QEC スタートガイド 17：EtherCAT 経由で Modbus RTU を送信して IO を制御する（2）

本ガイドでは、QEC-M-01 と QEC-RXXHU シリーズを使用して Modbus RTU を送信する方法を説明します。

QEC の PoE（Power over Ethernet）に関する注意事項

QEC 製品では、PoE タイプと非 PoE タイプを簡単に見分けることができます。RJ45 コネクタのハウジングが赤色の場合は PoE タイプ、黒色の場合は非 PoE タイプです。



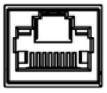
非 PoE タイプ



PoE タイプ

PoE（Power over Ethernet）は、ネットワークケーブル経由で電力を供給する機能です。QEC では配線を削減するためにオプションで PoE 機能を搭載できます。実際の運用では、PoE の採用はシステム構成機器に応じて判断されるため、評価・テストの際は次の点に注意してください。

1. QEC の PoE 機能は EtherCAT P とは異なり、互換性はありません。QEC の PoE 機能は PoE Type B に準拠しており、ピン機能は次のとおりです。



ピン番号	信号名	ピン番号	信号名
1	LAN1_TX+	2	LAN1_TX-
3	LAN1_RX+	4	VS+
5	VP+	6	LAN1_RX-
7	VS- (GND)	8	VP- (GND)

   * PoE 対応 LAN は赤色ハウジング、通常の LAN は黒色ハウジングです。
   * L4、L5、L7、L8 ピンはオプションで、RJ45 の電源入出力用です。

2. PoE 対応機器と非 PoE 機器を接続する場合は、必ずイーサネットケーブルの 4、5、7、8 番ピンを切断してください（例：PoE 対応の QEC EtherCAT M デバイスを他社製の EtherCAT Sub デバイスと接続する場合）。

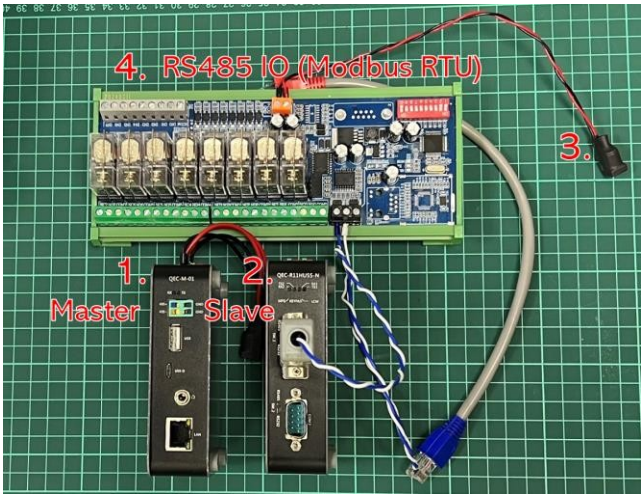


3. QEC の PoE 給電能力は最大 24V/3A です。

ハードウェアの接続と配線

ここでは次の機器を使用します。

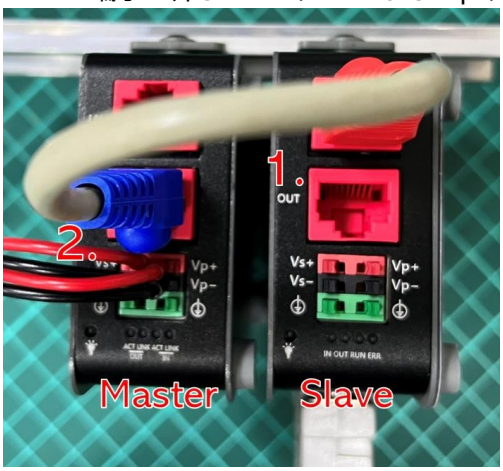
1. QEC-M-01P（EtherCAT M デバイス／PoE）
2. QEC-R11HU5S-N（EtherCAT HID Sub デバイス／PoE、UART×2 対応）
3. 24V 電源、EU タイプ端子ケーブル、RJ45 ケーブル
4. RS-485 通信 IO モジュール（プロトコル：Modbus RTU）



QEC-M-01P

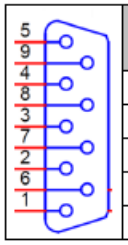
PoE 機能付きの QEC EtherCAT M デバイスです。

1. EtherCAT Out ポート（上側）から RJ45 ケーブルで QEC-R11HU9S の EtherCAT In ポートに接続します（PoE で給電）。
2. EU 端子を介して Vs+/Vs- および Vp+/Vp- に 24V 電源を接続します。



QEC-R11HU5S-N

- QEC-R11HU5S-N の COM ポートでは、RS-485- 信号が 1 番ピン、RS-485+ 信号が 2 番ピンです。



No.	Pin Assignment	No.	Pin Assignment
1	RS485-	6	DSR
2	RS485+/RXD	7	RTS
3	TXD	8	CTS
4	DTR	9	VCC
5	GND	-	-

