxx_X<1, 2, 3>***	1-65536 1-65535	<b>Word</b> 、Short、BCD Float、DWord、 Long、LBCD	読み取り/書き込み	03, 06, 16
D4xxxxx_S	1-65535	<b>Long</b>	読み取り/書き込み	03, 06, 16
D4xxxxx_B	1-65535	<b>LBCD</b>	読み取り/書き込み	03, 06, 16
D4xxxxx_X<1, 2, 3>***	1-65535	<b>DWord</b>	読み取り/書き込み	03, 06, 16
F4xxxxx_X<1, 2, 3>***	1-65535	<b>Float</b>	読み取り/書き込み	03, 06, 16



アドレス	範囲	データ型	アクセス	ファンクションコード
			き込み	
M_4xxxxx_n(H) (String)、HiLo バイトオーダー (H はオプション)	xxxxx=1-65536 n は文字列長 範囲は 1 から 120 Word	String	読み取り/書き込み	03, 16
M_4xxxxx_nL (String)、LoHi バイトオーダー	xxxxx=1-65536 n は文字列長 範囲は 1 から 120 Word	String	読み取り/書き込み	03, 16

\*詳細については、[設定](#)の「ゼロベースのビットアドレス指定」を参照してください。

\*\*文字列の長さは 2 バイトです。

\*\*\*詳細については、[バイト交換のサフィックス](#)を参照してください。

### 配列のサポート

保持レジスタアドレスでは配列がサポートされています。10 進アドレス指定を使用して配列を宣言する構文を次に示します。

4xxxxx\_列数 (行数は 1 であるものと見なされます)。

4xxxxx\_行数\_列数。

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。

### 文字列のサポート

Applicom 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。文字列の長さは 1 から 120 Word の範囲です。

● 文字列タグに対してブロック読み取りを実行する方法については、[ブロックサイズ](#)を参照してください。

● **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。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、そのデバイスはこの文字列長をサポートしていません。これを解決するには、サポートされている長さまで文字列を短くしてください。

### バイト交換のサフィックス

これらのサフィックスは 16 ビット Word、32 ビット DWord、または 32 ビット Float のデータを構成するバイトの交換に使用されます。バイト交換は、デバイスレベルの設定である「Modbus バイトオーダー」と「最初の Word を下位とする」が適用された後で適用されます。詳細については、[設定](#)を参照してください。

バイト交換のサフィックスは内部レジスタおよび保持レジスタでのみ使用できます。アイテムのサフィックスとデータ型に依存する各種交換については、次の表を参照してください。

サフィックス	16 ビットデータ型 (Word、Short、BCD)	32 ビットデータ型 (DWord、Long、LBCD、Float)
_X1	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 04 03 02 01 (バイト交換)
_X2	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 03 04 01 02 (Word 交換)
_X3	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 02 01 04 03 (Word 内のバイト交換)

## TSX Premium

すべてのファンクションコードが 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。

### 出力コイル

アドレス	範囲	データ型	アクセス	ファンクションコード
%MXxxxxx	0-65535	Boolean	読み取り/書き込み	01, 05, 15
%Mxxxxx	0-65535	Boolean	読み取り/書き込み	01, 05, 15

### 配列のサポート

出力コイルのアドレスでは配列がサポートされています。配列を宣言する構文を次に示します。

%MXxxxxx\_列数 (行数は1であるものと見なされます)。

%MXxxxxx\_行数\_列数。

ベースアドレス + (行数 \* 列数) が 65535 を超えてはなりません。要求されるコイルの総数が、このデバイスに指定された出力コイルのブロックサイズを超えてはなりません。

