

# Modbus RTU Server Serialドライバー

© 2024 PTC Inc. All Rights Reserved.

# 目次

Modbus RTU Server Serial ドライバー .....	1
目次 .....	2
Modbus RTU Server Serial ドライバー .....	3
概要 .....	3
設定 .....	3
チャンネルのプロパティ - 一般 .....	4
タグ数 .....	5
チャンネルのプロパティ - シリアル通信 .....	5
チャンネルのプロパティ - 書き込み最適化 .....	7
チャンネルのプロパティ - 詳細 .....	8
チャンネルのプロパティ - タイミング .....	8
デバイスのプロパティ - 一般 .....	10
デバイスのプロパティ - スキャンモード .....	11
デバイスのプロパティ - メモリ .....	12
メモリのアドレス指定 .....	14
データ型の説明 .....	16
アドレスの説明 .....	17
Modbus のアドレス指定 .....	17
Daniels/Enron のアドレス指定 .....	18
イベントログメッセージ .....	21
アドレスサイズが変更されました。  以前のサイズ = <数値>、現在のサイズ = <数値>。 .....	21
エラーマスクの定義 .....	21
Modbus 例外コード .....	22
索引 .....	23

## Modbus RTU Server Serial ドライバー

---

ヘルプバージョン 1.053

### 目次

#### 概要

Modbus RTU Server Serial ドライバー とは

#### 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

#### データ型の説明

このドライバーでサポートされるデータ型

#### アドレスの説明

サーバーデバイスでデータ位置のアドレスを指定する方法

#### イベントログメッセージ

Modbus RTU Server Serial ドライバー で生成されるメッセージ

## 概要

---

Modbus RTU Server Serial ドライバー は Modbus シリアルデバイスが 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。これは、シリアル通信ネットワーク上で最大 255 台の Modbus サーバーデバイスをシミュレートします。その他のデバイスや PC は、Modbus プロトコルを使用して、シミュレーション対象の各 Modbus サーバーデバイスと通信することができます。

● **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。

## 設定

---

### サポートされるデバイス

Modbus 対応 デバイス

### 通信プロトコル

Modbus RTU プロトコル

シリアル通信/ポート 設定

「**ボーレート**」: 1200、2400、9600、19200

「**パリティ**」: 奇数、偶数、なし

「**データビット**」: 8

「**ストップビット**」: 1、2

「**フロー制御**」: RS232/RS485 コンバータを使用している場合、必要なフロー制御のタイプはコンバータの要件によって異なります。コンバータには、フロー制御を必要としないものと、RTS フローを必要とするものがあります。コンバータのフロー要件についてはコンバータのドキュメントを参照してください。自動フロー制御を備えた RS485 コンバータを使用することをお勧めします。

● **注記:**

1. 製造メーカーから供給されている通信ケーブルを使用している場合、フロー制御の設定として「**RTS**」または「**RTS 常時**」を選択する必要があることがあります。
2. リストされている構成がすべてのデバイスでサポートされるわけではありません。

### サポートされているファンクションコード

- コイルのステータスの読み取り - コード 01H
- 入力ステータスの読み取り - コード 02H
- 保持レジスタの読み取り - コード 03H
- 内部レジスタの読み取り - コード 04H

- 単一コイルの適用 - コード 05H
- 単一レジスタのプリセット - コード 06H
- 診断ループバック - コード 08H
- 複数コイルの適用 - コード 0FH
- 複数レジスタのプリセット - コード 10H

● **注記:** その他のすべてのファンクションコードについて、ドライバーは要求元デバイスに例外コード 01H (ファンクションが実装されていません) を返します。

### ブロードキャストコマンド

Modbus RTU Server Serial ドライバー はブロードキャスト書き込みメッセージを受信できます。ブロードキャストメッセージはステーション ID 0 を使用して定義されています。ステーション ID が 0 である書き込みメッセージ (ファンクション 05H、06H、0FH、または 10H) をドライバーが受信すると、そのコマンドを受信したチャンネル下で定義されているすべてのデバイスに書き込み対象の値が配置されます。基本的に、ドライバーで設定されている各デバイスに同時に 1 ピースのデータを送信するには、ブロードキャストコマンドを使用できます。

● **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。

### チャン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ンネルの最大数は 100 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 つのチャンネルにつき 255 です。このドライバーは、シリアル通信ネットワーク上で最大 255 台の Modbus サーバーデバイスをシミュレートします。

### チャンネルのプロパティ - 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ンネルと呼びます。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ンネルから成ります。チャンネルは、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ンネルプロパティを指定するときに使用します。

プロパティグループ	<input type="checkbox"/> <b>識別</b>	
一般	名前	
イーサネット通信	説明	
書き込み最適化	ドライバー	
詳細	<input type="checkbox"/> <b>診断</b>	
プロトコル設定	診断取り込み	無効化
	<input type="checkbox"/> <b>タグ数</b>	
	静的タグ	1