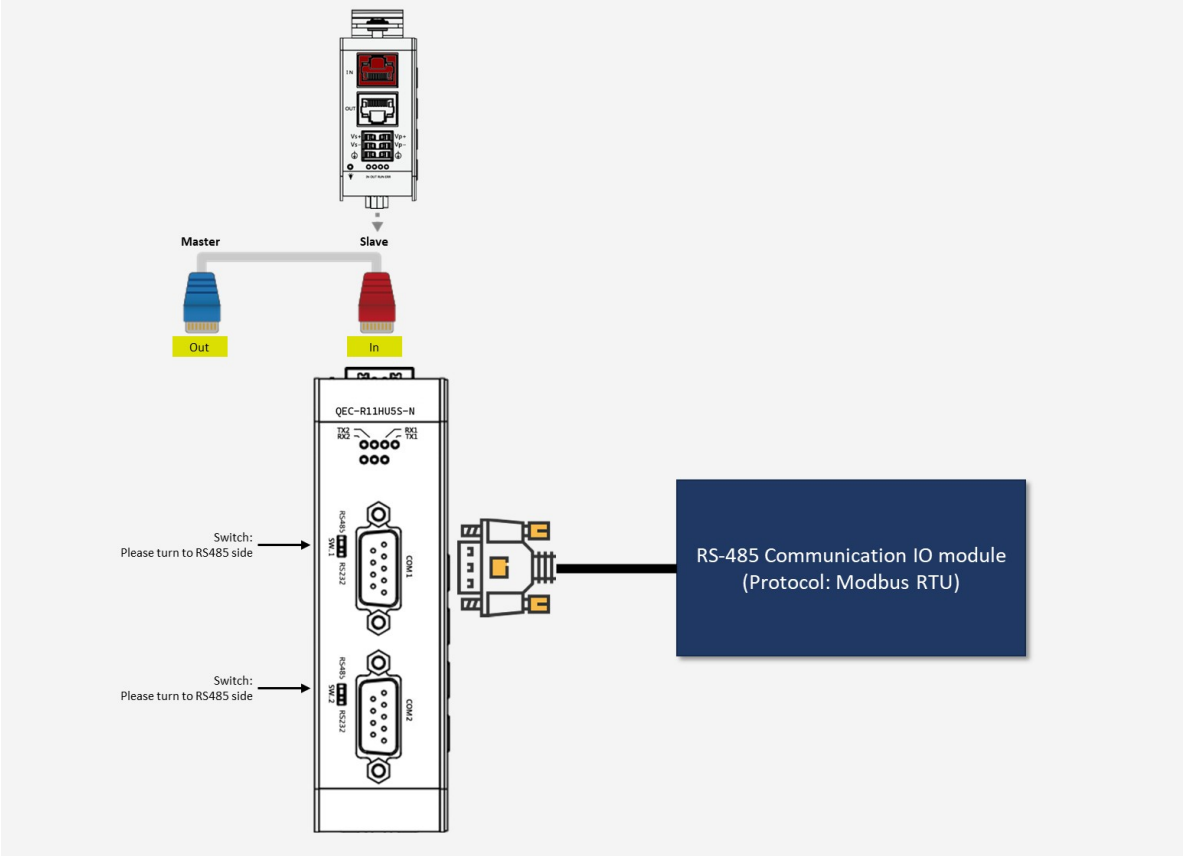
* Note: RS232 and RS485 cannot be used simultaneously.

- この例では RS-485 を使用します。

  ※注意：2 つの UART ポートの横にはそれぞれスイッチがあり、RS-232 と RS-485 の機能を切り替えられます。スイッチを切り替えた後は、デバイスの電源を入れ直してください。

