スタートガイド

QEC-R11CFFG：EtherCAT リモート デジタル I/O、アナログ I/O、シリアル（86EVA 編）

86Duino Coding IDE 501

EtherCAT ライブラリ

(Version 1.0)

改訂履歴

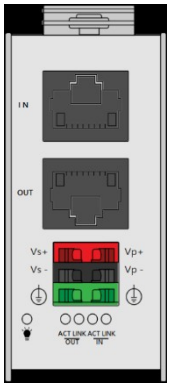
日付	バージョン	内容
2025/9/28	Version1.0	初版リリース。

はじめに

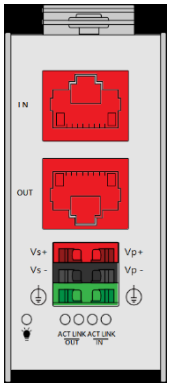
本ガイドでは、EtherCAT M デバイス QEC-M-01 と QEC-R11CFFG シリーズ（DIO・AIO・シリアル COM ポートを備えた EtherCAT 複合 I/O モジュール）の使用方法を説明します。

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。



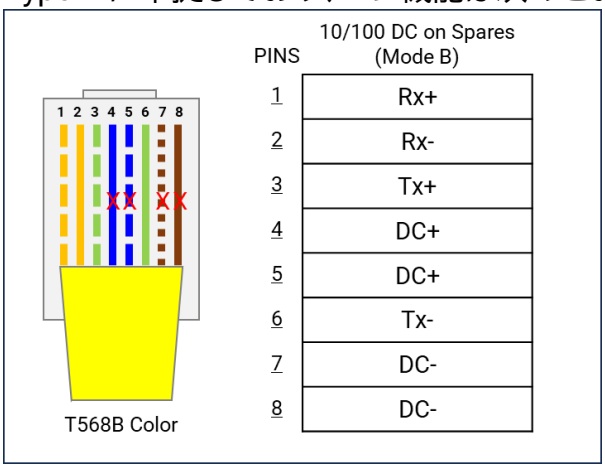
非 PoE タイプ



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。



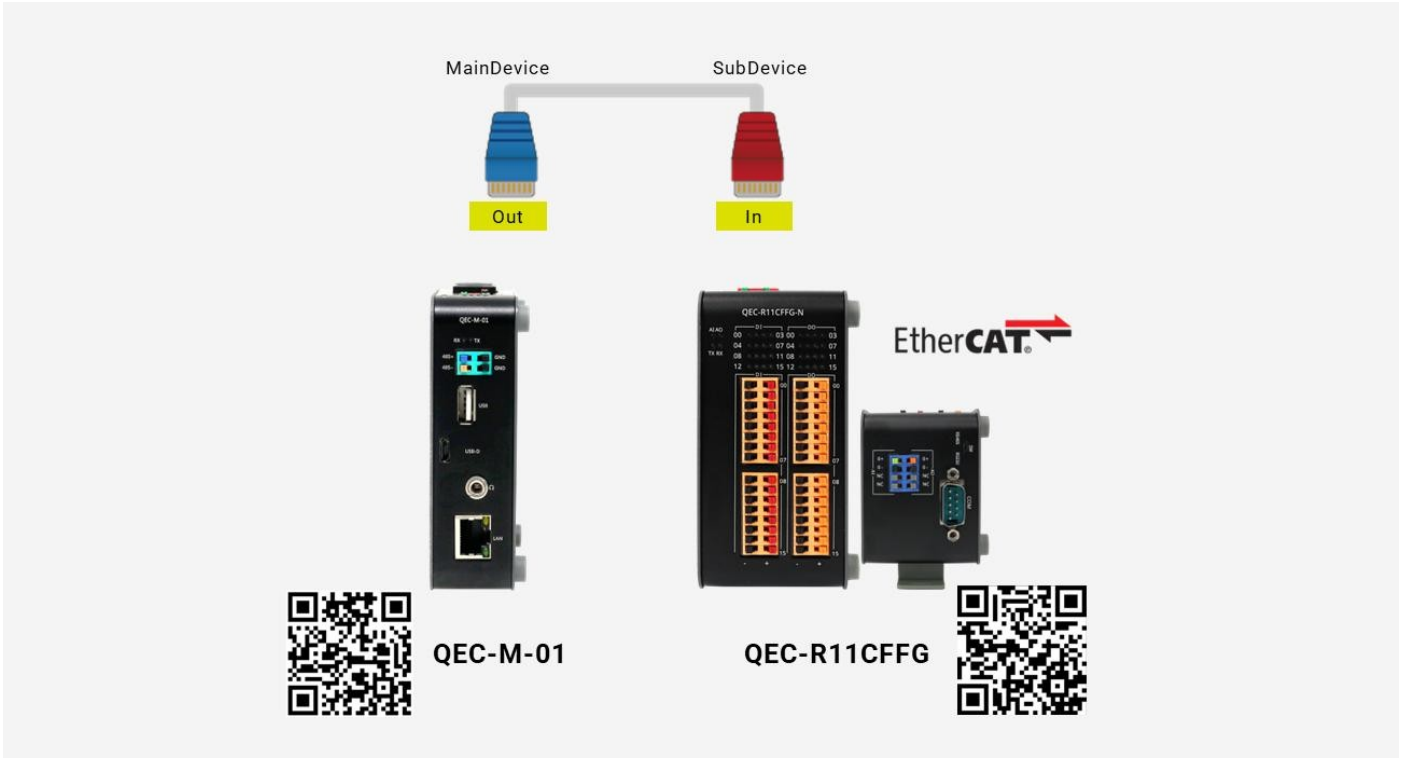
2. PoE 対応機器と非 PoE 機器を接続する場合は、必ずイーサネットケーブルの 4、5、7、8 番ピンを切断してください（例：PoE 対応の QEC EtherCAT M デバイスを他社製の EtherCAT Sub デバイスと接続する場合）。