### 識別

「**名前**」: このチャン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、それぞれのチャン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。チャンネル名は OPC ブラウザ情報の一部です。チャン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「チャンネル、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**」: このチャン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**」: このチャンネル用のプロトコルドライバーを指定します。チャン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。チャン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。チャン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ンネル名を変更することも含まれます。チャン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。チャン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、古いチャン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。

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、チャンネルの診断情報が 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ン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。

## チャンネルのプロパティ - シリアル通信

シリアル通信のプロパティはシリアルドライバーで設定でき、選択されているドライバー、接続タイプ、オプションによって異なります。使用可能なプロパティのスーパーセットを以下に示します。

次のいずれかのセクションをクリックしてください: [接続タイプ](#)、[シリアルポートの設定](#)および[実行動作](#)。

### ● 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。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。
- 使用する特定の通信パラメータを定義する必要があります。ドライバーによっては、チャンネルが同一の通信パラメータを共有できる場合とできない場合があります。仮想ネットワークに設定できる共有シリアル接続は1つだけです ([「チャンネルのプロパティ - シリアル通信」](#)を参照してください)。

プロパティグループ	<input type="checkbox"/> <b>接続タイプ</b>	
一般	物理メディア	COM ポート
<b>シリアル通信</b>	共有	いいえ
書き込み最適化	<input type="checkbox"/> <b>シリアルポートの設定</b>	
詳細	COM ID	3
通信シリアル化	ボーレート	19200
リンク設定	データビット	8
	パリティ	なし
	ストップビット	1
	フロー制御	なし
	<input type="checkbox"/> <b>実行動作</b>	
	通信エラーを報告	有効化

## 接続タイプ

「**物理メディア**」: データ通信に使用するハードウェアデバイスのタイプを選択します。次のオプションがあります: 「モデム」、「COM ポート」、「なし」。デフォルトは「COM ポート」です。

- 「**なし**」: 物理的な接続がないことを示すには「なし」を選択します。これによって[通信なしの動作](#)セクションが表示されます。
- 「**COM ポート**」: [シリアルポートの設定](#)セクションを表示して設定するには、「COM ポート」を選択します。
- 「**モデム**」: 通信に電話回線を使用する場合 ([モデム設定](#)セクションで設定)、「モデム」を選択します。
- 「**共有**」: 現在の構成を別のチャンネルと共有するよう接続が正しく識別されていることを確認します。これは読み取り専用プロパティです。

## シリアルポートの設定

「COM ID」: チャンネルに割り当てられているデバイスと通信するときに使用する通信 ID を指定します。有効な範囲は 1 から 9991 から 16 です。デフォルトは 1 です。

「ボーレート」: 選択した通信ポートを設定するときに使用するボーレートを指定します。

「データビット」: データワードあたりのデータビット数を指定します。オプションは 5、6、7、8 です。

「パリティ」: データのパリティのタイプを指定します。オプションには「奇数」、「偶数」、「なし」があります。

「ストップビット」: データワードあたりのストップビット数を指定します。オプションは 1 または 2 です。

「フロー制御」: RTS および DTR 制御回線の利用方法を選択します。一部のシリアルデバイスと通信する際にはフロー制御が必要です。以下のオプションがあります。

- ・「なし」: このオプションでは、制御回線はトグル(アサート)されません。
- ・「DTR」: このオプションでは、通信ポートが開いてオンのままになっている場合に DTR 回線がアサートされます。
- ・「RTS」: このオプションでは、バイトを転送可能な場合に RTS 回線がハイになります。バッファ内のすべてのバイトが送信されると、RTS 回線はローになります。これは通常、RS232/RS485 コンバータハードウェアで使用されます。
- ・「RTS、DTR」: このオプションは DTR と RTS を組み合わせたものです。
- ・「RTS 常時」: このオプションでは、通信ポートが開いてオンのままになっている場合に、RTS 回線がアサートされます。
- ・「RTS 手動」: このオプションでは、「RTS 回線制御」で入力したタイミングプロパティに基づいて RTS 回線がアサートされます。これは、ドライバーが手動による RTS 回線制御をサポートしている場合 (またはプロパティが共有され、このサポートを提供するドライバーに 1 つ以上のチャンネルが属している場合) にのみ使用できます。「RTS 手動」を選択した場合、次のオプションから成る「RTS 回線制御」プロパティが追加されます。
  - ・「事前オン」: データ転送の前に RTS 回線を事前にオンにする時間を指定します。有効な範囲は 0 から 9999 ミリ秒です。デフォルトは 10 ミリ秒です。
  - ・「遅延オフ」: データ転送後に RTS 回線を解放するまでの時間を指定します。有効な範囲は 0 から 9999 ミリ秒です。デフォルトは 10 ミリ秒です。
  - ・「ポーリング遅延」: 通信のポーリングが遅延する時間を指定します。有効な範囲は 0 から 9999 です。デフォルトは 10 ミリ秒です。