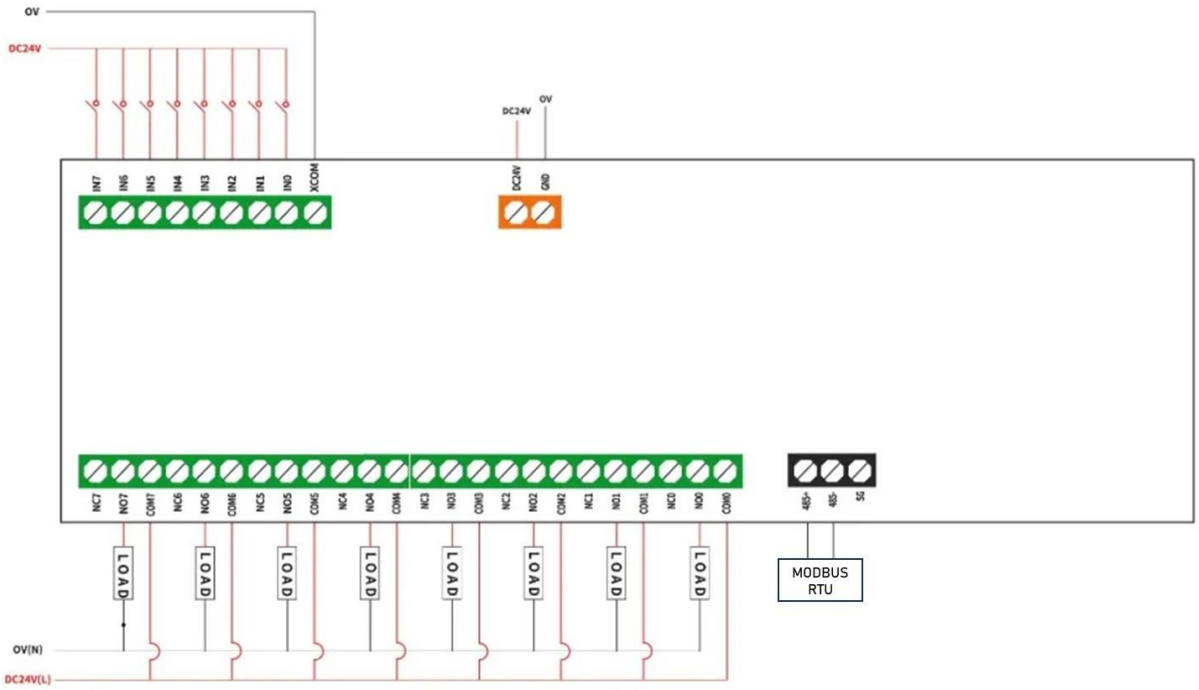


- QEC-R11HU5S-N の COM1 と RS-485 通信 IO モジュール(通信プロトコル：Modbus RTU)を、RS-485 伝送ケーブルで接続します。



※注意：QEC-RXXHU シリーズの RS-485 では、信号品質を最適化するため、使用環境に応じて 120 Ω の終端抵抗を追加する必要があります。終端抵抗は RS-485 のインピーダンスを整合させて信号の反射を抑えるもので、長距離や高ノイズの環境では特に重要です。RS-485 ネットワークを設計する際は、ネットワークの長さとデバイスの台数を考慮して終端抵抗の要否を判断することが、通信の信頼性と効率を確保するうえで重要です。

この例で使用している RS-485 通信 IO モジュール(通信プロトコル：Modbus RTU)は、Huaqingjun Electronics Co., Ltd. 製の RS-485 通信モジュールで、回路構成は次のとおりです。

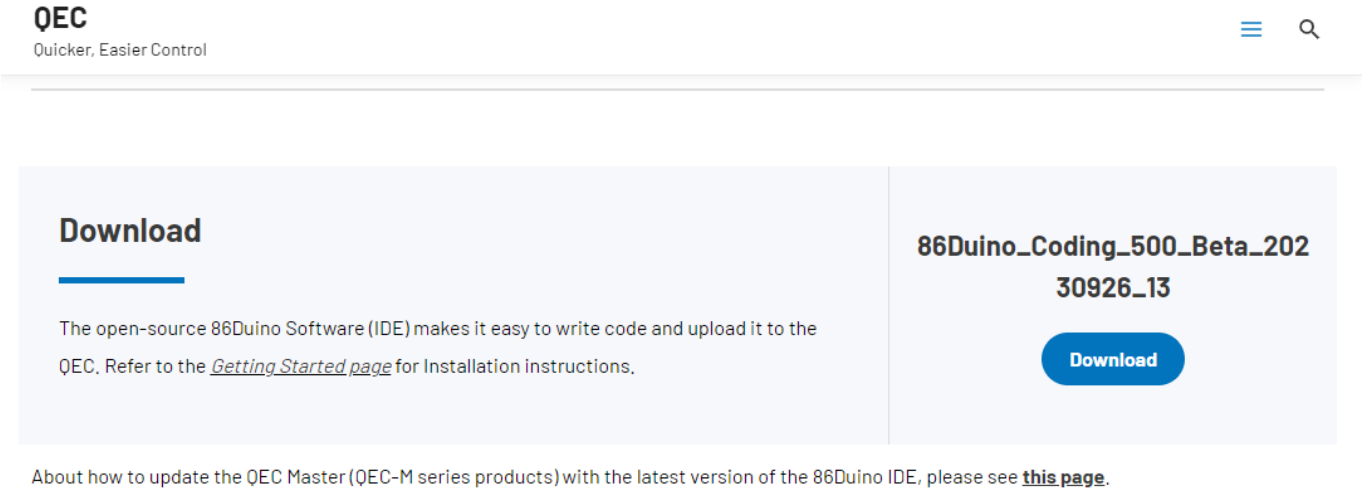


（Input compatible NPN and PNP）

ご要望やご質問がある場合は、Taobao の Huaqingjun 旗艦店 にお問い合わせください。

ソフトウェア／開発環境：86Duino IDE

86Duino IDE は https://www.qec.tw/software/ からダウンロードしてください。

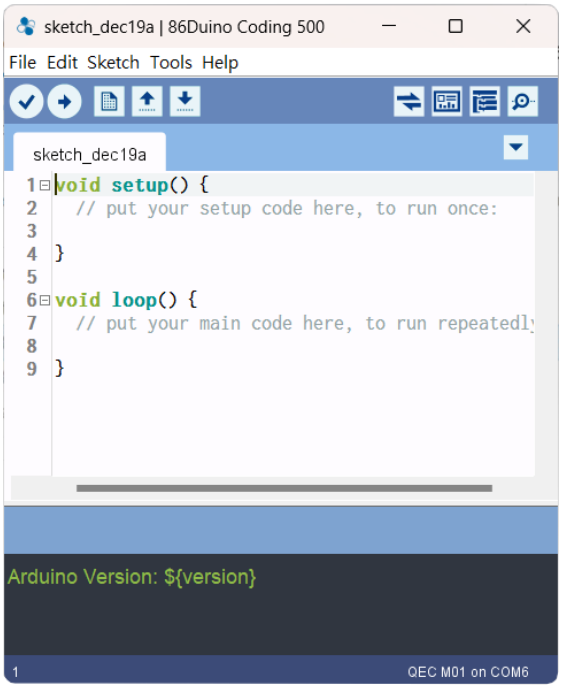


ダウンロードした zip ファイルを解凍してください。追加のソフトウェアをインストールする必要はなく、86duino.exe をダブルクリックするだけで IDE が起動します。