3. QEC の PoE 給電能力は最大 24V/3A です。

1. 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。

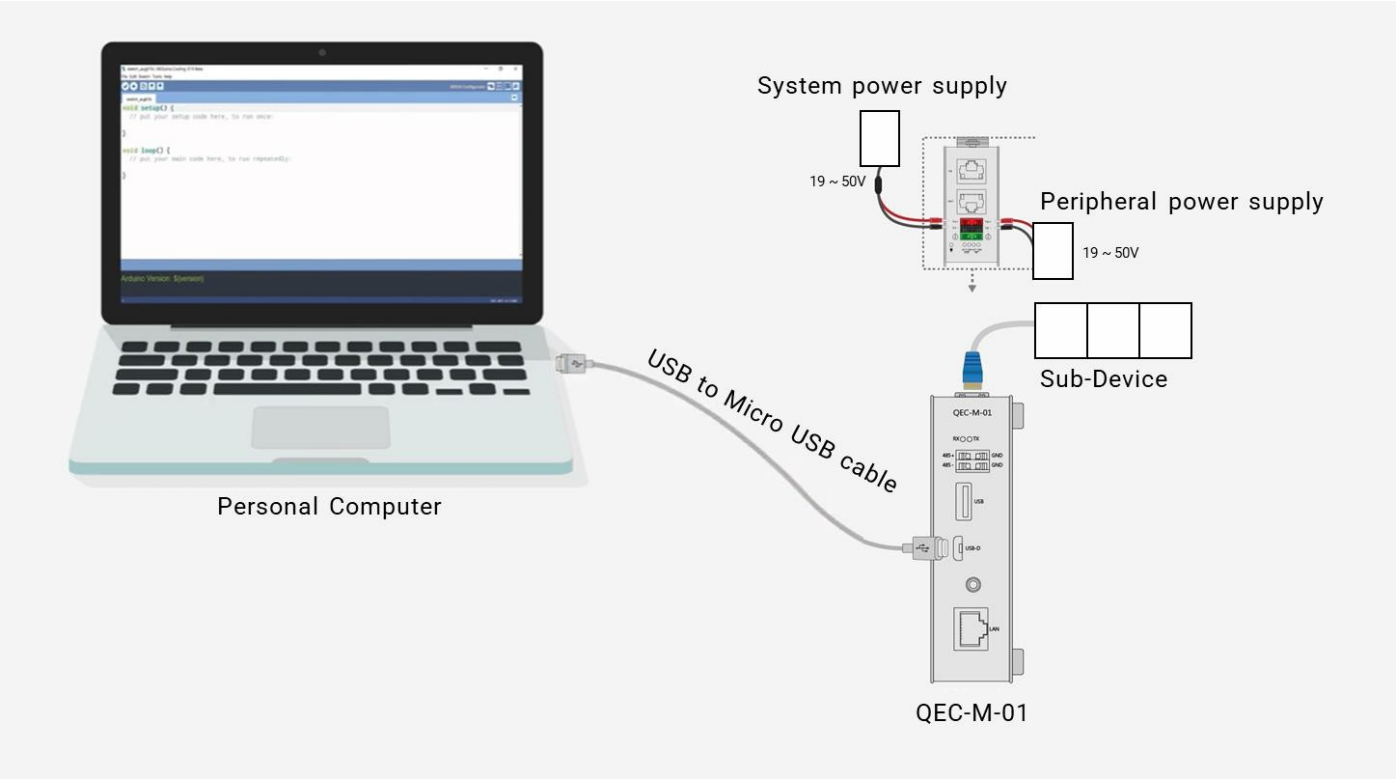
1. QEC-M-01（EtherCAT M デバイス）
2. QEC-R11CFFG シリーズ（DIO・AIO・シリアル COM ポートを備えた EtherCAT 複合 I/O モジュール）
3. 24VDC 電源、EU タイプ端子ケーブル、LAN ケーブル



1.1 QEC-M-01

QEC EtherCAT M デバイスです。

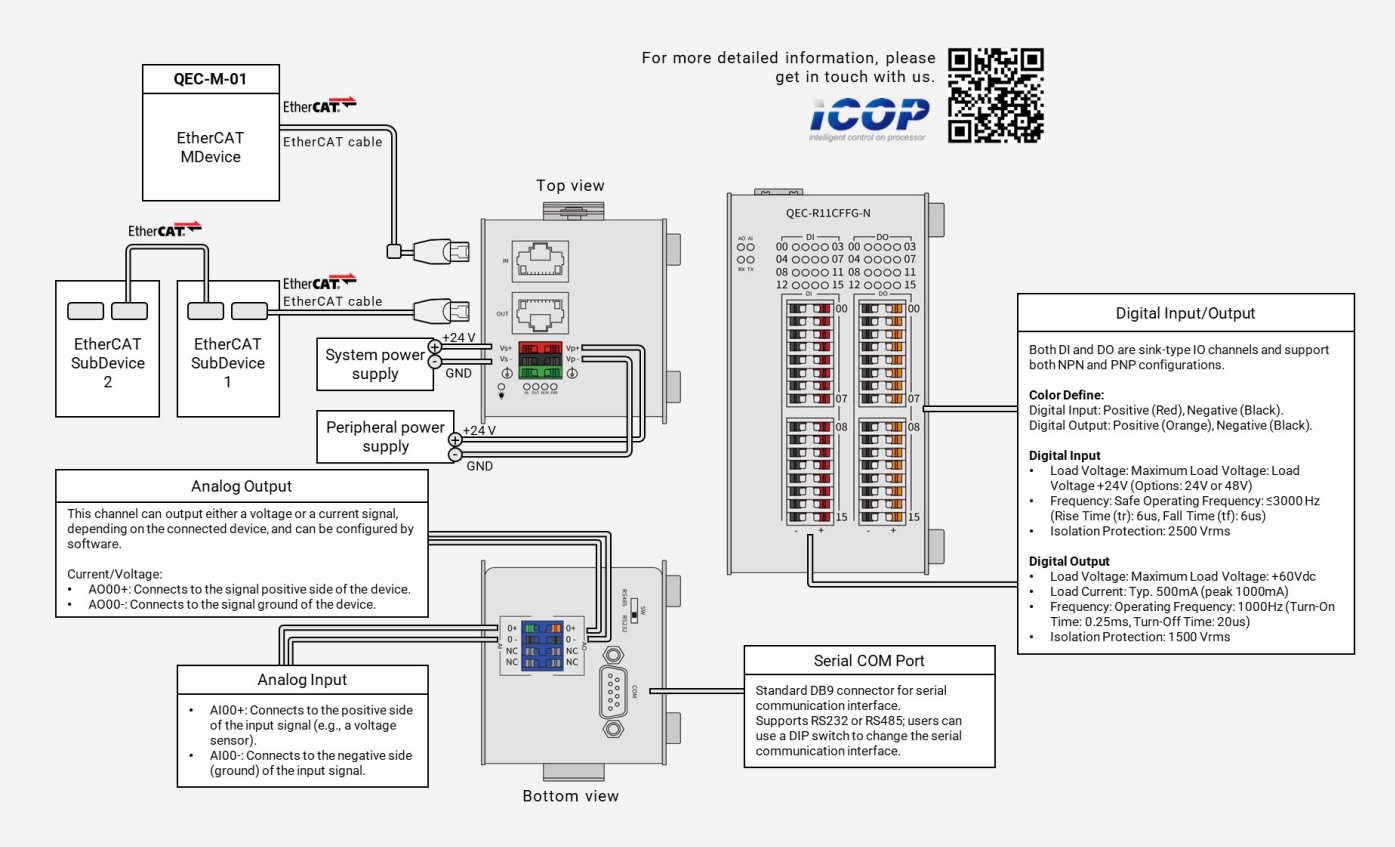
1. 電源：EU 端子を介して Vs+/Vs- および Vp+/Vp- に 24V 電源を接続します。
2. EtherCAT 接続：EtherCAT Out ポート（上側）から RJ45 ケーブルで EtherCAT Sub デバイスの EtherCAT In ポートに接続します。



1.2 QEC-R11CFFG

QEC-R11CFFG は、絶縁型 32 チャンネルのデジタル I/O（DI16/DO16）、アナログ I/O スロット（構成に応じて電圧／電流）、シリアル COM ポート（RS-232/RS-485 切り替え可能）を 1 台にまとめた EtherCAT Sub デバイスです。