● **ヒント**: 2 回線 RS 485 を使用している場合、通信回線上で "エコー" が発生することがあります。この通信はエコー除去をサポートしていないので、エコーを無効にするか、RS-485 コンバータを使用することをお勧めします。

## 実行動作

- ・「通信エラーを報告」: 低レベル通信エラーに関するレポートを有効または無効にします。オンにした場合、低レベルのエラーが発生するとイベントログに書き込まれます。オフにした場合、通常の要求の失敗は書き込まれますが、これと同じエラーは書き込まれません。デフォルトは「有効化」です。
- ・「アイドル接続を閉じる」: チャンネル上のクライアントによっていずれのタグも参照されなくなった場合、接続を閉じます。デフォルトは「有効化」です。
- ・「クローズするまでのアイドル時間」: すべてのタグが除去されてから COM ポートを閉じるまでサーバーが待機する時間を指定します。デフォルトは 15 秒です。

## モデム設定

- ・「モデム」: 通信に使用するインストール済みモデムを指定します。
- ・「接続タイムアウト」: 接続が確立される際に待機する時間を指定します。この時間を超えると読み取りまたは書き込みが失敗します。デフォルトは 60 秒です。
- ・「モデムのプロパティ」: モデムハードウェアを設定します。クリックした場合、ベンダー固有のモデムプロパティが開きます。
- ・「自動ダイヤル」: 電話帳内のエントリに自動ダイヤルできます。デフォルトは「無効化」です。詳細については、サーバーのヘルプで「モデム自動ダイヤル」を参照してください。
- ・「通信エラーを報告」: 低レベル通信エラーに関するレポートを有効または無効にします。オンにした場合、低レベルのエラーが発生するとイベントログに書き込まれます。オフにした場合、通常の要求の失敗は書き込まれますが、これと同じエラーは書き込まれません。デフォルトは「有効化」です。

- ・「**アイドル接続を閉じる**」: チャネル上のクライアントによっていずれのタグも参照されなくなった場合、モデム接続を閉じます。デフォルトは「有効化」です。
- ・「**クローズするまでのアイドル時間**」: すべてのタグが除去されてからモデム接続を閉じるまでサーバーが待機する時間を指定します。デフォルトは 15 秒です。

## 通信なしの動作

- ・「**読み取り処理**」: 明示的なデバイス読み取りが要求された場合の処理を選択します。オプションには「無視」と「失敗」があります。「無視」を選択した場合には何も行われません。「失敗」を選択した場合、失敗したことがクライアントに通知されます。デフォルト設定は「無視」です。

## 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。

プロパティグループ	<input checked="" type="checkbox"/> <b>書き込み最適化</b>	
一般	最適化方法	すべてのタグの最新の値のみを書き込み
シリアル通信	デューティサイクル	10
<b>書き込み最適化</b>		

## 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ています。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。

## チャンネルのプロパティ - 詳細

このグループは、チャン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。

プロパティグループ	<input type="checkbox"/> <b>非正規化浮動小数点処理</b>	
一般	浮動小数点値	ゼロで置換
シリアル通信	<input type="checkbox"/> <b>デバイス間遅延</b>	
書き込み最適化	デバイス間遅延 (ミリ秒)	0
<b>詳細</b>		
通信シリアル化		

「**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。

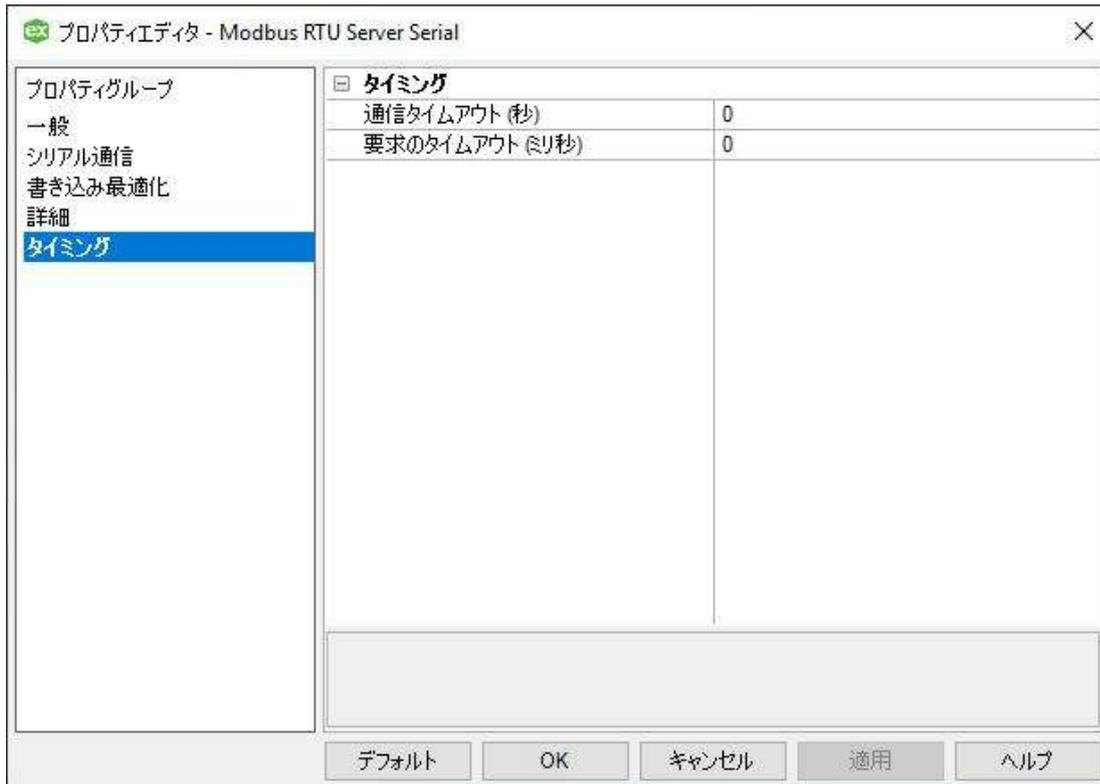
● **注記**: ドライバーが浮動小数点値をサポートしていない場合や、表示されているオプションだけをサポートする場合、このプロパティは無効になります。チャンネルの浮動小数点正規化の設定に従って、リアルタイムのドライバータグ (値や配列など) が浮動小数点正規化の対象となります。たとえば、EFM データはこの設定の影響を受けません。