※注意：Windows で警告が表示された場合は、「詳細情報」を 1 回クリックしてから「実行」ボタンを 1 回クリックしてください。

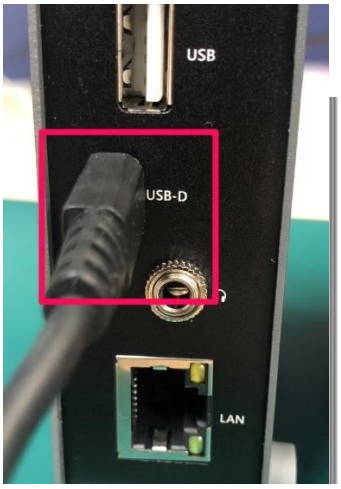
86Duino Coding IDE 500 以降の画面は次のとおりです。



PC との接続と環境設定

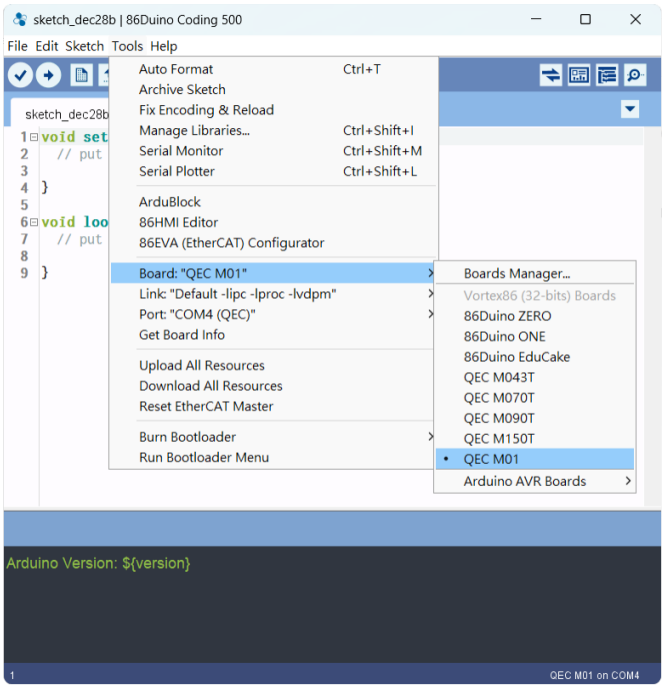
次の手順で環境を設定します。

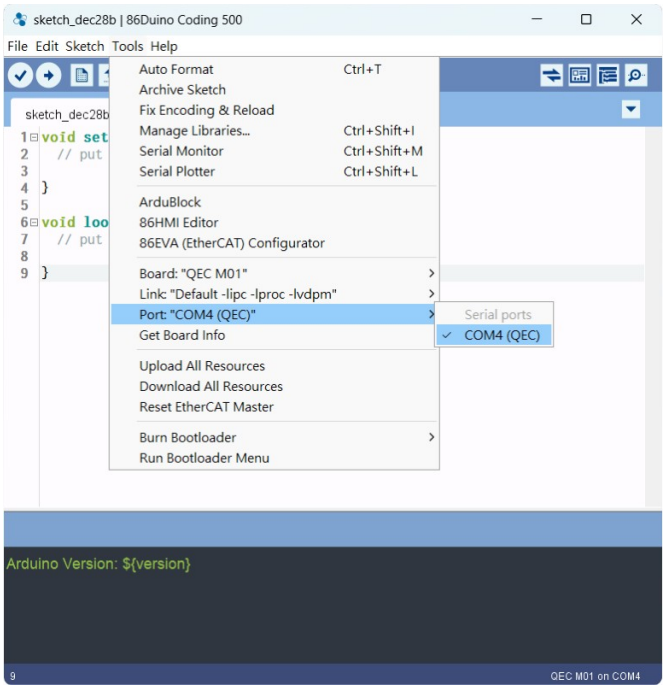
1. Micro USB－USB ケーブルで、QEC-M-01P を PC(86Duino IDE をインストール済み)に接続します。
2. QEC の電源を入れます。
3. PC で「Device Manager(デバイス マネージャー)」(Win+X キーを押して表示されるメニューから選択)を開き、「Ports (COM & LPT)(ポート(COM と LPT))」を展開します。「Prolific PL2303GC USB Serial COM Port (COMx)」が検出されていることを確認してください。表示されない場合は、必要なドライバーをインストールする必要があります。(Windows 用 PL2303 ドライバーはこちらからダウンロードできます)





4. 86Duino IDE を起動します。
5. ボードの選択：IDE のメニューから Tools(ツール)> Board(ボード)> QEC-M-01(または使用している QEC-M M デバイスの型番)を選択します。
6. ポートの選択：IDE のメニューから Tools(ツール)> Port(シリアルポート)を選択し、QEC-M M デバイスが接続されている USB ポート(この例では COM4 (QEC))を選択します。