下図は、QEC M デバイスおよび EtherCAT ネットワークとの代表的な配線例です。



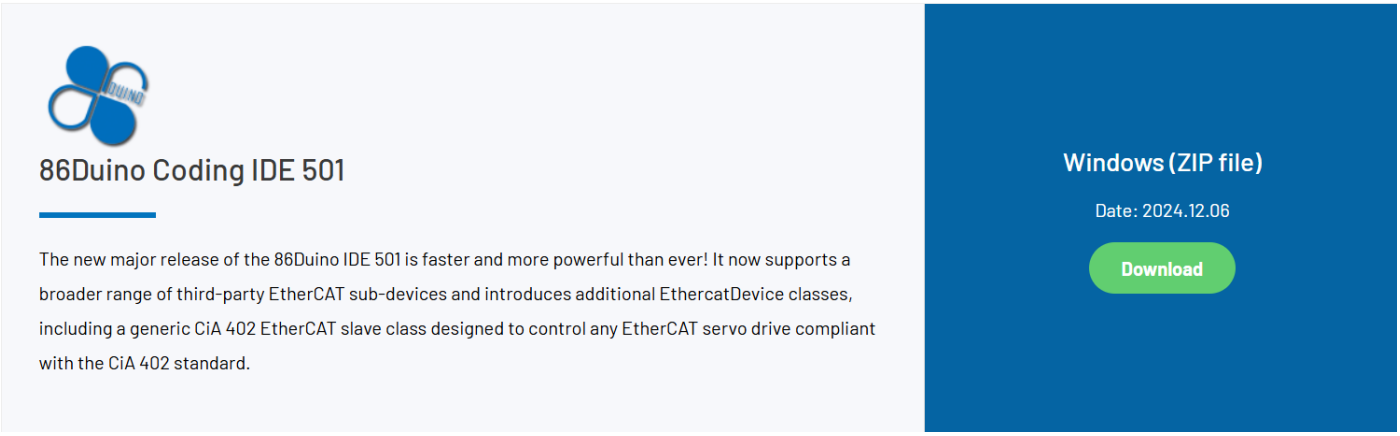
接続は機能ごとに次のように分かれています。

- EtherCAT：M デバイス → IN、OUT はデイジーチェーン接続用
- 電源・接地：
  - VS+/VS−：システム電源 +24 V/GND
  - VP+/VP−：フィールド I/O 電源 +24 V/GND
- デジタル I/O：
  - DI 16 点（シンクタイプ）：NPN/PNP 対応、安全動作周波数 ≥ 3 kHz、tr/tf 6 µs、入力 +24 V（オプションで 24/48 V）
  - DO 16 点：最大 +60 Vdc、標準 500 mA（ピーク 1 A）、1 kHz（Ton 0.25 ms / Toff 20 µs）。誘導性負荷にはフライバックダイオード／スナバ回路を追加してください。
  - 端子グループ：DI00～07 / DI08～15、DO00～07 / DO08～15（それぞれ + / − 端子付き）
  - 絶縁：DI 2500 Vrms、DO 1500 Vrms
- アナログ I/O：
  - AO 電圧／電流（ソフトウェアで切り替え可能）：AO00+ を機器のプラス側に、AO00− をリターン側に接続
  - AI：AI00+ をセンサーのプラス側に、AI00− をセンサーのリターン側に接続

- シリアル COM：DB9、DIP スイッチで RS-232/RS-485 を切り替え
  RS-485 では終端抵抗／バイアス抵抗を使用し、基準電位（GND）を共通にしてください。
- インジケーター：PWR / RUN / ERR / L/A（後述）
- 配線色：DI は + 赤 / − 黒、DO は + オレンジ / − 黒

2. ソフトウェア／開発環境

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。

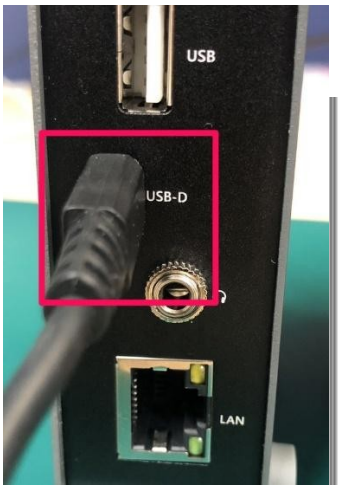
86Duino Coding IDE 501 以降の画面は次のとおりです。



3. PC との接続と環境設定

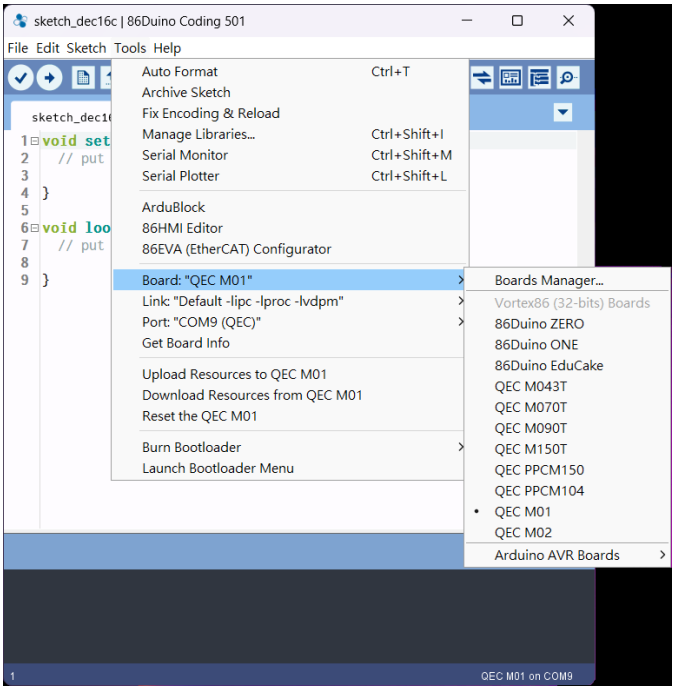
次の手順で環境を設定します。

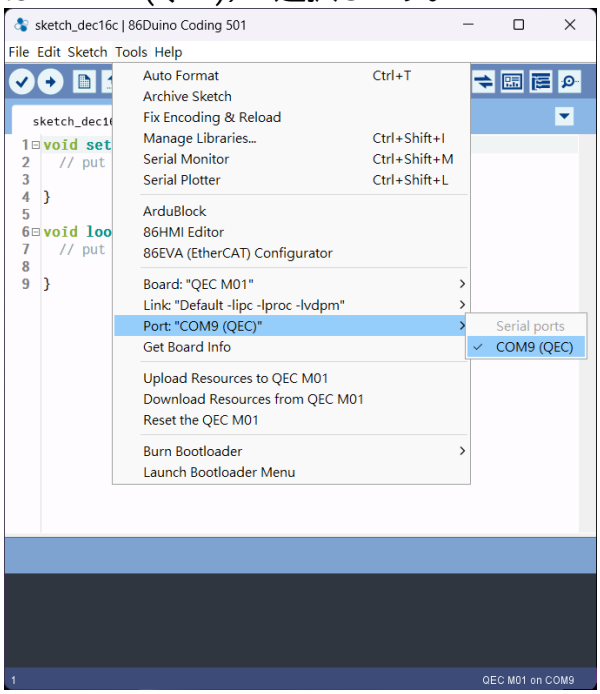
1. Micro USB－USB ケーブルで、QEC-M-01 を PC（86Duino IDE をインストール済み）に接続します。
2. QEC の電源を入れます。
3. PC で「**Device Manager（デバイス マネージャー）**」（Win+X キーを押して表示されるメニューから選択）を開き、「**Ports (COM & LPT)（ポート（COM と LPT））**」を展開します。「**Prolific PL2303GC USB Serial COM Port (COMx)**」が検出されていることを確認してください。表示されない場合は、必要なドライバーをインストールする必要があります。
   （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 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 デバイスが接続されている USB ポート（この例では COM9 (QEC)）を選択します。





