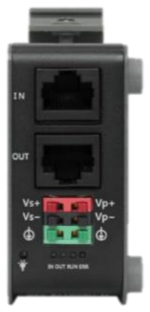
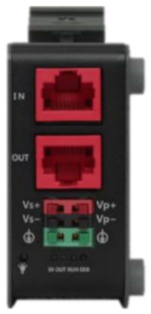
QEC Start Guide 17: Using EtherCAT to Transmit Modbus RTU control IO (2)

In this guide, we will demonstrate how to transmit Modbus RTU using the QEC-M-01 and the QEC-RXXHU series.

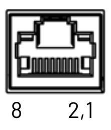
Notes QEC's PoE (Power over Ethernet)

In QEC product installations, users can easily distinguish between PoE and non-PoE: if the RJ45 house is red, it is PoE type, and if the RJ45 house is black, it is non-PoE type.

	
Non-PoE type	PoE type

PoE (Power over Ethernet) is a function that delivers power over the network. QEC can be equipped with an optional PoE function to reduce cabling. In practice, PoE is selected based on system equipment, so please pay attention to the following points while evaluating and testing:

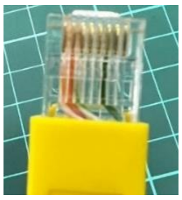
1. The PoE function of QEC is different and incompatible with EtherCAT P, and the PoE function of QEC is based on PoE Type B, and the pin functions are as follows:

	Pin #	Signal Name	Pin #	Signal Name
	1	LAN1_TX+	2	LAN1_TX-
	3	LAN1_RX+	4	VS+
	5	VP+	6	LAN1_RX-
	7	VS-(GND)	8	VP-(GND)

* PoE LAN with the Red Housing; Regular LAN with Black Housing.

* L4, L5, L7, L8 pins are option, for RJ45 Power IN/OUT.

2. When connecting PoE and non-PoE devices, make sure to disconnect Ethernet cables at pins 4, 5, 7, and 8 (e.g., when a PoE-supported QEC EtherCAT master connects with a third-party EtherCAT slave).

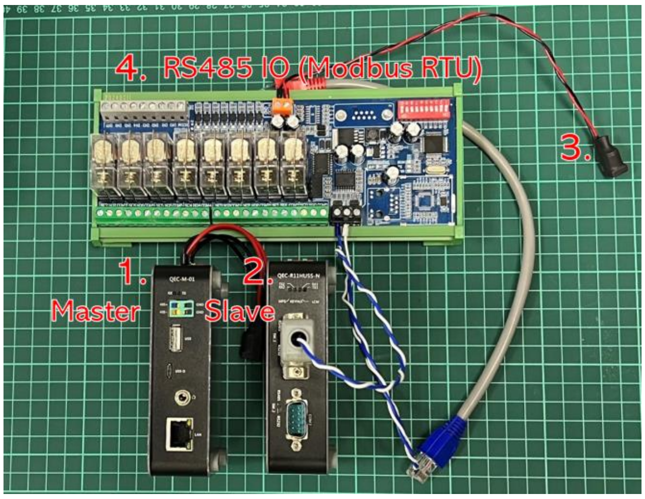


3. QEC's PoE power supply is up to 24V/3A.

Connection and wiring hardware

The following devices are used here:

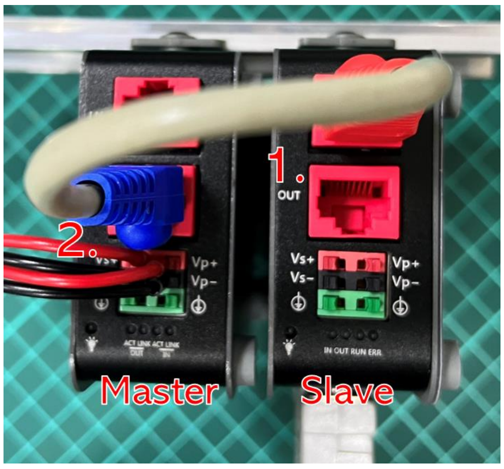
1. QEC-M-01P（EtherCAT Master/PoE）
2. QEC-R11HU5S-N (EtherCAT HID Slave/PoE, support 2 UART)
3. 24V power supply & EU-type terminal cable & RJ45 Cable
4. RS-485 communication IO Module (protocol: Modbus RTU)