● 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**」: 通信チャンネルが同じチャン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。

## チャンネルのプロパティ - タイミング



**「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を指定します。この時間が経過すると、ドライバーはそのチャンネル上のすべてのサーバーデバイスタグの品質を不良として設定します。「通信タイムアウト」が経過した後、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、デバイスとの通信を再確立します。あるいは、チャンネルのプロパティのタイミンググループで「通信タイムアウト」を0(ゼロ)に設定することでタイムアウトを無効にします。無効: 0、有効: 1-->64,800 秒 (18 時間)。

**「要求のタイムアウト」:** ドライバーが完全な要求フレームを受信するまで待機する時間を指定します。新しい要求の1つ目のバイトを受信するとただちに経過時間の計算が開始されます。この時間内に完全な要求フレームを受信しなかった場合、ドライバーは受信済みデータのバッファをフラッシュし、次に受信したバイトを新しい要求の開始と見なします。

**● ヒント:**

この設定は慎重に選択してください。「要求タイムアウト」設定の値は0から30,000ミリ秒の範囲であり、デフォルトは0です。0が入力された場合、ドライバーは次の式を使用して適切なタイムアウトを計算します。

$$T_{\text{default}} = 1000 * (\text{バイトあたりのビット数}) * 512 * 3 / \text{ボー}$$

これは512バイトのフレームを転送するのに必要な時間を3倍にしたものです。バイトあたりのビット数には、指定されているスタートビット、データ数、およびストップビットが含まれます。たとえば、ボーレートが9600、データビットが8、ストップビットが1の場合、デフォルトのタイムアウトは1600ミリ秒になります。ハードウェアが比較的短い要求フレームを送信するので、失敗した要求をデフォルトの計算(この例では1600ミリ秒)より短い時間で再試行させる場合、「要求タイムアウト」の設定を短くしてみてください。

「要求タイムアウト」は、そのチャンネル上の任意のデバイスによって送信される最も長い要求フレームを受信するのにかかる時間より短くしてはなりません。これは次の式を使用して計算されます。

$$T_{\text{min}} = 1000 * (\text{バイトあたりのビット数}) * (\text{最大フレーム長さ}) / \text{ボー}$$

## デバイスのプロパティ - 一般

プロパティグループ	<input type="checkbox"/> 識別	
一般	名前	Modbus
スキャンモード	説明	
メモリ	ドライバー	Modbus RTU Server Serial
	モデル	Modbus
	チャンネル割り当て	Modbus RTU Server Serial
	ID フォーマット	10 進数
	ID	1
	<input type="checkbox"/> 動作モード	
	データコレクション	有効化
	シミュレーション	いいえ
	<input type="checkbox"/> タグ数	
	静的タグ	1

### 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](#)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 フォーマット」: デバイス識別情報のフォーマット方法を選択します。オプションには「10 進数」、「8 進数」、「16 進数」があります。

「ID」: Modbus シリアルデバイスには 0 から 255 の範囲のデバイス ID が割り当てられます。

### 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。

## 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限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、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 - メモリ



### MemoryMemory

「**アドレス範囲**」: コイルとレジスタのアドレス範囲を 9999 から 65536 の任意の値に設定できます。指定した範囲以内でタグのアドレスを指定できます。