### 保持レジスタ

デフォルトのデータ型を太字で示しています。

保持レジスタの位置では Boolean と String 以外のすべてのデータ型で配列がサポートされています。

● **注記:** スレーブデバイスの場合、読み取り専用の位置は読み取り/書き込みが可能です。

アドレス	範囲	データ型	アクセス	ファンクションコード
%MWxxxxx	0-65535 0-65534 0-65532	<b>Word</b> , Short, BCD Float, DWord, Long, LBCD Double	読み取り/書き 込み	03, 06, 16
%MWxxxxx.bb	xxxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り/書き 込み	03, 06, 16, 22
%MWxxxxx:Xbb	xxxxx=0-65535 bb=0/1-15/16*	<b>Boolean</b>	読み取り/書き 込み	03, 06, 16, 22
%DWxxxxx または %MDxxxxx	0-65534	<b>DWord</b>	読み取り/書き 込み	03, 06, 16
%FWxxxxx または %MFxxxxx	0-65534	<b>Float</b>	読み取り/書き 込み	03, 06, 16
%MWxxxxx_S	0-65535	<b>Short</b>	読み取り/書き 込み	03, 06, 16
%MWxxxxx_B	0-65535	<b>BCD</b>	読み取り/書き 込み	03, 06, 16
%MWxxxxx_A**	0-65535	<b>String</b>	読み取り専用	03, 16
%MWxxxxx_X<1, 2, 3>***	0-65535 0-65534	<b>Word</b> , Short, BCD Float, DWord, Long, LBCD	読み取り/書き 込み	03, 06, 16
%DWxxxxx_S	0-65534	<b>Long</b>	読み取り/書き 込み	03, 06, 16
%DWxxxxx_B	0-65534	<b>LBCD</b>	読み取り/書き 込み	03, 06, 16
%DWxxxxx_X<1, 2, 3>*** または %MDxxxxx_X<1, 2, 3>***	0-65534	<b>DWord</b>	読み取り/書き 込み	03, 06, 16
%FWxxxxx_X<1, 2, 3>*** または %MFxxxxx_X<1, 2, 3>***	0-65534	<b>Float</b>	読み取り/書き 込み	03, 06, 16
M_%MWxxxxx_n(H) String、HiLo バイトオーダー (H はオプション)	xxxxx=0-65535 n は文字列長 範囲は 1 から 120 Word	<b>String</b>	読み取り/書き 込み	03, 16

アドレス	範囲	データ型	アクセス	ファンクションコード
M_%MWxxxx_nL (String)、LoHi バイトオーダー	xxxx=0-65535 n は文字列長 範囲は 1 から 120 Word	String	読み取り書き込み	03, 16

\*詳細については、[設定](#)の「ゼロベースのビットアドレス指定」を参照してください。

\*\*文字列の長さは 2 バイトです。

\*\*\*詳細については、[バイト交換のサフィックス](#)を参照してください。

#### 配列のサポート

保持レジスタアドレスでは配列がサポートされています。10 進アドレス指定を使用して配列を宣言する構文を次に示します。

%MWxxxx\_列数 (行数は 1 であるものと見なされます)。

%MWxxxx\_行数\_列数。

Word、Short、BCD 配列の場合、ベースアドレス + (行数 \* 列数) が 65535 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 65534 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。

#### 文字列のサポート

Applicom 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。文字列の長さは 1 から 120 Word の範囲です。文字列タグに対してブロック読み取りを実行する方法については、[ブロックサイズ](#)を参照してください。

● **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。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、そのデバイスはこの文字列長をサポートしていません。これを解決するには、サポートされている長さまで文字列を短くしてください。

#### バイト交換のサフィックス

これらのサフィックスは 16 ビット Word、32 ビット DWord、または 32 ビット Float のデータを構成するバイトの交換に使用されます。バイト交換は、デバイスレベルの設定である「Modbus バイトオーダー」と「最初の Word を下位とする」が適用された後で適用されます。詳細については、[設定](#)を参照してください。

バイト交換のサフィックスは内部レジスタおよび保持レジスタでのみ使用できます。アイテムのサフィックスとデータ型に依存する各種交換については、次の表を参照してください。

サフィックス	16 ビットデータ型 (Word、Short、BCD)	32 ビットデータ型 (DWord、Long、LBCD、Float)
_X1	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 04 03 02 01 (バイト交換)
_X2	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 03 04 01 02 (Word 交換)
_X3	01 02 -> 02 01 (バイト交換)	01 02 03 04 -> 02 01 04 03 (Word 内のバイト交換)