開発方法 1：コードを記述する

EtherCAT M デバイス（QEC-M-01P）と HID Sub デバイス（QEC-R11HU9S-N）は、86Duino IDE の EtherCAT ライブラリで設定とプログラミングができます。Arduino の開発環境は主に `setup()` と `loop()` の 2 つの部分で構成され、それぞれ初期化処理とメインプログラムに対応しています。EtherCAT ネットワークを動作させる前に、一度設定を行う必要があります。EtherCAT デバイスは Pre-OP モードから OP モードへ移行させます。

以下の例では、EtherCAT ライブラリと Modbus ライブラリを使用して、EtherCAT 通信と Modbus 通信を組み合わせます。まず、EtherCAT M デバイスと EtherCAT デバイス（QEC-R11HU5S）を初期化し、EtherCAT ネットワークを設定します。次に、仮想シリアルポート（COM1）を設定し、ボーレートを 115200 にします。メインループ（`loop()`）では、Modbus コマンドで ID 30 の特定のスレーブ機器から保持レジスタのアドレス 4 の値を 1 秒ごとに読み取り、その値をシリアルモニタに 2 進数で表示します。読み取りに失敗した場合は、エラー処理を追加できます。

QEC の Sub デバイスを使用する場合は、専用の QEC EtherCAT Sub デバイスライブラリを利用できます。たとえば QEC-R11HU5S では EthercatDevice_QECRXXHU Class の機能を使用できます。EtherCAT 上で Modbus を使用する場合は、VirtualSerial 機能を使用する必要があります。Modbus ライブラリの詳細は Modbus Library を参照してください。
サンプルコードは次のとおりです。VirtualSerial は EtherCAT ライブラリ（EtherCAT.h）に統合されています。

```
#include "Ethercat.h"
#include "Modbus.h"

// デバイスのグローバルインスタンス領域
EthercatMaster EcatMaster;
EthercatDevice_QECR11HU5S Slave;
MyVirtualSerial VirtualSerial1;
ModbusMaster bus;

void setup() {
  Serial.begin(115200);

  EcatMaster.begin(ECAT_ETH_0, NULL);
  Slave.attach(0, EcatMaster, ECAT_SLAVE_NO);
  EcatMaster.start();

  VirtualSerial1.init(&Slave, COM1);
  VirtualSerial1.begin(115200);

  bus.begin(MODBUS_RTU, VirtualSerial1);
}

void loop() {
  static uint32_t lastTime = 0;
```

```
  uint32_t currentTime = millis();
  // 1 秒経過したかを確認
  if (currentTime - lastTime >= 1000) {


    uint16_t data[1];
    // uint8_t readHoldingRegisters(uint8_t slave_id, uint16_t address, uint16_t size, uint16_t *data);
    uint8_t result = bus.readHoldingRegisters(30, 4, 1, data);
    if (result == 0) {
      Serial.print("Holding Register 4 Value: ");
      Serial.println(data[0], BIN);
    } else {
      // エラー処理
    }
    lastTime = millis();   // 最後に実行した時刻を更新
  }
}
```

コードの書き込みが完了すると、Modbus コマンドで ID 30 の特定のスレーブ機器から保持レジスタのアドレス 4 の値を 1 秒ごとに読み取り、シリアルモニタに 2 進数で表示できます。読み取りに失敗した場合は、エラー処理を追加できます。この構成は、遠隔監視システムなど、定期的なデータ更新を必要とする自動化制御の用途に利用できます。保持レジスタの値を 1 秒ごとに読み取ることで、デバイスの状態を継続的に監視し、データのリアルタイム更新や異常検出が可能になります。

※注意：コードを記述したら、ツールバーの をクリックしてコンパイルします。コンパイルが完了しエラーがないことを確認したら、 をクリックして書き込みます。書き込みが完了するとプログラムが実行されます。

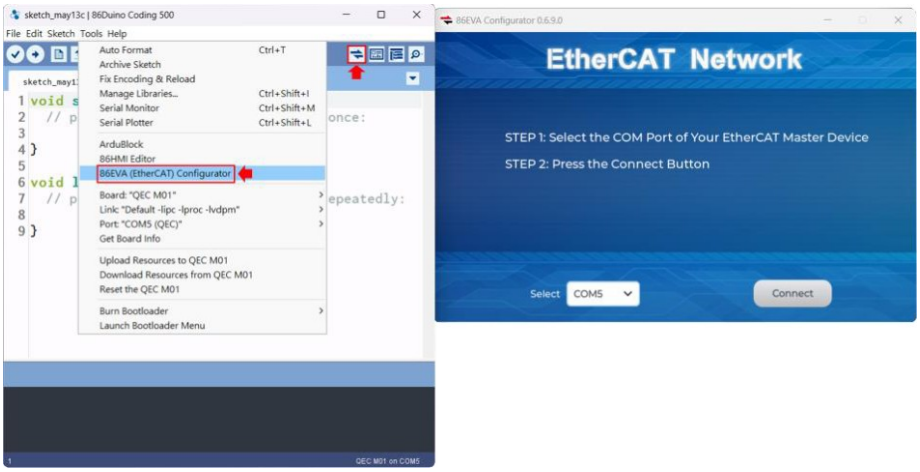


開発方法 2：86EVA とコードを使用する

86EVA は、86Duino IDE の EtherCAT ライブラリをベースにしたグラフィカルな EtherCAT 設定ツールで、86Duino IDE の開発キットの一部です。86EVA で EtherCAT M デバイス(QEC-M-01P)と HID Sub デバイス(QEC-R11HU9S-N)を設定してから、プログラミングを始めます。