● **注記:**

1. タグが処理中の場合はアドレス範囲を変更できません。
2. アドレスの範囲が変更され、要求されたメモリアドレスが新しいアドレスの範囲外になった場合は、(Modbus クライアントからの) リモート要求が却下される可能性があります。
3. アドレス範囲が変更され、新しい上限が以前の上限よりも大きい場合、以前のデータがすべて保持され、残りのメモリは '0' に初期化されます。これに対し、新しい上限が以前の上限よりも小さい場合、新しいメモリサイズのみデータが保持され、残りのデータは失われます。Boolean のメモリを処理する場合、これには例外があります。

● この詳細、例、図については[メモリのアドレス指定](#)を参照してください。

「**ゼロベースアドレス指定**」: 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 (有効) の動作は Modicon PLC の規則に従います。

● **注記:** Daniels/Enron デバイスを使用している場合、「ゼロベースアドレス指定」を無効にする必要があります。

「**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 とする場合には有効にします。「最初の Word を下位とする」が有効になっている場合 (デフォルト)、最初の Word が下位と見なされ、Modicon Modsoft プログラミングソフトウェアの規則に従います。

● **注記:** Daniels/Enron デバイスを使用している場合、「最初の Word を下位とする」を無効にする必要があります。

「**最初の DWord を下位とする**」: 64 ビットデータ型には 4 つの連続するレジスタアドレス (それぞれ 2 つのグループが 2 つ) が使用されます。ユーザーはドライバーが最初のペア (つまり最初の 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。「最初の DWord を下位とする」が有効な場合は最初の DWord が下位と見なされ、無効な場合には 2 つ目の DWord が下位と見なされます。

● **注記:** Daniels/Enron デバイスを使用している場合、「最初の DWord を下位とする」を無効にする必要があります。

**「書き込みまで OPC 品質を不良に設定」:** このオプションは、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 クライアント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になります。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。

## メモリのアドレス指定

### アクセス可能なメモリ位置

- 出力コイル - 00001 から 065536
- 入力コイル - 10001 から 165536
- 内部レジスタ - 30001 から 365536
- 保持レジスタ - 40001 から 465536

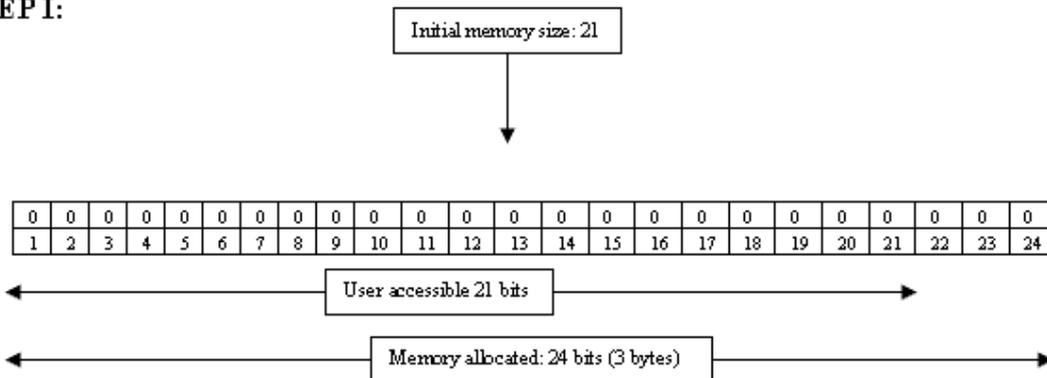
● これらの設定は変更可能です。詳細については、[メモリ](#)を参照してください。

コイルとレジスタのアドレス範囲を 9999 から 65536 の任意の値に設定できます。指定した範囲以内でタグのアドレスを指定できます。

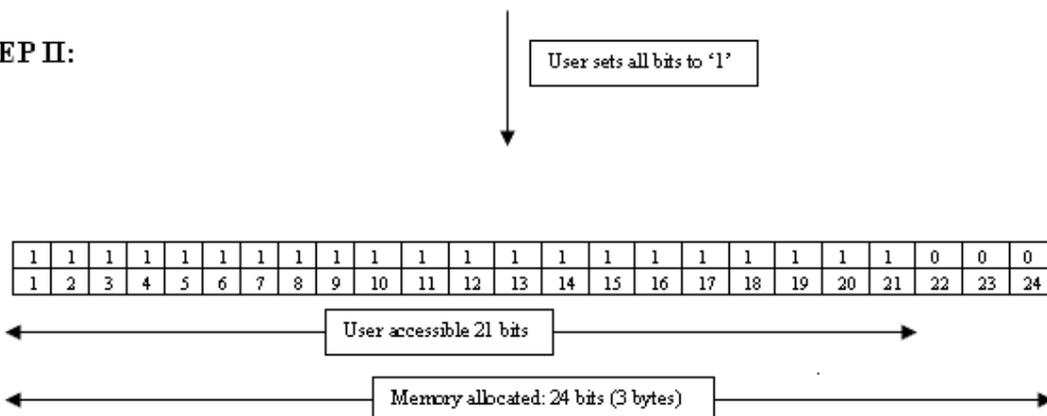
たとえば、初期メモリサイズが 21 である場合、Boolean では 3 バイト (連続するバイト) のメモリとして解釈されます。メモリサイズを 12 (2 バイト) に変更した場合、2 つのメモリサイズの小さい方が 2 バイトとなり、この量のデータが保持されます。12 ビットだけ保持されていると思われるかもしれませんが、16 ビット (2 バイト) が保持されています。メモリサイズが 12 なので、インデックス 12 のメモリまでしかアクセスできないため、通常はこれに気が付きません。メモリサイズを 22 (3 バイト) に増やした場合、以前のメモリから保持されるデータの量は 2 バイト (2 つのうち小さい方) となります。それまでは 12 ビットを操作できましたが (メモリサイズは 12)、ビット 13-16 には以前のデータが残っています (これは最初にメモリサイズが 21 であったときになんらかの値に初期化されている可能性があります)。