4. 86EVA とコードの使用

この例では、86Duino IDE のグラフィカルなローコード開発ツール 86EVA を使用して、EtherCAT M デバイス（QEC-M-01）と QEC-R11CFFG（DIO・AIO・シリアル COM ポートを備えた EtherCAT 複合 I/O モジュール）を操作する方法を説明します。

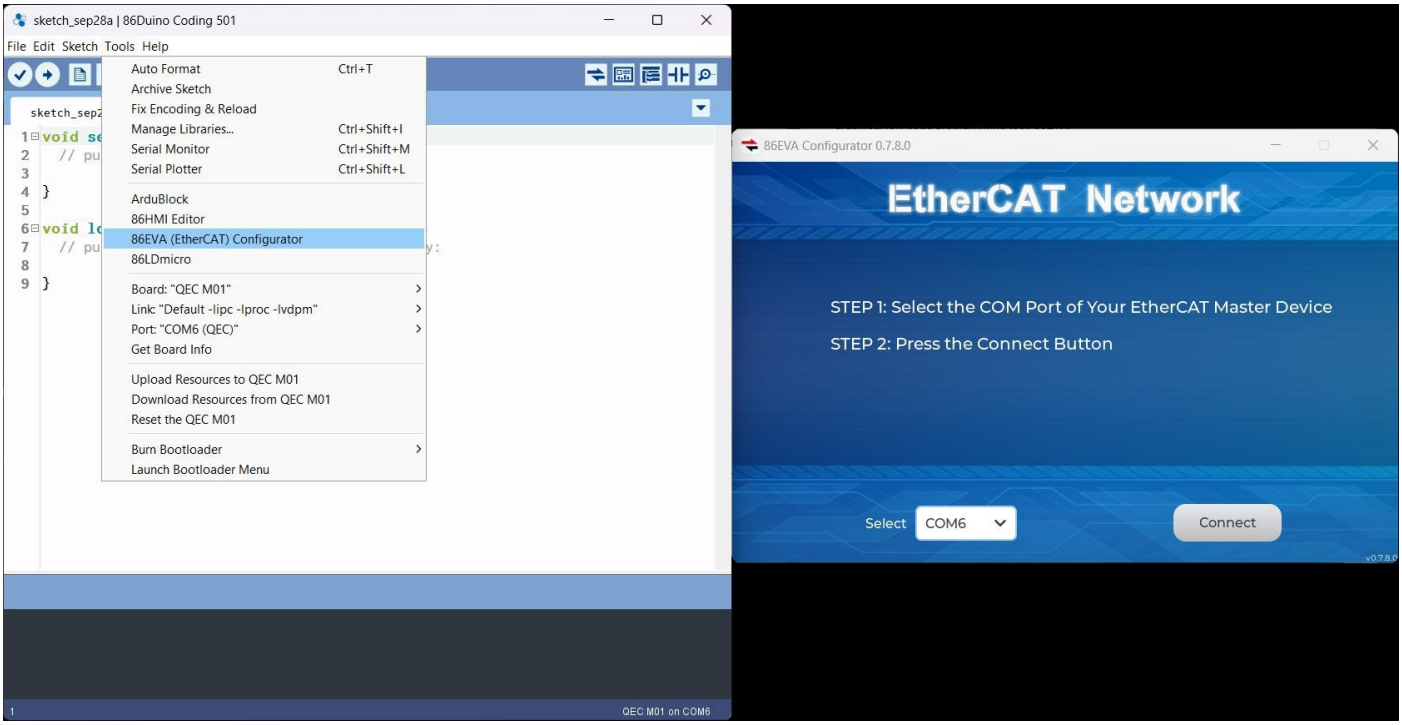
ソフトウェアツールの説明：

- 86EVA（EVA、EtherCAT-Based Virtual Arduino）：
  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 の開発キットの 1 つです。

以下では、デジタル I/O、アナログ I/O、シリアル COM ポート（RS232/RS485）の使い方を順を追って説明します。

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。



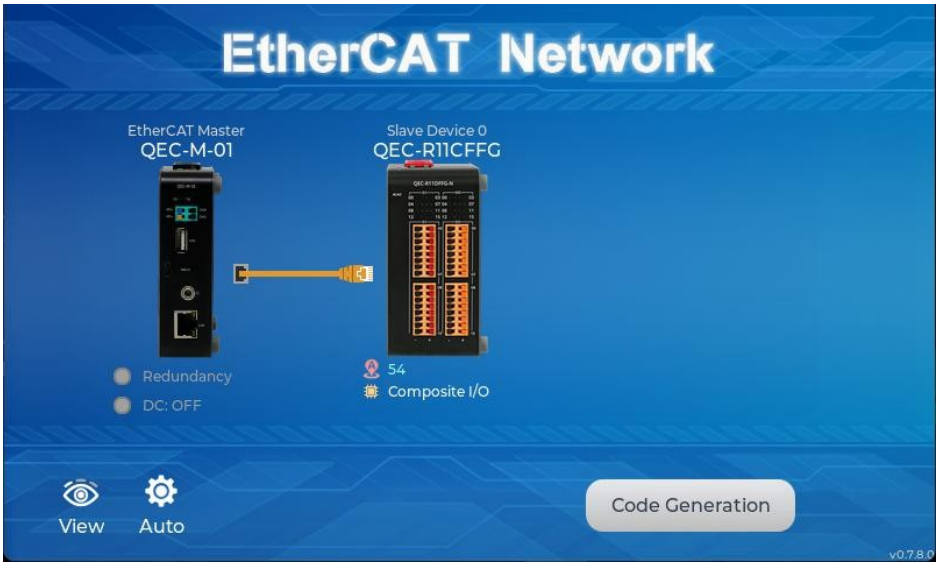
正しい COM ポートを選択し、「Connect（接続）」ボタンをクリックしてください。