ステップ 1：86EVA を起動してスキャンする

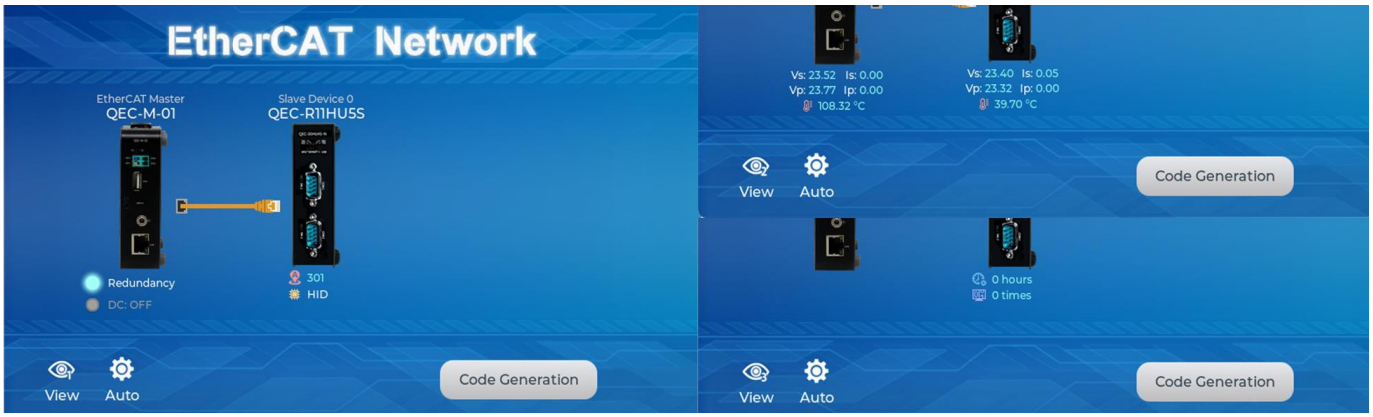
86EVA ツールは次のボタンから起動できます。



QEC-M-01P の正しい COM ポートが選択されていることを確認したら、Connect ボタンを押して EtherCAT ネットワークのスキャンを開始します。



EtherCAT ネットワークのスキャンが完了すると、接続されているデバイスが表示されます。左下の「View」ボタンを押すと、デバイスの状態(電圧、電流、温度：View2)と稼働時間(時間：View3)を確認できます。



ステップ 2：パラメーターを設定する

スキャンされたデバイスの画像をダブルクリックすると、対応するパラメーター設定画面が開きます。



QEC-M-01

QEC-M-01 の画像をダブルクリックして、パラメーター設定を表示します。

この例ではデフォルト設定のまま変更しません。左上の「Back（戻る）」をクリックして元の画面に戻ってください。

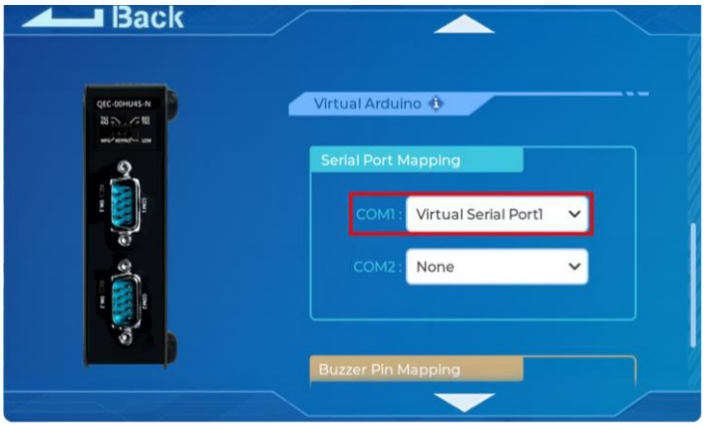


QEC-R11HU5S-N

QEC-R11HU5S の画像をダブルクリックして、パラメーター設定を表示します。



下にスクロールして「Serial Port Mapping」セクションに進み、「Serial Port Mapping」エリアの COM1 のドロップダウンで「Virtual Serial Port1」を選択します。