以下の図は、上記の例におけるコイルタイプのメモリを図で表したものです。

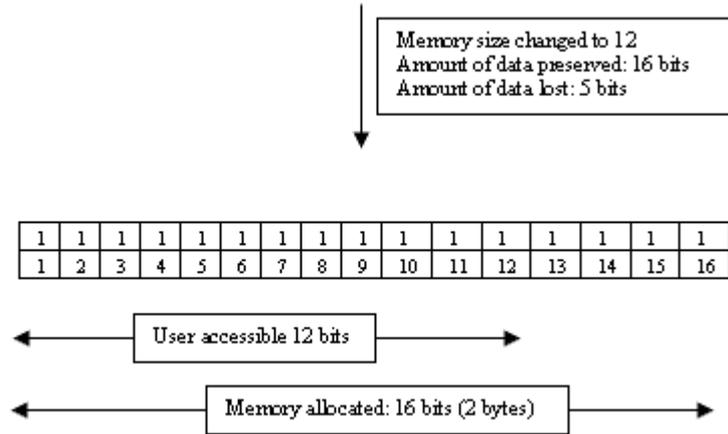
#### STEP I:



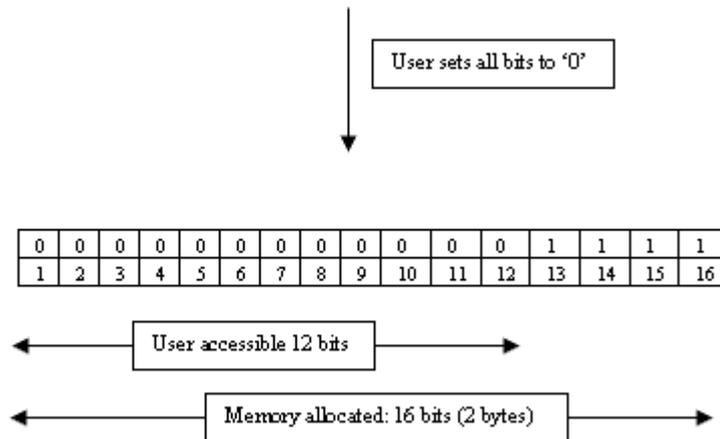
#### STEP II:



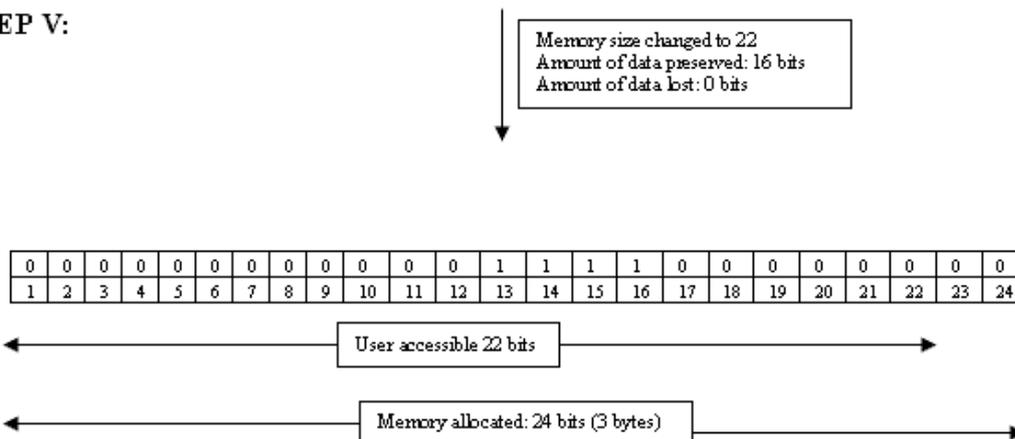
## STEP III:



## STEP IV:



## STEP V:



上記の Step V は、ビット 13-16 に Step II で初期化された古いデータが残っていることを示しています。Step V のすべてのビットが '0' であると思われるかもしれませんが、ビット 13-16 は Step II から繰り越されたものであり、まだ '1' に設定されています。