QEC-M-01P

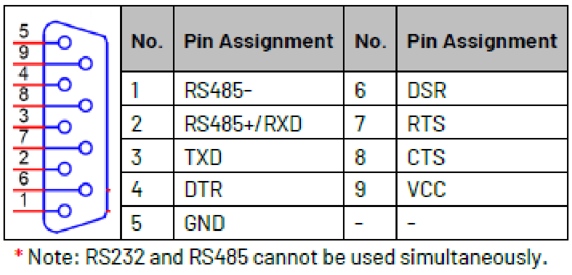
QEC EtherCAT master with PoE function.

1. Using the EtherCAT Out port (top side) connected to the EtherCAT In port of QEC-R11HU9S via RJ45 cable (powered by PoE).
2. Connect to Vs+/Vs- and Vp+/Vp- power supplies via EU terminals for 24V power.



QEC-R11HU5S-N

- The COM Port signal RS-485- on the QEC-R11HU5S-N is pin 1, and RS-485+ is pin 2.

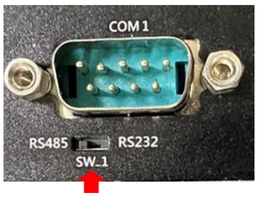


No.	Pin Assignment	No.	Pin Assignment
1	RS485-	6	DSR
2	RS485+/RXD	7	RTS
3	TXD	8	CTS
4	DTR	9	VCC
5	GND	-	-

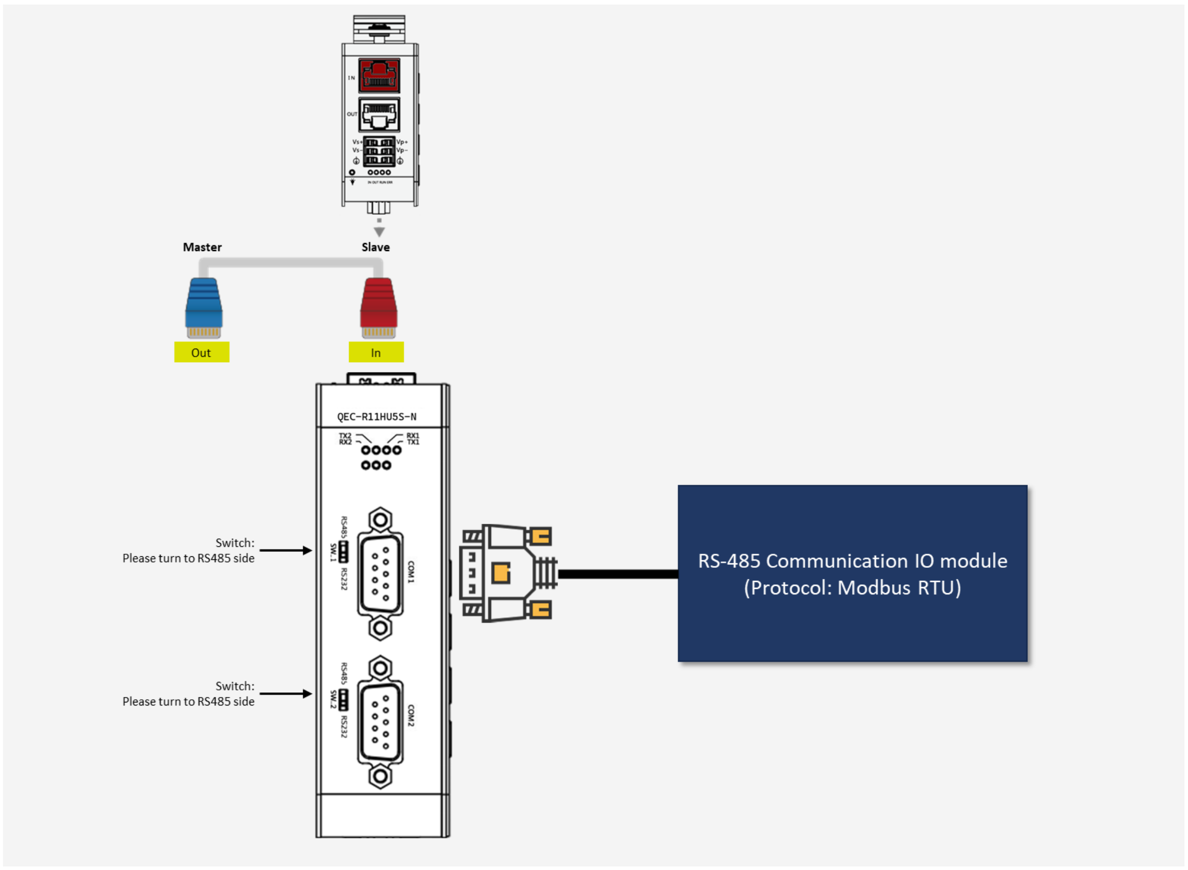
* Note: RS232 and RS485 cannot be used simultaneously.