QEC-M-01 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。



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。

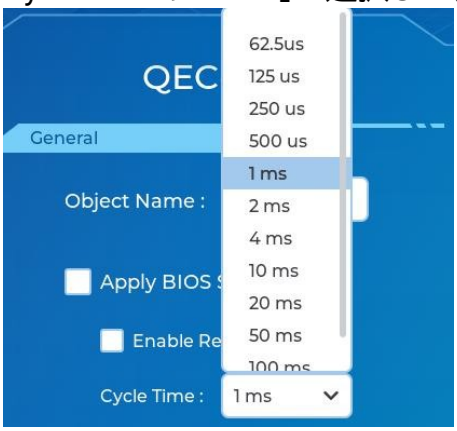
ステップ 2.1：QEC-M-01

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。



次の設定を確認してください。

1. 「**Apply BIOS Settings**」をオフにします。
2. Cycle Time に「**1ms**」を選択します。

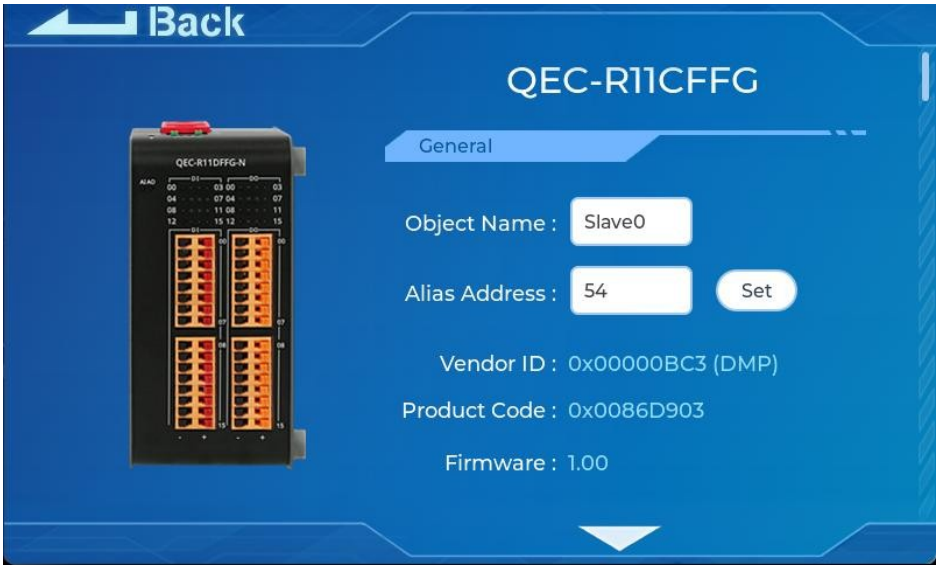


左上の「**Back（戻る）**」をクリックして元の画面に戻ります。



ステップ 2.2：QEC-R11CFFG

QEC-R11CFFG 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。



この画面には、Object Name、Alias Address、Vendor ID、Product Code、Virtual Arduino Mapping、Virtual Servo Configuration の各パラメーターが表示されます。

Object Name を「**slave0**」に変更してください。
Object Name をクリックすると、キーボードが表示されます。

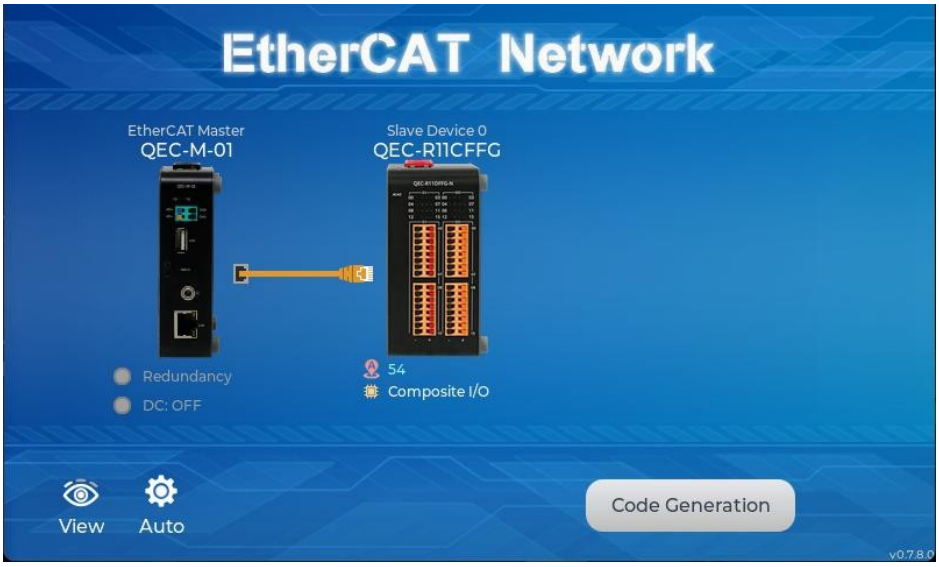


左上の「**Back（戻る）**」をクリックして元の画面に戻ります。

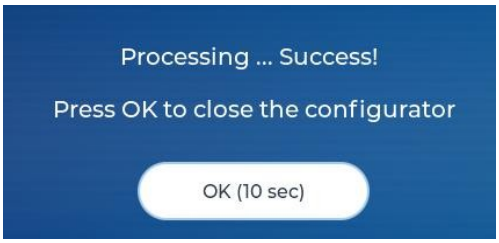


ステップ 3：コードを生成する

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。

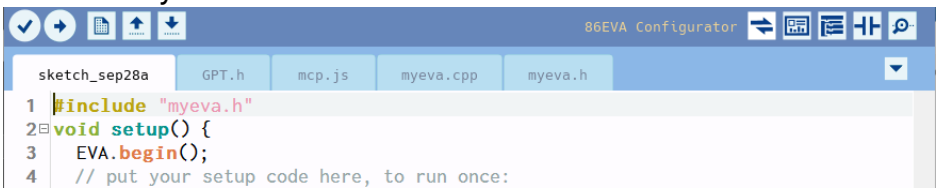


生成が完了したら OK ボタンをダブルクリックして 86EVA を閉じます。操作しない場合も 10 秒後に自動的に閉じます。



生成されるコードとファイルは次のとおりです。

- sketch_sep10b：メインプロジェクト（.ino、ファイル名はプロジェクト名により異なります）
- myeva.cpp：86EVA の C++ プログラムコード
- 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

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。86EVA はバックグラウンドで EtherCAT ステートマシンを自動的に処理します。

86EVA が生成するプログラムコードは、デフォルトで次のように設定されています。

- QEC-R11CFFG モジュール：EthercatDevice_QECR11CFFG オブジェクト
- EtherCAT モード：ECAT_SYNC

ユーザーが設定した項目は次のとおりです。

- EtherCAT サイクルタイム：1 ミリ秒
- EthercatMaster オブジェクト（“EcatMaster”）は QEC-M-01 を表し、EthercatDevice_QECR11CFFG オブジェクト（“slave0”）は QEC-R11CFFG モジュールを表します。

以下では、デジタル I/O、アナログ I/O、シリアル COM ポート（RS232/RS485）の使い方を順を追って説明します。

ステップ 4.1：デジタル I/O

このセクションでは、デジタル入力 DI00（例：ボタン）を読み取り、その状態をデジタル出力 DO00（例：LED／負荷）に反映させます。

A. setup 関数：

`setup()` 関数では、通信を初期化し、EtherCAT ネットワークを OP モードまで立ち上げます。次の手順で行います。

1. シリアル通信の初期化
    - ボーレート 115200 でシリアル通信を開始します。
2. 86EVA の開始
    - `EVA.begin()` 関数を使用して、EtherCAT ネットワークを開始・初期化します。

B. loop 関数：

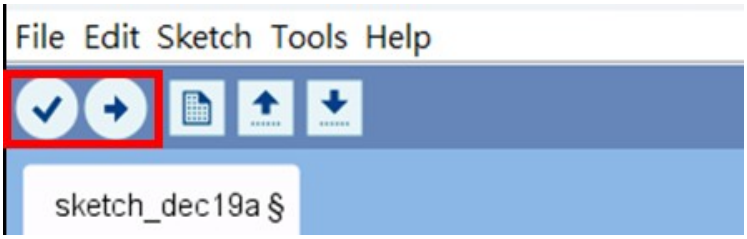
`loop()` 関数では、DI00 を読み取り、その状態を DO00 に継続的に反映させます。

1. ロジック
    - `digitalRead(0)` が HIGH の場合は `digitalWrite(0, HIGH);` を、それ以外の場合は `digitalWrite(0, LOW)` を実行します。
2. コードロジックのまとめ
    - `digitalRead(n)` で DI チャンネル n（0～15）を読み取ります。
    - `digitalWrite(n, HIGH/LOW)` で DO チャンネル n（0～15）を駆動します。