#### CEG のアドレス指定

CEG デバイスマodelのアドレス指定は Modbus デバイスマodelのアドレス指定と同じです。

● 詳細については、[Modbus のアドレス指定](#)を参照してください。

#### Fluenta のアドレス指定

デフォルトのデータ型を太字で示しています。

アドレス	範囲	データ型	アクセス
システム	400000-409999	<b>Float</b> 、Double	読み取り/書き込み
出力	410000-410999 420000-420999 430000-430999	<b>Float</b> 、Double	読み取り専用
ユーザー	411000-411999 421000-421999 431000-431999	<b>Float</b> 、Double	読み取り/書き込み
サービス	412000-412999 422000-422999 432000-432999	<b>Float</b> 、Double	読み取り/書き込み
累積	413000-413999 423000-423999 433000-433999	<b>Float</b> 、Double	読み取り専用

## Instromet のアドレス指定

デフォルトのデータ型を太字で示しています。

アドレス	範囲	データ型	アクセス
短整数	400000-400199	<b>Word</b> 、Short	読み取り専用
長整数	400200-400399	<b>DWord</b> 、Long	読み取り専用
単精度実数	400400-400599	<b>Float</b>	読み取り専用

## メールボックスのアドレス指定

デフォルトのデータ型を太字で示しています。

### 10 進アドレス指定

アドレス	範囲	データ型	アクセス
4xxxxx	1-65536	Word、Short、BCD	読み取り/書き込み
4xxxxx.bb	xxxxx=1-65536 bb=0-15	<b>Boolean</b>	読み取り/書き込み
4xxxxx	1-65535	Float、DWord、Long、LBCD	読み取り/書き込み

### 16 進アドレス指定

アドレス	範囲	データ型	アクセス
H4yyyyy	1-10000	Word、Short、BCD	読み取り/書き込み
H4yyyyy.c	yyyyy=1-10000 c=0-F	<b>Boolean</b>	読み取り/書き込み
H4yyyyy	1-FFFF	Float、DWord、Long、LBCD	読み取り/書き込み

● **注記:** Modbus メールボックスではファンクションコード 22 (0x16) はサポートされていません。0x10 (複数の保持レジスタへの書き込み) と 0x6 (単一の保持レジスタへの書き込み) はサポートされています。デバイスのプロパティの「設定」タブにある「**保持レジスタのビットマスク**」をオフにすることで、個々のビットに書き込むことが可能です。これにより、ビットに直接書き込む代わりに、読み取り/修正/書き込みシーケンスが使用されます。これが機能するためには、(メールボックスではなく) マスター Modbus デバイスの設定のみを変更する必要があります。

## 配列

保持レジスタアドレスでは配列もサポートされています。(10 進アドレス指定を使用して) 配列を宣言する構文を次に示します。

4xxxx[列数] (行数は 1 であるものと見なされます)。

4xxxx[行数][列数]。

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。

## Modbus のアドレス指定

このドライバーでは、「スレーブ」と「非送信請求」という用語は同義で用いられています。

### 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。データアイテムの最大値は 2 バイトの符号なし整数 (65,535) です。内部では、このドライバーはアドレステーブルとアイテム全体を表すのに 6 桁を必要とします。Modbus デバイスの多くはすべてのデータアイテムをサポートしていないことに注意してください。そのようなデバイスのアドレスを入力する際の混乱を回避するため、このドライバーはアドレスフィールドに入力されたものに従ってアドレスに"パディング" (桁を追加) します。プライマリテーブルタイプの後ろに最大 4 桁ある場合 (例: 4x、4xx、4xxx、4xxxx)、アドレスはそのままになるか、5 桁までゼロが追加されます。プライマリテーブルタイプの後ろに 5 桁ある場合 (例: 4xxxxx)、アドレスは変わりません。内部では、41、401、4001、40001、または 400001 として入力されたアドレスはすべて、プライマリテーブルタイプ 4 とデータアイテム 1 を示すアドレスを表します。

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)

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	<b>Word</b> 、Short、BCD Float、DWord、Long、LBCD Double	読み取り専用	04
	300001-365535		読み取り専用	04
	300001-365533		読み取り専用	04
	xxxxx=1-65536 bb=0/1-15/16**	<b>Boolean</b>	読み取り専用	04
	300001.2H-365536.240H*** 300001.2L-365536.240L***	String String	読み取り専用 読み取り専用	04 04
保持レジスタ	400001-465536 400001-465535 400001-465533	<b>Word</b> 、Short、BCD Float、DWord、Long、LBCD Double	読み取り/書き込み	03, 06, 16
	xxxxx=1-65536		<b>Boolean</b>	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*	ファンクションコード
	bb=0/1-15/16*		き込み	
	400001.2H-465536.240H***	String	読み取り/書き込み	03, 16
	400001.2L-465536.240L***	String	読み取り/書き込み	03, 16
			読み取り/書き込み	