- This example uses RS-485.

  Note: There is a switch next to each of the two UART Ports. This switch can be used to toggle between RS-232 and RS-485 functionalities. After adjusting the switch, please restart the device's power.

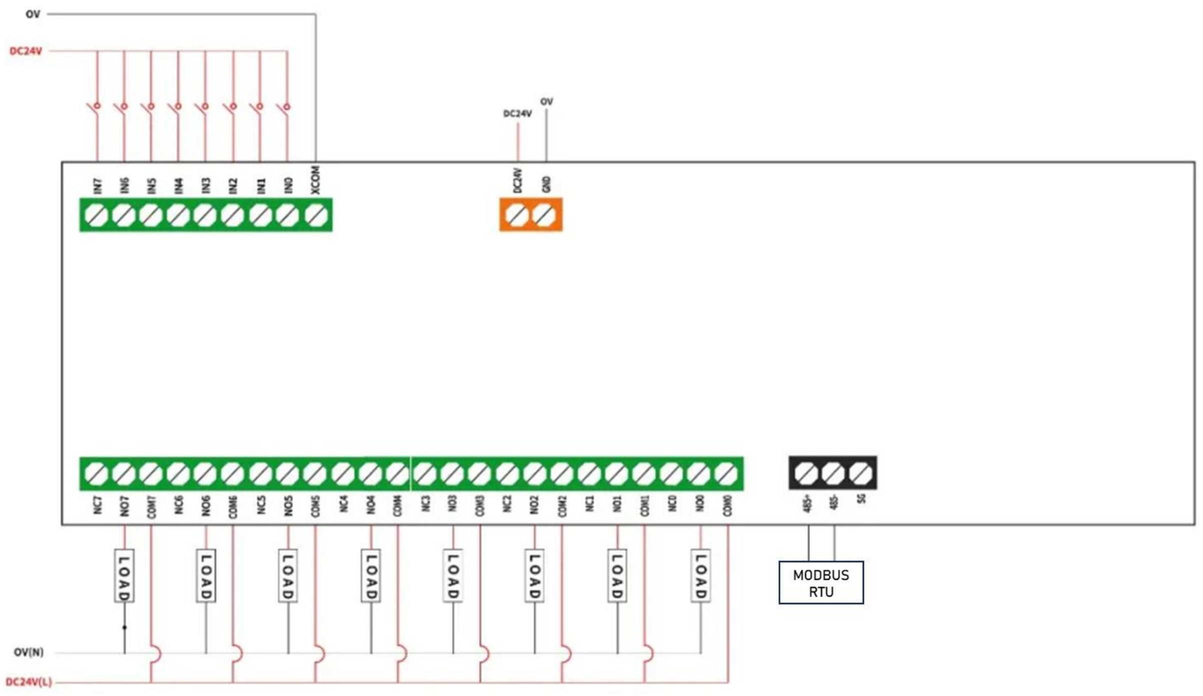


- Connect COM1 on the QEC-R11HU5S-N to the RS-485 communication IO module (communication protocol: Modbus RTU) using an RS-485 transmission line.


Master
Slave
Out
In
QEC-R11HU5S-N
Switch: Please turn to RS485 side
COM1
Switch: Please turn to RS485 side
COM2
RS-485 Communication IO module
(Protocol: Modbus RTU)

Note: The RS-485 supported by the QEC-RXXHU series requires users to add a 120-ohm termination resistor according to the specific scenario to optimize signal quality. Termination resistors can match the impedance of RS-485, reducing signal reflection, which is especially important in long-distance or high-interference environments. When designing an RS-485 network, considering the network length and the number of devices to determine the need for termination resistors is key to ensuring communication reliability and efficiency.