サンプルコードは次のとおりです。

```
#include "myeva.h"

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

void loop()
{
  if (slave0.digitalRead(0) == HIGH)
  {
    slave0.digitalWrite(0, HIGH);
  }
  else
  {
    slave0.digitalWrite(0, LOW);
  }
}
```

※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。

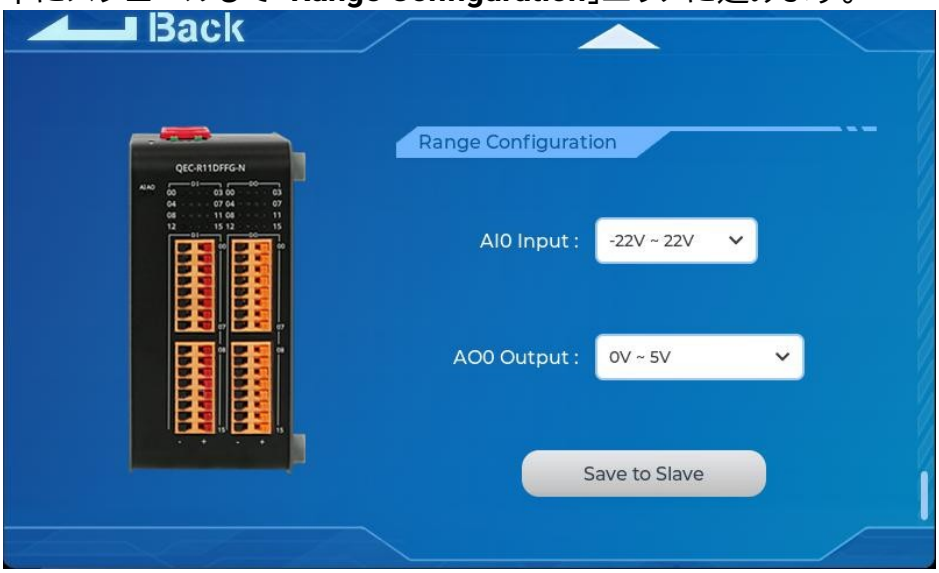


ステップ 4.2：アナログ I/O

このセクションでは、アナログ入力 AI00（電圧）を読み取り、アナログ出力 AO00 を 1 秒ごとに 1 V ずつ 0 V → 5 V の範囲で変化させます。

プログラムを作成する前に、86EVA でアナログ入力と出力のモードを設定する必要があります。
これは ステップ 2.2：QEC-R11CFFG の応用設定です。

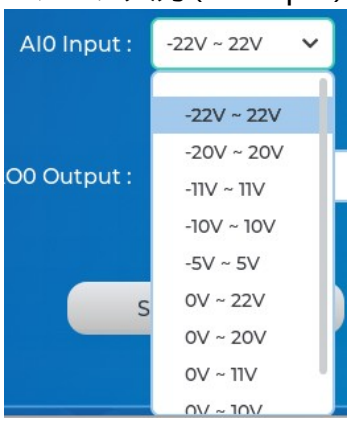
下にスクロールして「**Range Configuration**」エリアに進みます。



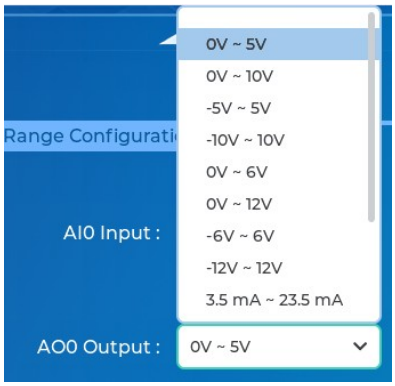
アナログ入力とアナログ出力のレンジは、それぞれ選択できます。
本ガイドでは、AI0 Input のアナログ入力レンジを「**-22V ~ 22V**」（デフォルト）に、AO0 Output のアナログ出力レンジを「**0V ~ 5V**」（デフォルト）に設定してください。

下図のようになります。

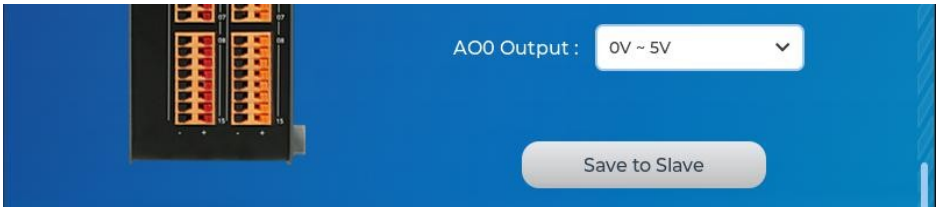
1. アナログ入力（AI0 Input）



2. アナログ出力（AO0 Output）



次に「**Save to Slave**」ボタンをクリックして、レンジ設定を QEC-R11CFFG に保存します。



Sub デバイスの設定が完了すると「**Success**」ウィンドウが表示され、新しい設定を反映するために Sub デバイスのリセットを求められます。



これでプログラムの作成を始められます。

A. setup 関数：

`setup()` 関数では、通信を初期化して OP モードに移行し、アナログのレンジを設定します。次の手順で行います。

1. シリアル通信の初期化
    - ボーレート 115200 でシリアル通信を開始します。
2. 86EVA の開始
    - `EVA.begin()` 関数を使用して、EtherCAT ネットワークを開始・初期化します。

B. loop 関数：

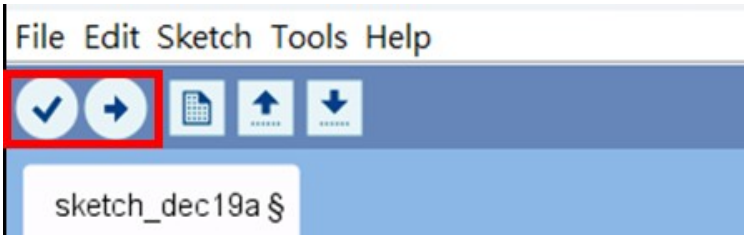
`loop()` 関数では、測定した AI 電圧を表示し、AO 電圧を段階的に変化させます。

1. ロジック
    - `voltageRead(0)` で AI00 を読み取り、値を表示します。
    - `voltageWrite(0, i)` で AO00 に出力します。i は 0 → 5 → 0（単位：V）の範囲で 1 秒ごとに更新されます。
2. コードロジックのまとめ
    - `voltageRead(ch)` でチャンネル ch のアナログ入力電圧を取得します。
    - `voltageWrite(ch, volts)` でチャンネル ch のアナログ出力電圧を設定します。