\*スレーブデバイスの場合、読み取り専用の位置は読み取り/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	範囲	データ型	アクセス*
出カコイル	H000001-H010000	Boolean	読み取り/書き込み
入カコイル	H100001-H110000	Boolean	読み取り専用
内部レジスタ	H300001-H310000	Word、Short、BCD	読み取り専用
	H300001-H30FFFF	Float、DWord、Long、LBCD	読み取り専用
	H300001-H30FFFD	Double	読み取り専用
	yyyyy=1-10000 cc=0/1-F/10	Boolean	読み取り専用
	H300001.2H-H3FFFF.240H**	String	読み取り専用
	H300001.2L-H3FFFF.240L**	String	読み取り専用
保持レジスタ	H400001-H410000	Word、Short、BCD	読み取り/書き込み
	H400001-H40FFFF	Float、DWord、Long、LBCD	読み取り/書き込み
	H400001-H40FFFD	Double	読み取り/書き込み
	yyyyy=1-10000 cc=0/1-F/10	Boolean	読み取り/書き込み
	H400001.2H-H4FFFF.240H	String	読み取り/書き込み
	H400001.2L-H4FFFF.240L	String	読み取り/書き込み
			読み取り/書き込み

\*スレーブデバイスの場合、読み取り専用の位置は読み取り/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。この機能は入カコイルと出カコイルの両方で、ポーリングモードでのみ使用できます。非送信請求メモリマップにアクセスするよう設定されているデバイスやメールボックスモードのデバイスでは使用できません。10進の構文は0xxxx#nnであり、ここで:

\*スレーブデバイスの場合、読み取り専用の位置は読み取り/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。

### バックコイル

バ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。この機能は入力コイルと出力コイルの両方で、ポーリングモードでのみ使用できます。非送信請求メモリマップにアクセスするように設定されているデバイスやメールボックスモードのデバイスでは使用できません。10進の構文は0xxxx#nnであり、ここで:

- xxxxx は1つ目のコイルのアドレスです (範囲は000001-065521)。
- nn はアナログ値にパックされるコイルの数です (範囲は01-16)。

16進の構文はH0yyyyy#nnであり、ここで:

- yyyyy は1つ目のコイルのアドレスです (範囲はH000001-H000FFF1)。
- nn はアナログ値にパックされるコイルの数です (範囲は01-16)。

#### ● 注記:

1. 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。10進アドレス指定では、出力コイルはファンクションコード 01 と 15 をサポートするのに対し、入力コイルはファンクションコード 02 をサポートします。
2. 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変更した場合、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。

● **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。

● **注記:** Modbus モデルの文字列タグに対してブロック読み取りを実行する方法については、[ブロックサイズ](#)を参照してください。

#### 例

1. 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。
2. 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。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、文字列を短くしてみてください。

### 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。

4xxxx [行数] [列数]

4xxxx [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。

## Roxar のアドレス指定

デフォルトのデータ型を太字で示しています。

アドレス	範囲	データ型	アクセス
短整数	403000-403999	<b>Word</b> 、Short	読み取り/書き込み
単精度実数	407000-407999	<b>Float</b>	読み取り/書き込み
単精度実数	409000-409999	<b>Float</b>	読み取り専用



## 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ーフェースの「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と並べ替えについては、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。

---

### Winsock 通信を開始できませんでした。

**エラータイプ:**

エラー

---

### 非送信請求通信を開始できませんでした。

**エラータイプ:**

エラー

**考えられる原因:**

ドライバーは非送信請求通信用のリッスンソケットを作成できませんでした。

**解決策:**

チャンネルレベルで定義されているポートがシステム上の別のアプリケーションによって使用されていないことを確認してください。

**● 注記:**

このドライバーでは、「スレーブ」と「非送信請求」という用語は同義で用いられています。

---

### 未定義のデバイスに対して非送信請求メールボックスアクセスが行われました。ソケットを閉じています。| IP アドレス = '<アドレス>'。

**エラータイプ:**

エラー

**考えられる原因:**

1. 指定されている IP アドレスを持つデバイスがサーバーに対してメールボックスメッセージを送信しようとした。その IP を持つデバイスがメールボックスプロジェクトで設定されていないため、メッセージは検証を通過しませんでした。
2. 指定されている IP アドレスを持つデバイスがサーバーに対してメールボックスメッセージを送信しようとした。デバイスは設定されていますが、そこからデータを要求しているクライアントがないため、メッセージは検証を通過しませんでした。