The RS-485 communication IO module (communication protocol: Modbus RTU) used in this example is an RS-485 communication module from Huaqingjun Electronics Co., Ltd., with the following circuit configuration:

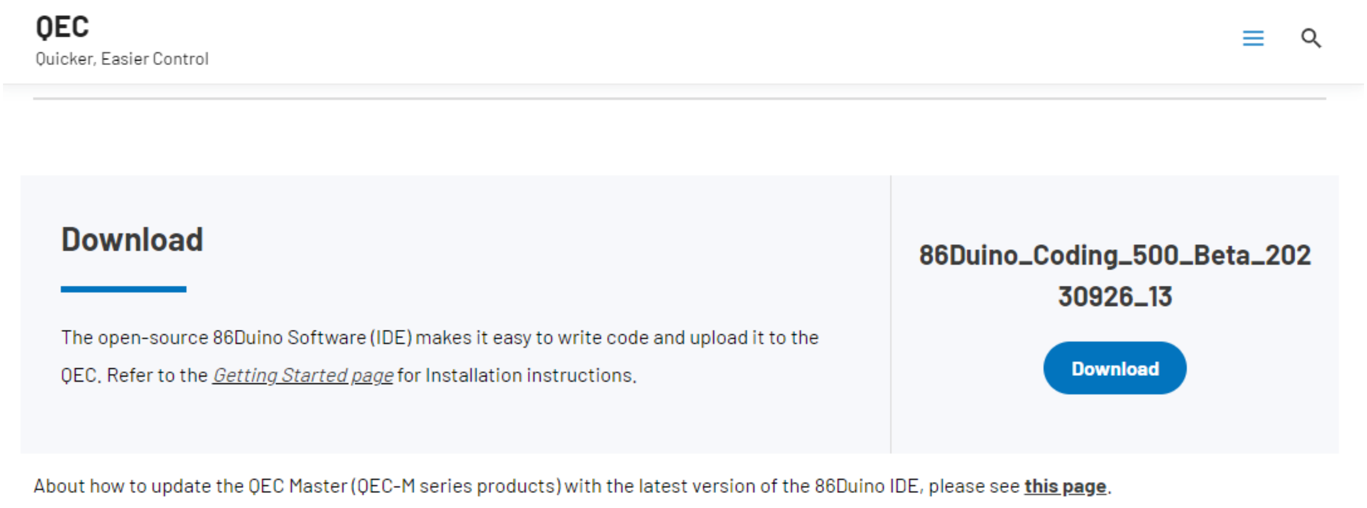


(Input compatible NPN and PNP)

If you have any requirements or questions, please contact the [Huaqingjun flagship store on Taobao].

Software/Development Environment: 86Duino IDE

Download 86duino IDE from https://www.qec.tw/software/.



QEC
Quicker, Easier Control

Download

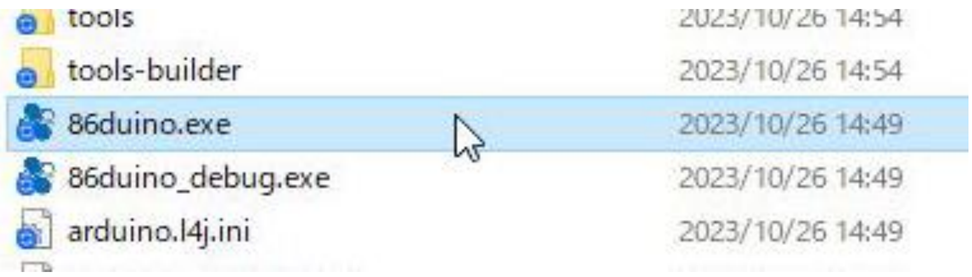
The open-source 86Duino Software (IDE) makes it easy to write code and upload it to the QEC. Refer to the *Getting Started page* for Installation instructions.

86Duino_Coding_500_Beta_20230926_13

Download

About how to update the QEC Master (QEC-M series products) with the latest version of the 86Duino IDE, please see **this page**.