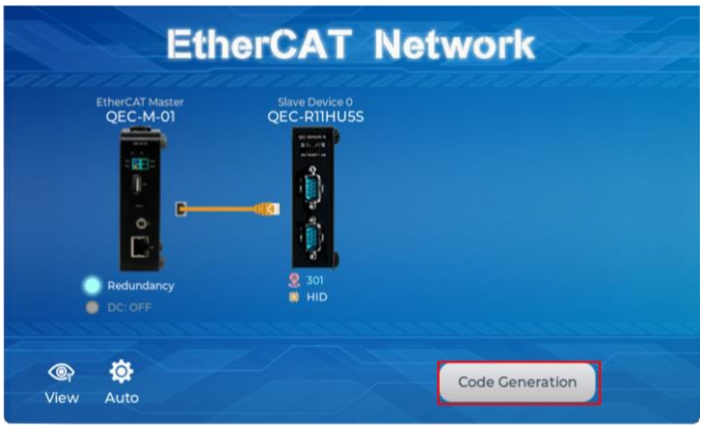
設定が終わったら、左上の「Back（戻る）」をクリックして元の画面に戻ります。



これにより、QEC-R11HU5S の COM1 が EVA の Virtual Serial Port1 に割り当てられます。

ステップ 3：コードを生成する

デバイスのパラメーターを設定したら、ホーム画面に戻り、右下の「Code Generation(コード生成)」ボタンを押します。



生成が完了したら OK ボタンをダブルクリックして 86EVA を閉じます。操作しない場合も 10 秒後に自動的に閉じます。



生成されるコードとファイルは次のとおりです。

- sketch_may13c：メインプロジェクト(.ino ファイル、ファイル名はプロジェクト名により異なります)
- GPT.h：ChatGPT に参照させるためのパラメーター
- myeva.cpp：86EVA の C++ コード
- myeva.h：86EVA のヘッダーファイル



補足：86EVA でコードを生成すると、メインプログラム(.ino)に次のコードが自動的に追加されます。いずれかが欠けていると 86EVA は動作しません。

1. `#include “myeva.h”`：EVA ヘッダーファイルのインクルード
2. `setup();` 内の `EVA.begin()`：EVA 機能の初期化

ステップ 4:コードを記述する

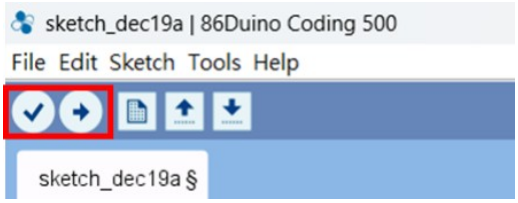
以下の例では、EtherCAT ライブラリと Modbus ライブラリを使用して、EtherCAT 通信と Modbus 通信を組み合わせます。まず、EtherCAT M デバイスと EtherCAT デバイス（QEC-R11HU5S）を初期化し、EtherCAT ネットワークを設定します。次に、仮想シリアルポート（COM1）を設定し、ボーレートを 115200 にします。メインループ（`loop()`）では、Modbus コマンドで ID 30 の特定のスレーブ機器から保持レジスタのアドレス 4 の値を 1 秒ごとに読み取り、その値をシリアルモニタに 2 進数で表示します。読み取りに失敗した場合は、エラー処理を追加できます。
Modbus ライブラリの詳細は Modbus Library を参照してください。

```
#include "myeva.h"
#include "Modbus.h"

ModbusMaster bus;

void setup() {
  Serial.begin(115200);
 EVA.begin();
  // 一度だけ実行するセットアップコードをここに記述
  bus.begin(MODBUS_RTU, VirtualSerial1);
}

void loop() {
  static uint32_t lastTime = 0;
  uint32_t currentTime = millis();
  // 1 秒経過したかを確認
  if (currentTime - lastTime >= 1000) {
    uint16_t data[1];
    // uint8_t readHoldingRegisters(uint8_t slave_id, uint16_t address, uint16_t size, uint16_t *data);
    uint8_t result = bus.readHoldingRegisters(30, 4, 1, data);
    if (result == 0) {
      Serial.print("Holding Register 4 Value: ");
      Serial.println(data[0], BIN);
    } else {
      // エラー処理
    }
    lastTime = millis();   // 最後に実行した時刻を更新
  }
}
```

コードの書き込みが完了すると、Modbus コマンドで ID 30 の特定のスレーブ機器から保持レジスタのアドレス 4 の値を 1 秒ごとに読み取り、シリアルモニタに 2 進数で表示できます。読み取りに失敗した場合は、エラー処理を追加できます。

この構成は、遠隔監視システムなど、定期的なデータ更新を必要とする自動化制御の用途に利用できます。保持レジスタの値を 1 秒ごとに読み取ることで、デバイスの状態を継続的に監視し、データのリアルタイム更新や異常検出が可能になります。

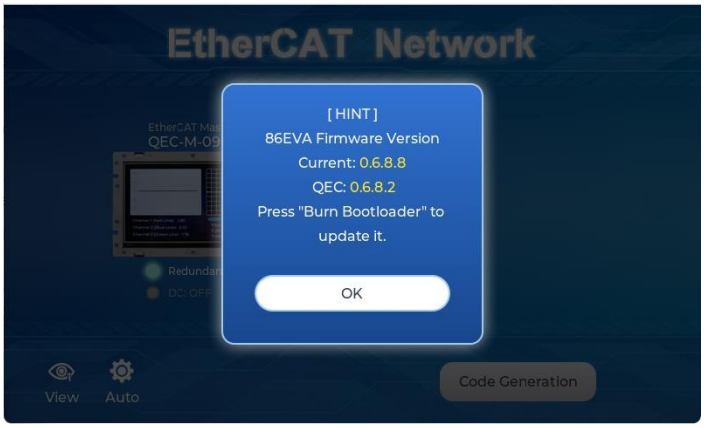
※注意：コードを記述したら、ツールバーの をクリックしてコンパイルします。コンパイルが完了しエラーがないことを確認したら、 をクリックして書き込みます。書き込みが完了するとプログラムが実行されます。