**解決策:**

サーバーがメールボックスメッセージを受け入れるためには、指定されているデバイス IP がプロジェクトで設定されている必要があります。デバイスの少なくとも 1 つのデータアイテムがクライアントによって要求されている必要があります。

---

### 受信した要求は非送信請求メールボックスでサポートされていません。| IP アドレス = '<アドレス>'。

**エラータイプ:**

エラー

**考えられる原因:**

指定されているデバイス IP からサポートされていない要求を受信しました。要求のフォーマットは無効であり、Modbus の仕様の範囲内にありません。

**解決策:**

メールボックスデータの送信用に設定されているデバイスが有効な要求を送信していることを確認してください。

---

### 非送信請求メールボックスのメモリ割り当てエラー。| IP アドレス = '<アドレス>'。

#### エラータイプ:

エラー

#### 考えられる原因:

1. 指定されている IP アドレスを持つデバイスがサーバーに対してメールボックスメッセージを送信しようとしました。その IP を持つデバイスがメールボックスプロジェクトで設定されていないため、メッセージは検証を通過しませんでした。
2. 指定されている IP アドレスを持つデバイスがサーバーに対してメールボックスメッセージを送信しようとしました。デバイスは設定されていますが、そこからデータを要求しているクライアントがないため、メッセージは検証を通過しませんでした。

#### 解決策:

サーバーがメールボックスメッセージを受け入れるためには、指定されているデバイス IP がプロジェクトで設定されている必要があります。デバイスの少なくとも 1 つのデータアイテムがクライアントによって要求されている必要があります。

---

### ソケット接続を作成できません。

#### エラータイプ:

エラー

#### 考えられる原因:

サーバーは指定されたデバイスとの TCP/IP ソケット接続を確立できませんでしたが、引き続き接続を試みます。

#### 解決策:

1. デバイスがオンラインであることを確認してください。
2. デバイス IP が、サーバーがバインドされている IP のサブネット内であることを確認してください。ほかのネットワークへの接続が可能な有効なゲートウェイが使用可能であることを確認してください。

---

### 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

---

### 不良配列。| 配列範囲 = <開始> ~ <end>。

#### エラータイプ:

エラー

#### 考えられる原因:

アドレス空間の末端を超えてアドレスの配列が定義されています。

#### 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。

---

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

#### エラータイプ:

エラー

#### 考えられる原因:

ドライバーは、おそらく範囲外である、PLC 内の存在しない位置を読み取ろうとしました。たとえば、保持レジスタ 40001 ~ 41400 を持つ PLC でアドレス 41405 を要求した場合にこのエラーが生成されます。このエラーが生成された場合、ドライバーは指定されたデータブロックをその PLC から再び要求しません。この同じブロックから要求されているその他のアドレスはすべて無効と見なされます。

**解決策:**

デバイスの範囲内にあるアドレスを要求するようクライアントアプリケーションを更新してください。

**● 関連項目:**

エラー処理

---

**ホストの解決に失敗しました。| ホスト名 = '<名前>'。****エラータイプ:**

エラー

**考えられる原因:**

このデバイスは IP アドレスではなく DNS ホスト名を使用するよう設定されています。このホスト名をサーバーによって IP アドレスに解決することはできません。