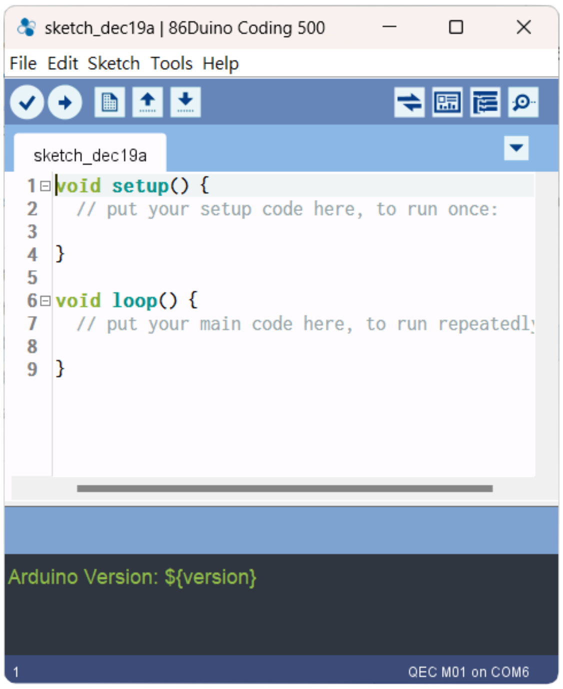
After downloading, please unzip the downloaded zip file, no additional software installation is required, just double-click 86duino.exe to start the IDE.



tools	2023/10/26 14:54
tools-builder	2023/10/26 14:54
86duino.exe	2023/10/26 14:49
86duino_debug.exe	2023/10/26 14:49
arduino.l4j.ini	2023/10/26 14:49

*Note: If Windows displays a warning, click Details once and then click the Continue Run button once.

86Duino Coding IDE 500+ looks like below.