## 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 保持レジスタの範囲内でサポートされ、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

アドレスの様子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](#)

### [Daniels/Enron のアドレス指定](#)

## Modbus のアドレス指定

### 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	出カコイル
1	入カコイル
3	内部レジスタ
4	保持レジスタ

## Modbus のアドレス指定

次に示すアドレスの説明は、シミュレーション対象の各 Modbus サーバードバイスのクライアントアプリケーションのアクセスに関するものです。クライアントアプリケーションがシミュレーション対象の Modbus サーバードバイスのメモリを制御するため、すべての領域への読み取り/書き込みアクセスが可能です。動的に定義されるタグの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
出カコイル	000001-065536	<b>Boolean</b>	読み取り/書き込み
入カコイル	100001-165536	<b>Boolean</b>	読み取り/書き込み
内部レジスタ	300001-365536 300001-365535 3xxxxx.0-3xxxxx.15	<b>Word</b> 、Short、BCD Float、DWord、Long、 LBCD Boolean、Double	読み取り/書き込み
String の内部レジスタ HiLo バイトオーダー **	300001.2H-365536.240H ピリオドの後ろのビット番号は文字列長 (2 から 240 バイトの範囲) を示します。	<b>String</b>	読み取り専用
String の内部レジスタ LoHi バイトオーダー **	300001.2L-365536.240L ピリオドの後ろのビット番号は文字列長 (2 から 240 バイトの範囲) を示します。	<b>String</b>	読み取り専用
保持レジスタ	400001-465536 400001-465535 4xxxxx.0-4xxxxx.15	<b>Word</b> 、Short、BCD Float、DWord、Long、 LBCD Boolean、Double	読み取り/書き込み
String の保持レジスタ HiLo バイトオーダー	400001.2H-465536.240H ピリオドの後ろのビット番号は文字列長 (2 から 240 バイトの範囲) を示します。	<b>String</b>	読み取り/書き込み
String の保持レジスタ	400001.2L-465536.240L	<b>String</b>	読み取り/書き込み

アドレス	範囲*	データ型	アクセス
タ LoHi バイトオーダー	ピリオドの後ろのビット番号は文字列長 (2 から 240 バイトの範囲) を示します。		み

\*最大範囲はメモリのデバイスのプロパティの値によって決まります。詳細については、[メモリ](#)を参照してください。

\*\*このアドレスはファンクションコード 04 をサポートし、10 進アドレス指定のみに適用されます。

### 配列のサポート

内部レジスタと保持レジスタの位置では Boolean 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。入力コイルと出力コイルでも配列がサポートされます (Boolean データ型)。配列のアドレス指定には 2 つの方法があります。例では保持レジスタの位置が使用されています。

4xxxx [行数] [列数]

4xxxx [列数] - この方法では行数が 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。

### 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。バイトオーダーはアドレスの末尾に "H" または "L" を付けることによって指定します。

● **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。

### 文字列の例

- 400200 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、400200.100H と入力します。
- 400500 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、400500.78L と入力します。

## Daniels/Enron のアドレス指定

次に示すアドレスの説明は、シミュレーション対象の各 Daniels/Enron サーバーデバイスへのクライアントアプリケーションのアクセスに関するものです。クライアントアプリケーションがシミュレーション対象のサーバーデバイスのメモリを制御するため、すべての領域への読み取り書き込みアクセスが可能です。

該当する場合、動的に定義されるタグのデフォルトのデータ型を**太字**で示しています。次の表では、サーバーデバイスのアドレスの最大範囲が 0 から 65535 であるものとします。詳細については、[メモリ](#)を参照してください。

アドレス	範囲*	データ型	アクセス
出力コイル	000000-065535	<b>Boolean</b>	読み取り書き込み
入力コイル	100000-165535	<b>Boolean</b>	読み取り書き込み
内部レジスタ	300000-365535 300000-365534 300000-365532 300000.0-365535.15	<b>Word</b> 、Short、BCD Float、DWord、Long、 LBCD Double Boolean	読み取り書き込み
保持レジスタ	400000-405000 406000-407000 408000-465535	<b>Word</b> 、Short、BCD	読み取り書き込み

アドレス	範囲*	データ型	アクセス
	400000-404999 406000-406999 408000-465534	DWord、LBCD	
	400000-404999 405001-405999 406000-406999 408000-465534	Long	
	400000-404999 406000-406999 407001-407999 408000-465534	Float	
	400000-404997 406000-406997 408000-465532	Double	
Boolean の保持レジスタ	400000.xx-405000.xx 405001.yy-405999.yy 406000.xx-465535.xx  xx は 0-15 のビット番号 yy は 0-31 のビット番号	Boolean	読み取り/書き込み
String の保持レジスタ、HiLo バイトオーダー	400000.xxxH-405000. xxxH 406000.xxxH-407000. xxxH 408000.xxxH-465535. xxxH  xxx は文字列長 (2 から 240 バイトの範囲) を示します。	String	読み取り/書き込み
String の保持レジスタ、LoHi バイトオーダー	400000.xxxL-405000.xxxL 406000.xxxL-407000.xxxL 408000.xxxL-465535.xxxL  xxx は文字列長 (2 から 240 バイトの範囲) を示します。	String	読み取り/書き込み