**解決策:**

デバイスがオンラインでありドメインに登録されていることを確認してください。

---

**指定された出力コイルブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (コイル)、最大ブロックサイズ = <数値> (コイル)。****エラータイプ:**

エラー

---

**指定された入力コイルブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (コイル)、最大ブロックサイズ = <数値> (コイル)。****エラータイプ:**

エラー

---

**指定された内部レジスタブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (レジスタ)、最大ブロックサイズ = <数値> (レジスタ)。****エラータイプ:**

エラー

---

**指定された保持レジスタブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (レジスタ)、最大ブロックサイズ = <数値> (レジスタ)。****エラータイプ:**

エラー

---

**ブロック要求で例外が返されました。| ブロック範囲 = <アドレス> ~ <address>、例外 = <コード>。****エラータイプ:**

警告

**考えられる原因:**

デバイスが例外コードを返しました。

**解決策:**

例外コードのドキュメントを参照してください。

**● 関連項目:**

Modbus 例外コード

---

**ブロック要求で例外が返されました。| ブロック範囲 = <アドレス> ~ <address>、関数コード = <コード>、例外 = <コード>。****エラータイプ:**

警告

**考えられる原因:**

デバイスが例外コードを返しました。

**解決策:**

例外コードのドキュメントを参照してください。

**● 関連項目:**

Modbus 例外コード

---

**受信したブロックの長さは不適切です。| ブロック範囲 = <開始> ~ <end>。****エラータイプ:**

警告

**考えられる原因:**

ドライバーは PLC 内のメモリブロックを読み取ろうとしました。PLC はエラーなしで応答しましたが、要求されたブロックサイズのデータをドライバーに返しませんでした。

**解決策:**

その範囲のメモリが PLC に存在することを確認してください。

---

**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でした。

**解決策:**

不要なアプリケーションをすべて終了してから、再試行してください。

---

**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 = <field>。****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が無効なため、解析できませんでした。

**解決策:**

可能な場合、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、サーバーに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

---

**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:**

デバイスが例外コードを返しました。

**解決策:**

例外コードのドキュメントを参照してください。

**● 関連項目:**

Modbus 例外コード

---

**イーサネット マネージャが開始されました。****エラータイプ:**

情報

---

**イーサネット マネージャが停止しました。****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

---

**クライアントアプリケーションはシステムタグ \_CEGExtension を介して CEG 拡張を変更しました。| 拡張 = '<拡張>'。****エラータイプ:**

情報

**考えられる原因:**

サーバーに接続しているクライアントアプリケーションは指定されたデバイスでの CEG 拡張を 0 (Modbus) または 1 (CEG) に変更しました。

**解決策:**

このデバイスプロパティは CEG モデルのデバイスのみ適用されます。変更はその他のモデルには影響しません。クライアントアプリケーションがこのプロパティを変更できないようにするには、OPC DA 設定を介してシステムレベルのタグに対するクライアントの書き込み権限を無効にします。

---

**非送信請求通信を開始しています。| プロトコル = '<名前>'、ポート = <数値>'。****エラータイプ:**

情報

---

**スレーブデバイス用のメモリが作成されました。| スレーブデバイス ID = <デバイス>'。****エラータイプ:**

情報

---

**すべてのチャンネルが仮想ネットワークを利用しているため、非送信請求通信を停止しています。****エラータイプ:**

情報

---

**チャンネルは仮想ネットワーク内にあるため、すべてのデバイスがデバイスにつき 1 つのソケットを使用する設定に戻りました。****エラータイプ:**

情報

---

**接続されているクライアントでデバイス ID を 'マスター' から 'スレーブ' に変更できません。****エラータイプ:**

情報

---

**接続されているクライアントでデバイス ID を 'スレーブ' から 'マスター' に変更できません。**

エラータイプ:

情報

チャンネルが仮想ネットワーク内にある場合、スレーブモードは許可されません。デバイス ID にループバックアドレスまたはローカル IP アドレスが含まれていてはなりません。

---

エラータイプ:

情報

チャンネルが仮想ネットワーク内にある場合、メールボックスモデルは許可されません。

---

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は、サーバー (またはスレーブ) に対して実行できない操作です。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。サーバー (またはスレーブ) はレジスタ値を返すよう設定されていないがこれを要求されているなど、このタイプの要求を処理するには不適切な状態にある可能性もあります。
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は、サーバー (またはスレーブ) に対して使用できないアドレスです。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、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は、サーバー (またはスレーブ) に使用できない値です。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ため、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	SLAVE DEVICE FAILURE (スレーブデバイスエラー)	サーバー (またはスレーブ) が要求された操作を実行しようとしているときに回復不可能なエラーが発生しました。
05/0x05	ACKNOWLEDGE	スレーブは要求を受け入れて処理していますが、これには長い時間が必要です。マスターでタイムアウトエラーが発生しないようにするため、この応答が返されます。マスターは次にプログラム完了ポーリングメッセージを送信することで処理が完了したかどうかを判別します。
06/0x06	SLAVE DEVICE BUSY (スレーブデバイスビジー)	スレーブは長い時間がかかるプログラムコマンドを処理しています。マスターはスレーブが処理を終えた後でメッセージを再送信する必要があります。
07/0x07	NEGATIVE ACKNOWLEDGE (否定応答)	スレーブは照会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。マスターはスレーブから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)	スレーブは拡張メモリを読み取ろうとしましたが、メモリ内でパリティエラーを検出しました。マスターは要求を再試行できますが、スレーブデバイス上でサービスが必要な場合があります。
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。