sketch_dec19a | 86Duino Coding 500

File Edit Sketch Tools Help

sketch_dec19a

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedl
  
}
```

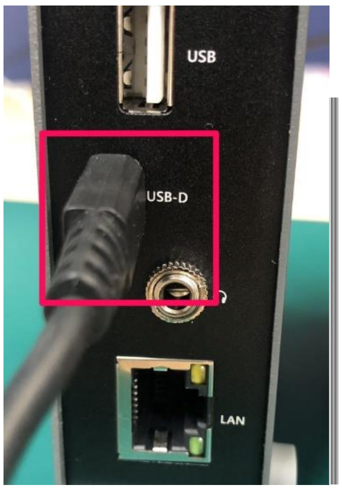
Arduino Version: ${version}

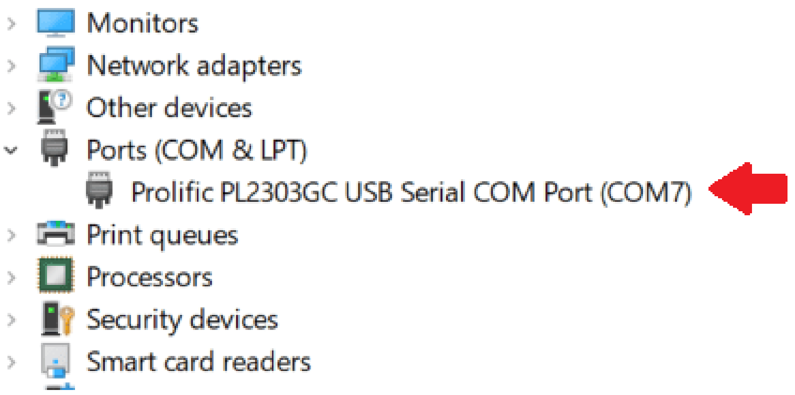
1 QEC M01 on COM6

Connect to your PC and set up the environment

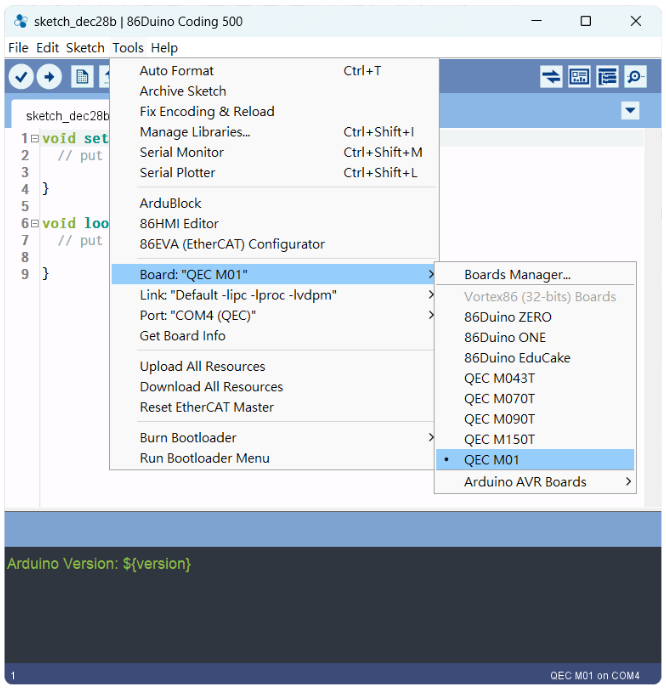
Follow the steps below to set up the environment:

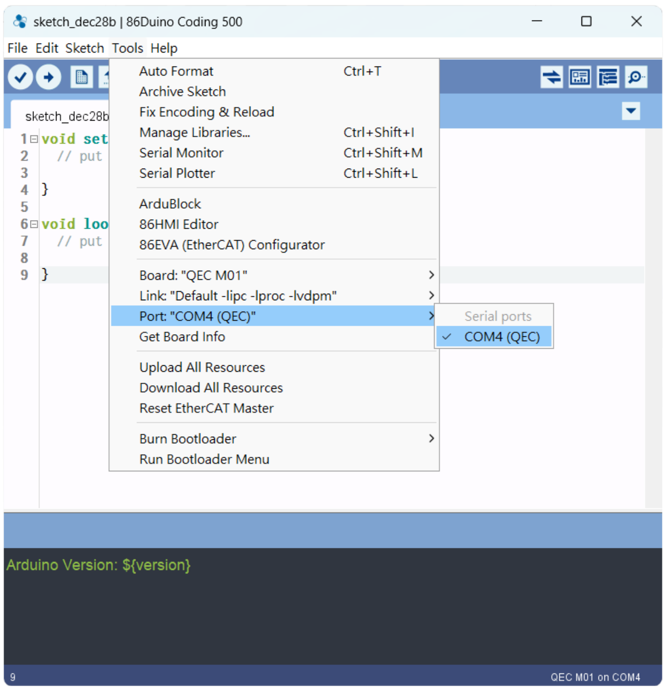
1. Connect the QEC-M-01P to your PC via a Micro USB to USB cable (86Duino IDE installed).
2. Turn on the QEC power.
3. Open “Device Manager” (select in the menu after pressing Win+X) ->” Ports (COM & LPT)” in your PC and expand the ports; you should see that the “Prolific PL2303GC USB Serial COM Port (COMx)” is detected; if not, you will need to install the required drivers.
   (For Windows PL2303 driver, you can download here)





4. Open the 86Duino IDE.
5. Select the correct board: In the IDE's menu, select Tools> Board > QEC-M-01 (or the QEC-M master model you use).
6. Select Port: In the IDE's menu, select Tools > Port and select the USB port to connect to the QEC-M master (in this case, COM4 (QEC)).





Development Method 1: Write code

The EtherCAT master (QEC-M-01P) and the HID slave (QEC-R11HU9S-N) can be configured and programmed via the EtherCAT library in the 86Duino IDE. The Arduino development environment has two main parts: `setup()` and `loop()`, which correspond to initialization and main programs. Before operating the EtherCAT network, you must configure it once. The process should be from Pre-OP to OP mode in EtherCAT devices.

In the example below, we integrate EtherCAT and Modbus communications using the EtherCAT and Modbus libraries. First, we initialize an EtherCAT master station and an EtherCAT device (QEC-R11HU5S), and set up the EtherCAT network. Then, we configure a virtual serial port (COM1) and set its baud rate to 115200. In the main loop (`loop()`), we read the value of holding register address 4 from a specific slave device with ID 30 every second using Modbus commands, and display the value in binary format on the serial monitor. If the read operation fails, we can add error handling logic.

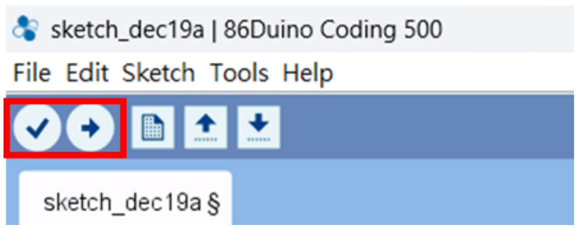
When using a QEC Slave, you can utilize the dedicated QEC EtherCAT Slave library. For instance, the QEC-R11HU5S can use the functionalities of the EthercatDevice_QECRXXHU Class; and if you want to use Modbus under EtherCAT, you will need to use the VirtualSerial feature. For detailed Modbus library description, please refer to Modbus Library.
An example of the code is provided below, note that VirtualSerial has been integrated into the EtherCAT library (EtherCAT.h).

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

// Device global instance area
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();
  // Check if one second has passed
  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 {
      // Handle error
    }
    lastTime = millis();   // Update the last time a write was attempted
  }
}
```