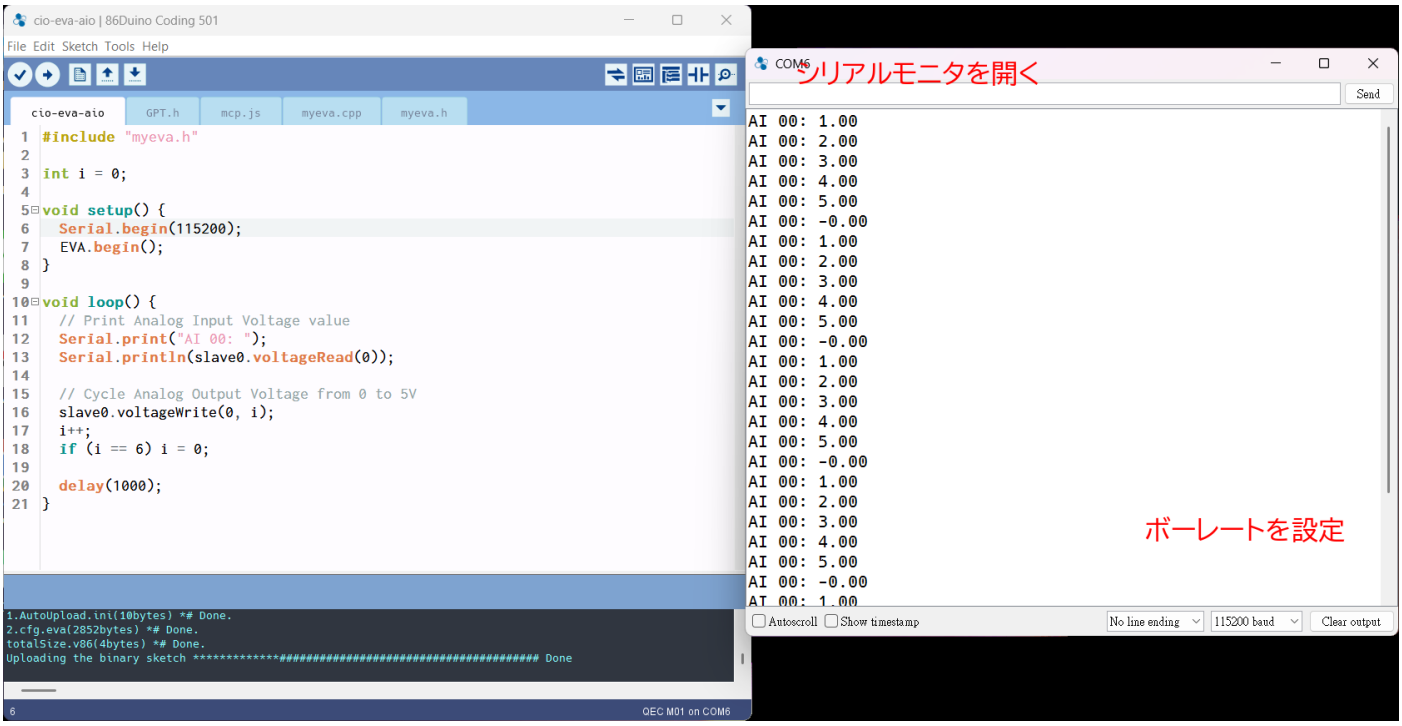
サンプルコードは次のとおりです。

```
#include "myeva.h"

int i = 0;

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

void loop() {
  // アナログ入力電圧の値を表示
  Serial.print("AI 00: ");
  Serial.println(slave0.voltageRead(0));

  // アナログ出力電圧を 0～5V で繰り返し変化
  slave0.voltageWrite(0, i);
  i++;
  if (i == 6) i = 0;

  delay(1000);
}
```

※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。



QEC-M-01 へのプログラムの書き込みが完了したら、86Duino IDE でシリアルモニタを開きます。シリアルのボーレートがプログラムの設定と同じであることを確認してください。



シリアルモニタにアナログ入力 AI00 の値（電圧）が表示されます。

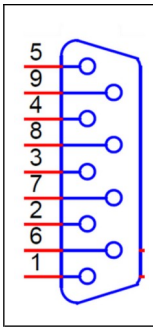
```
AI 00: -0.00
AI 00: -0.00
AI 00: 1.00
AI 00: 2.00
AI 00: 3.00
AI 00: 4.00
AI 00: 5.00
AI 00: -0.00
AI 00: 1.00
```

ステップ 4.3：シリアル COM ポート

このセクションでは、USB シリアルモニタから受け取ったバイトを QEC-R11CFFG の COM1（RS-232）に転送します。このテストでは RS-232 からデータが返ってくることは想定していないため、コードは送信のみを行い、受信したデータがあればそれを表示します。

配線

- 底面の DB9 コネクタを使用します（ピン配置は下図を参照）。
- 外部機器の TX ↔ RX、RX ↔ TX、GND ↔ GND をそれぞれ接続します。
- ケーブルは適切な長さにし、機器と基準電位（GND）を共通にしてください。



No.	ピン割り当て	No.	ピン割り当て
1	RS485-	6	DSR
2	RS485+/RXD	7	RTS
3	TXD	8	CTS
4	DTR	9	VCC
5	GND	-	-

※注意：RS232 と RS485 は同時に使用できません。

A. setup 関数：

`setup()` 関数では、通信を初期化して OP モードに移行し、COM1 を設定します。次の手順で行います。

1. シリアル通信の初期化
   - ボーレート 115200 でシリアル通信を開始します。
2. 86EVA の開始
   - `EVA.begin()` 関数を使用して、EtherCAT ネットワークを開始・初期化します。
3. COM ポート（RS-232）の設定
   - `uartSetBaud(COM1, 115200)` 関数を使用して、COM1 のボーレートを 115200 に設定します。
   - `uartSetFormat(COM1, SERIAL_8N1)` 関数を使用して、COM1 を SERIAL_8N1 フォーマットに設定します。

B. loop 関数：

`loop()` 関数では、EtherCAT／COM の処理を継続しながら、シリアルモニタから受け取った USB シリアルのバイトを COM1 に転送します。

1. ロジック
   - 毎サイクル `update()` を呼び出して、EtherCAT／メールボックス上の UART キューを更新します。
   - シリアルモニタにバイトがある場合は、それを読み取り `uartWrite(COM1, byte)` で COM1 に送信します。

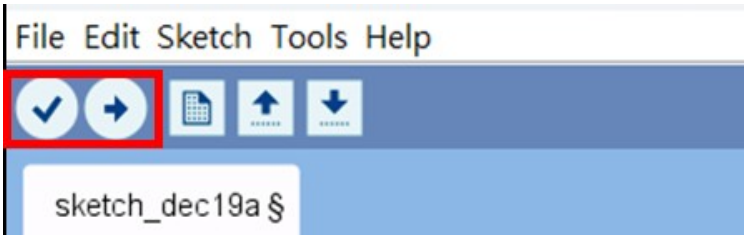
- このテストでは RS-232 からの応答はありませんが、コードはノンブロッキングの `uartRead(COM1)` を実行し、バイトがある場合のみ表示します。

2. コードロジックのまとめ
   - `Serial.available()` で RX FIFO を確認し、`Serial.read()` でシリアルからデータを取得します。
   - `slave0.uartRead(COM1)` / `uartWrite(COM1)` で、QEC-R11CFFG のシリアル COM ポートとバイトを送受信します。
   - EtherCAT メールボックスの処理を行うため、リモートシリアル I/O の処理は `loop()` 内に記述します。