● このドライバーでは、「スレーブ」と「非送信請求」という用語は同義で用いられています。



# 索引

## 1

10 進アドレス指定 36

16 進アドレス指定 36

## 5

5 桁のアドレス指定 37

## 6

6 桁のアドレス指定 37

## A

Appicom のアドレス指定 27

## B

BCD 25

BOOL 23

Boolean 25

BYTE 23

## C

CEG のアドレス指定 35

CEG 拡張 17

CEGExtension 26

CSV 23

## D

DINT 23

Double 25

DWord 25

**F**

Float 25  
Fluenta 5  
Fluenta のアドレス指定 35

**H**

HoldingRegisterBlockSize 26

**I**

ID 7  
InputCoilBlockSize 26  
Instromet 5  
Instromet のアドレス指定 36  
INT 23  
InternalRegisterBlockSize 26  
IP プロトコル 15

**L**

LBCD 25  
Long 25  
Long を 10 進数として扱う 18

**M**

Modbus イーサネット通信の最適化 24  
Modbus のアドレス指定 37  
Modbus バイトオーダー 17  
Modbus マスター 6  
Modbus マスターと Modbus 非送信請求の考慮事項 14  
Modbus メールボックス 36  
Modbus 関数 05 17  
Modbus 関数 06 16  
Modbus 非送信請求 6  
Modbus 例外コード 48  
Modicon ビットオーダー 17

## O

OPC 品質を不良に設定 13

OutputCoilBlockSize 26

## R

REAL 23

Roxar 6

Roxar のアドレス指定 40

## S

Short 25

String 25

STRING 23

## T

TIME 23

TSX Premium 33

TSX Quantum 30

## U

UDINT 23

UINT 23

## W

Winsock 通信を開始できませんでした。 41

Word 25

WORD 23

## あ

アドレス 24

アドレスに書き込みません。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>。 45

アドレスの説明 25

## い

イーサネット 15

イーサネットから Modbus Plus へのブリッジ 5

イーサネットマネージャが開始されました。 46

イーサネットマネージャが停止しました。 46

イベントログメッセージ 41

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>'。 45

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 = <field>。 44

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>。 45

## え

エラー時に格下げ 10

エラー処理 14

## か

カスタムアプリケーションからのインポート 23

カスタムタグ 23

## き

キャッシュからの初回更新 9

## く

クライアントアプリケーションはシステムタグ \_CEGExtension を介して CEG 拡張を変更しました。| 拡張 = '<拡張>'。  
46

## こ

コイルのステータスを読み取り 27

コメント 24

## さ

サービス 36

サブグループを許可 11

## し

ジェネリック Modbus のアドレス指定 27

システム 36

システムタグ 26

シミュレーション 8

## す

スキャンしない、要求ポールのみ 9

スキャンモード 8

すべてのチャンネルが仮想ネットワークを利用しているため、非送信請求通信を停止しています。 46

スレーブデバイス用のメモリが作成されました。|スレーブデバイス ID = <デバイス>。 46

## せ

ゼロベースアドレス指定 16

ゼロベースのビットアドレス指定 16

## そ

ソケット接続を作成できません。 42

## た

タイムアウト時にソケットを閉じる 15

タイムアウト前の試行回数 9

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>'。 42

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>'。 46

タグに指定のスキャン速度を適用 9

タグのインポート中にファイル例外が発生しました。 44

タグ生成 10

## ち

チャンネルが仮想ネットワーク内にある場合、スレーブモードは許可されません。デバイス ID にループバックアドレスまたはローカル IP アドレスが含まれてはなりません。 47