トラブルシューティング

QEC-M-01 にコードを書き込めない場合

コードを正常に書き込めない場合は、86EVA を開いて QEC EtherCAT M デバイスの環境に異常がないか確認してください。下図のように異常がある場合は、QEC EtherCAT M デバイスの環境を更新してください。更新対象は Bootloader、EtherCAT ファームウェア、EtherCAT ツールの 3 項目です。



以下、更新の手順を説明します。

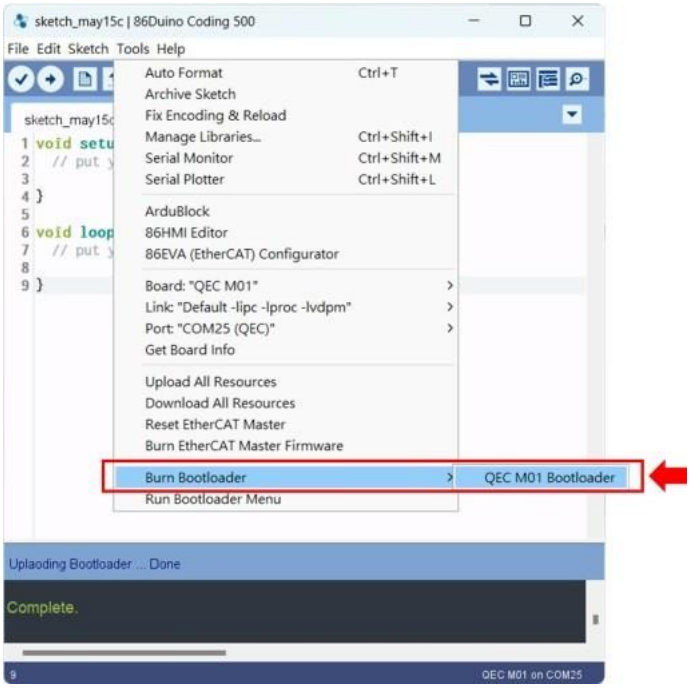
ステップ 1：QEC-M の準備

1. 86Duino IDE 500（またはそれ以降のバージョン）をダウンロードしてインストールします。ダウンロードは Software ページから行えます。
2. QEC-M の接続：USB ケーブルで QEC-M をコンピューターに接続します。
3. 86Duino IDE の起動：インストールが完了したら、86Duino IDE を起動します。
4. ボードの選択：IDE のメニューから「Tools（ツール）」>「Board（ボード）」>「QEC-M-01」（または使用している QEC-M の型番）を選択します。
5. ポートの選択：IDE のメニューから「Tools（ツール）」>「Port（シリアルポート）」を選択し、QEC-M が接続されている USB ポートを選択します。

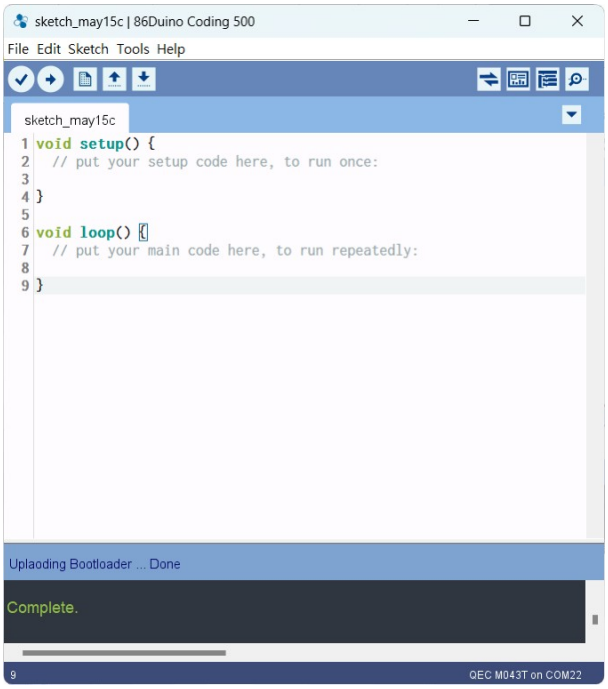
ステップ 2：「Burn Bootloader（ブートローダを書き込む）」をクリック

QEC-M 製品を接続したら、「Tools（ツール）」>「Burn Bootloader（ブートローダを書き込む）」を選択します。現在選択されている QEC-M の名前が表示されます。これをクリックすると更新が始まります。更新には約 5～20 分かかります。

QEC-M-01：



ステップ 3：更新の完了



以上の手順で、QEC-M の開発環境が最新バージョンに更新されます。

詳細情報やサンプルのご依頼は、info@icop.com.tw までメールでお問い合わせいただくか、最寄りの ICOP 拠点 にお電話いただくか、正規販売代理店 にお問い合わせください。

日本語版資料は、英語版を翻訳したもので、内容に相違が生じる場合には原文を優先します。