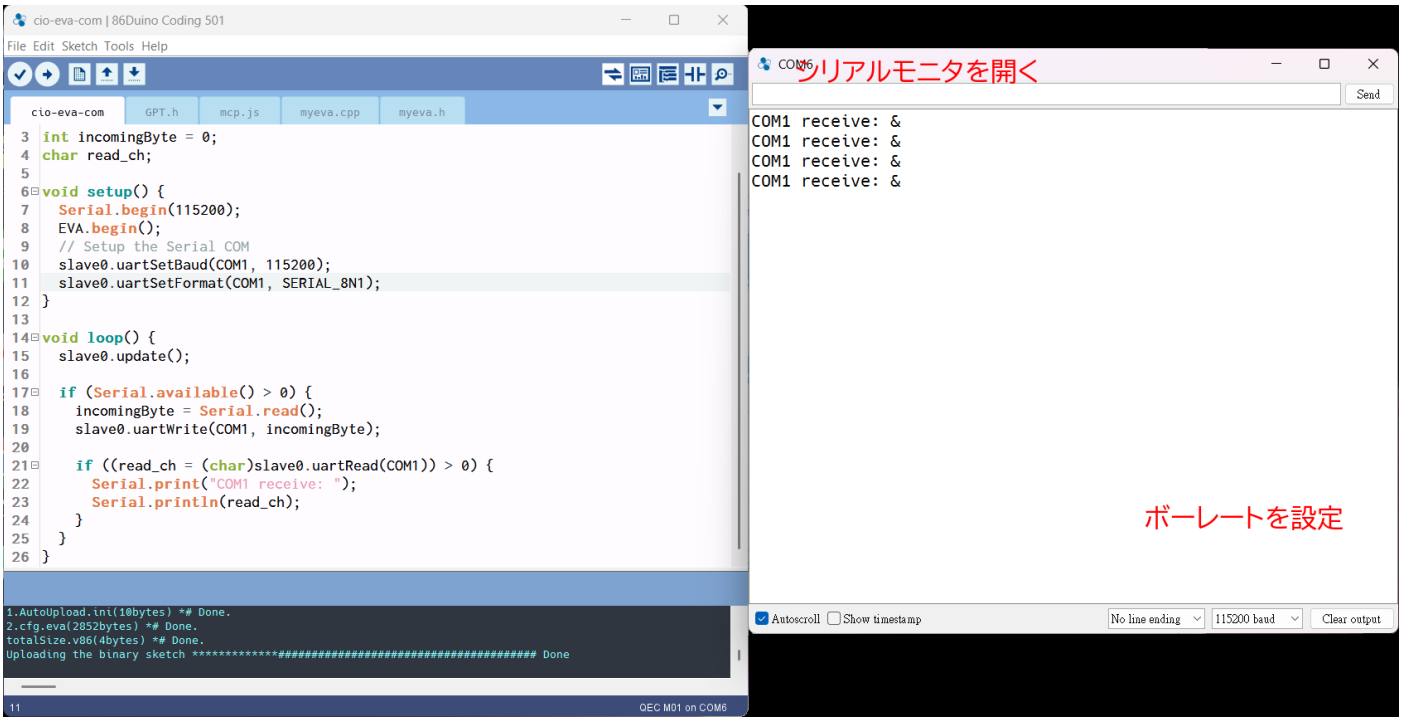
サンプルコードは次のとおりです。

```
#include "myeva.h"

int incomingByte = 0;
char read_ch;

void setup() {
  Serial.begin(115200);
  EVA.begin();
  // シリアル COM の設定
  slave0.uartSetBaud(COM1, 115200);
  slave0.uartSetFormat(COM1, SERIAL_8N1);
}

void loop() {
  slave0.update();

  if (Serial.available() > 0) {
    incomingByte = Serial.read();
    slave0.uartWrite(COM1, incomingByte);

    if ((read_ch = (char)slave0.uartRead(COM1)) > 0) {
      Serial.print("COM1 receive: ");
      Serial.println(read_ch);
    }
  }
}
```

※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。


File Edit Sketch Tools Help
sketch_dec19a §

QEC-M-01 へのプログラムの書き込みが完了したら、86Duino IDE でシリアルモニタを開きます。シリアルのボーレートがプログラムの設定と同じであることを確認してください。



QEC-R11CFFG の COM ポートがデータを受信すると、その値がシリアルモニタに表示されます。

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。

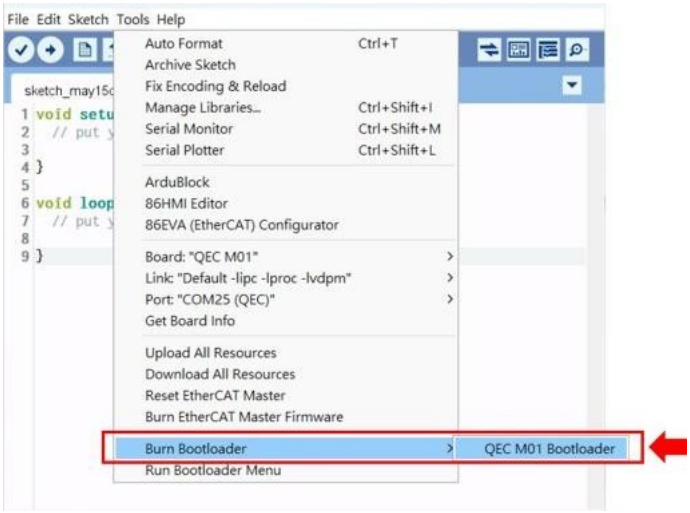
ステップ 1：QEC-M の準備

1. 86Duino IDE 500 以降（またはそれより新しいバージョン）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。

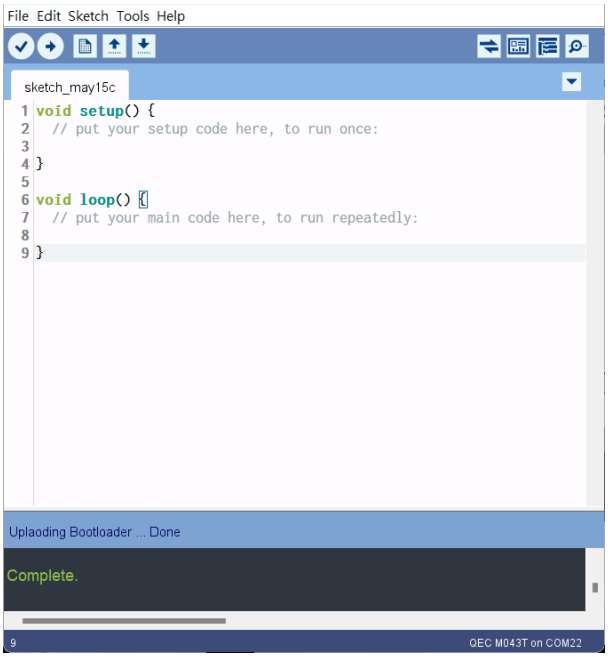
ステップ 2：「Burn Bootloader（ブートローダを書き込む）」をクリック

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。

保証

本製品は、ご購入日から 1 年間、正常に動作することを保証します。この期間中に本製品が正常に動作しなくなった場合は、以下の条件に定める場合を除き、当社の判断により無償で交換または修理いたします。本保証は、誤用、改造、事故または災害により損傷した製品には適用されません。販売元は、本製品の使用、誤用、または使用不能により生じたいかなる損害、逸失利益、逸失した節約、その他の付随的または結果的損害についても一切の責任を負いません。また、販売元はその他の関係者からのいかなる請求についても責任を負いません。返品をお受けするには、事前に販売元から返品承認を得る必要があります。返品承認は、販売元に電話または FAX で連絡し、返品承認番号（RMA 番号）を請求することで取得できます。返品の際は、必ず問題の内容を明記した説明を添付してください。

本書に記載されているすべての商標は、それぞれの所有者の登録商標です。すべての仕様は予告なく変更されることがあります。
© ICOP Technology Inc. 2025

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。