チャンネルが仮想ネットワーク内にある場合、メールボックスモデルは許可されません。 47

チャンネルは仮想ネットワーク内にあるため、すべてのデバイスがデバイスにつき 1 つのソケットを使用する設定に戻りました。 46

チャンネル割り当て 7

## て

データアクセス 16

データエンコーディング 17

データコレクション 8

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>'。 45

データ型の説明 25

デバイスのプロパティ-タグ生成 10

デバイスの設定 6

デバイス起動時 11

## と

ドライバー 7

ドライバーのシステムタグのアドレス指定 26

## は

バイト交換のサフィックス 30, 35

## ふ

ファンクションコードの説明 26

ブロックサイズ 18

ブロックに不良アドレスがあります。| ブロック範囲 = <アドレス> ~ <address>。 42

ブロック要求で例外が返されました。| ブロック範囲 = <アドレス> ~ <address>、関数コード = <コード>、例外 = <コード>。 44

ブロック要求で例外が返されました。| ブロック範囲 = <アドレス> ~ <address>、例外 = <コード>。 43

プロパティ変更時 11

## へ

ヘルプの目次 5

## ほ

ポート 15, 26

ホストの解決に失敗しました。| ホスト名 = '<名前>'。 43

## め

- メールボックス 6
- メールボックスクライアントの権限 17
- メールボックスのアドレス指定 36
- メールボックスモード 39
- 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。 44

## も

- モデル 5, 7

## ゆ

- ユーザー 36

## れ

- レコード 23
- レジスタへのマスク書き込み 27

## 漢字

- 一般 7
- 概要 5
- 格下げまでのタイムアウト回数 10
- 格下げ期間 10
- 格下げ時に要求を破棄 10
- 最初の DWord を下位とする 17
- 最初の Word を下位とする 17
- 作成 12
- 削除 11
- 指定された出力コイルブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (コイル)、最大ブロックサイズ = <数値> (コイル)。 43
- 指定された内部レジスタブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (レジスタ)、最大ブロックサイズ = <数値> (レジスタ)。 43
- 指定された入力コイルブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (コイル)、最大ブロックサイズ = <数値> (コイル)。 43
- 指定された保持レジスタブロックサイズは最大ブロックサイズを超えています。| 指定されたブロックサイズ = <数値> (レジスタ)、最大ブロックサイズ = <数値> (レジスタ)。 43
- 自動タグデータベース生成 23
- 自動格下げ 10
- 受信したブロックの長さは不適切です。| ブロック範囲 = <開始> ~ <end>。 44

受信した要求は非送信請求メールボックスでサポートされていません。| IP アドレス = '<アドレス>'。 41

重複タグ 11

出力 36

出力コイル 19, 27, 30, 33, 37

書き込み専用アクセス 39

上書き 11

冗長 20

親グループ 11

生成 11

接続されているクライアントでデバイス ID を 'スレーブ' から 'マスター' に変更できません。 47

接続されているクライアントでデバイス ID を 'マスター' から 'スレーブ' に変更できません。 46

接続のタイムアウト 9

設定 15

設定値 24

説明 7

説明を含める 12

単一コイルを適用 27

単一レジスタをプリセット 27

単精度実数 36, 40

短整数 36, 40

長整数 36

通信タイムアウト 9, 13

内部タグ 26

内部レジスタ 19, 28, 31, 37

内部レジスタを読み取り 27

入力コイル 19, 27, 30, 37

入力ステータスを読み取り 27

配列 36

配列のサポート 27-28, 30, 34-35, 39

非送信請求 12

非送信請求メールボックスのメモリ割り当てエラー。| IP アドレス = '<アドレス>'。 42

非送信請求通信を開始しています。| プロトコル = '<名前>', ポート = <数値>。 46

非送信請求通信を開始できませんでした。 41

不正なアドレスでタグを無効化 14

不良配列。| 配列範囲 = <開始> ~ <end>。 42

複数コイルを適用 27

複数レジスタをプリセット 27

文字列のサポート 30, 35, 39

文字列のブロック読み取り 19

変数 23

変数のインポート設定 12

保持レジスタ 19, 29, 32, 34, 37

保持レジスタのビットマスク 16

保持レジスタを読み取り 27

未定義のデバイスに対して非送信請求メールボックスアクセスが行われました。ソケットを閉じています。| IP アドレス



= '<アドレス>'. 41

名前 7

要求のタイムアウト 9

要求間遅延 9

累積 36