\*最大範囲はメモリのデバイスプロパティの値によって決まります。詳細については、[メモリ](#)を参照してください。

## 配列のサポート

内部レジスタと保持レジスタの位置では Boolean 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。入力コイルと出力コイルでも配列がサポートされます (Boolean データ型)。配列のアドレス指定には 2 つの方法があります。例では保持レジスタの位置が使用されています。

4xxxx [行数] [列数]

4xxxx [列数] - この方法では行数が 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 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。

## 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。バイトオーダーはアドレスの末尾に "H" または "L" を付けることによって指定します。

● **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。

## 文字列の例

- 400200 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、400200.100H と入力します。
- 400500 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、400500.78L と入力します。

## 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。

**アドレスサイズが変更されました。| 以前のサイズ = <数値>、現在のサイズ = <数値>。**

---

**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のアドレスサイズが変更されました。

**解決策:**

新しいアドレスサイズが正しいことを確認してください。

### エラーマスクの定義

---

**B** = ハードウェアの故障を検出

**F** = フレーミングエラー

**E** = I/O エラー

**O** = 文字バッファオーバーラン

**R** = RX バッファオーバーラン

**P** = 受信バイトパリティエラー

**T** = TX バッファフル

## 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	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 ACKNOWLEDGE (否定応答)	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

## 3

32 ビットデータ型 16

## 5

5 桁のアドレス指定 17

## 6

6 桁のアドレス指定 17

64 ビットデータ型 16

## B

BCD 16

Boolean 16

## C

COM ID 6

COM ポート 5

## D

Daniels/Enron のアドレス指定 18

Double 16

DTR 6

DWord 16

## F

Float 16

## I

I/O エラー 21

ID 10

ID フォーマット 10

## L

LBCD 16

Long 16

## M

Modbus のアドレス指定 17

Modbus 例外コード 22

## O

OPC 品質 13

## R

RS-485 6

RS232 3

RS485 3

RTS 6

RX バッファオーバーラン 21

## S

Short 16

String 16

## T

TX バッファフル 21

## W

Word 16

## あ

アイドル接続を閉じる 6-7

アドレスサイズが変更されました。| 以前のサイズ = <数値>、現在のサイズ = <数値>。 21

アドレスの説明 17

アドレス範囲 12

## い

イベントログメッセージ 21

## え

エラーマスクの定義 21

## お

オーバーラン 21

## き

キャッシュからの初期更新 11

## く

クローズするまでのアイドル時間 6-7

## さ

サポートされるデバイス 3

## し

シミュレーション 10

シリアルポートの設定 5

シリアル通信 3, 5

## す

スキャンしない、要求ポールのみ 11

スキャンモード 11

ストップビット 3, 6

すべてのタグのすべての値を書き込み 7

すべてのタグの最新の値のみを書き込み 7

## せ

ゼロで置換 8

ゼロベースアドレス指定 12

## た

タイミング 8

タグに指定のスキャン速度を適用 11

タグ数 5

## ち

チャンネルのプロパティ-シリアル通信 5

チャンネルのプロパティ-一般 4

チャンネルのプロパティ-書き込み最適化 7

チャンネルのプロパティ-詳細 8

チャンネル割り当て 10

## て

データコレクション 10

データビット 3, 6

データ型の説明 16

デバイス間遅延 8

デューティサイクル 7

## と

ドライバー 10

## な

なし 5

## は

ハードウェアの破損 21

パリティ 3, 6, 21

## ふ

ファンクションコード 3  
フレーミング 21  
ブロードキャストコマンド 4  
フロー制御 3, 6

## ほ

ポーリング遅延 6  
ボーレート 3, 6

## め

メモリ 12  
メモリのアドレス指定 14  
メモリ位置 14

## も

モデム 5-6  
モデム設定 6  
モデル 10

## 漢字

概要 3  
共有 5  
最初の DWord を下位とする 12  
最初の Word を下位とする 12  
最適化方法 7  
事前オン 6  
自動ダイヤル 6  
識別 4, 10  
実行動作 6  
出力コイル 14  
診断 5  
接続タイプ 5  
接続のタイムアウト 6  
設定 3  
説明 10  
遅延オフ 6

通信エラーを報告 6  
通信タイムアウト 9  
通信なしの動作 7  
通信プロトコル 3  
読み取り処理 7  
内部レジスタ 14  
入力コイル 14  
配列のサポート 18-19  
非 Boolean タグの最新の値のみを書き込み 7  
非正規化浮動小数点処理 8  
物理メディア 5  
文字列のサポート 18-19  
保持レジスタ 14  
未修正 8  
名前 10  
要求のタイムアウト 9