When the code is successfully uploaded, you can read the value of holding register address 4 from a specific slave device with ID 30 every second using Modbus commands, and display the value in binary format on the serial monitor. If the read operation fails, you can add error handling logic.

This setup can be used for automation control applications that require periodic data updates, such as remote monitoring systems. By reading the holding register value every second, we can continuously monitor the device's status, enabling real-time data updates and fault detection.

Note: Once the code is written, click on the toolbar to ✓ compile, and to confirm that the compilation is complete and error-free, you can click ➜ to upload. The program will run when the upload is complete.

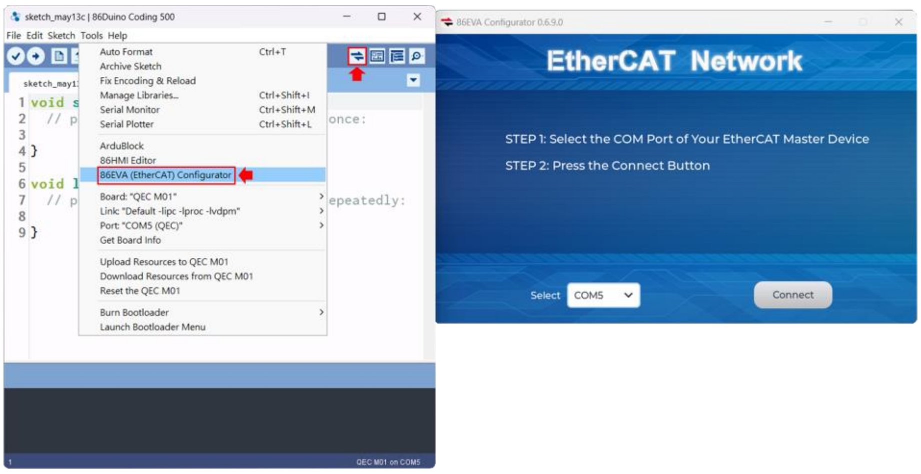

sketch_dec19a | 86Duino Coding 500
File Edit Sketch Tools Help
sketch_dec19a §

Development Method 2: Use 86EVA with code

86EVA is a graphical EtherCAT configuration tool based on the EtherCAT Library in the 86Duino IDE, part of the development kit of the 86Duino IDE. Users can configure the EtherCAT master (QEC-M-01P) and HID slave (QEC-R11HU9S-N) through 86EVA, and then start programming.

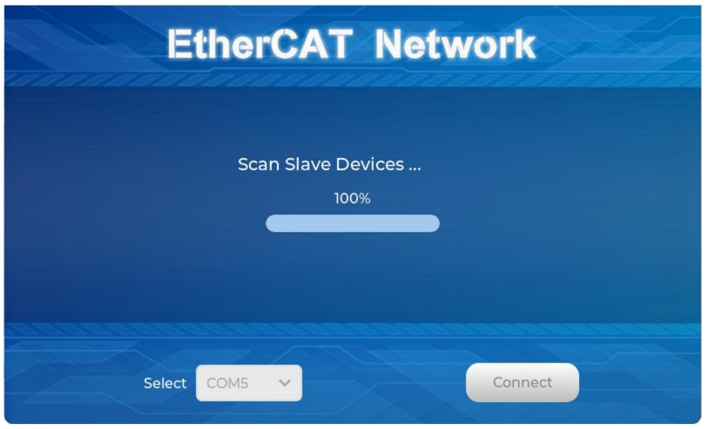
Step 1: Turn on 86EVA and scan

The 86EVA tool can be opened via the following buttons.

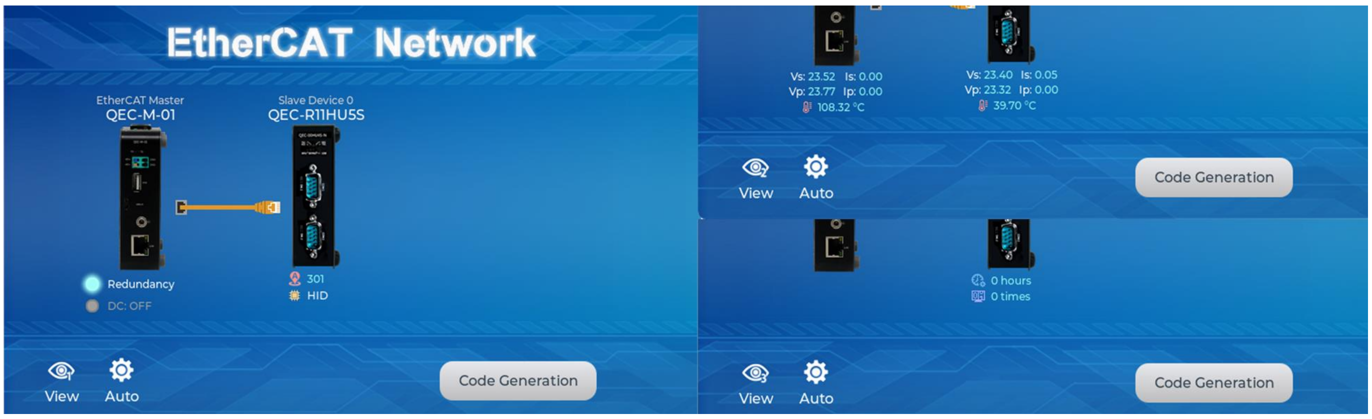

sketch_may13c | 86Duino Coding 500
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
ArduBlock
86HMI Editor
86EVA (EtherCAT) Configurator
Board: "QEC M01"
Link: "Default -lipc -lproc -lvdpm"
Port: "COM5 (QEC)"
Get Board Info
Upload Resources to QEC M01
Download Resources from QEC M01
Reset the QEC M01
Burn Bootloader
Launch Bootloader Menu
QEC M01 on COM5
86EVA Configurator 0.6.9.0
EtherCAT Network
Select
COM5
Connect

STEP 1: Select the COM Port of Your EtherCAT Master Device
STEP 2: Press the Connect Button

Once you have confirmed that the correct COM port has been selected of QEC-M-01P, press the Connect button to start scanning the EtherCAT network.

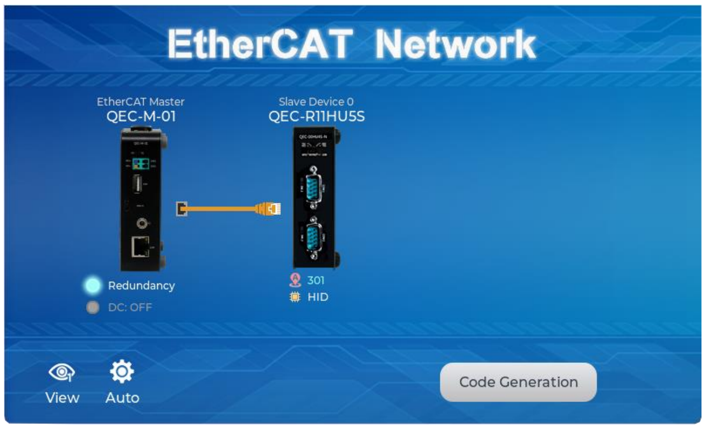


The connected devices will be displayed after the EtherCAT network has been scanned. Press the "View" button in the lower left corner to check the device's status (Voltage, Current, and Temperature; View2) and operating time (Hours; View3).



Step 2: Set the parameters

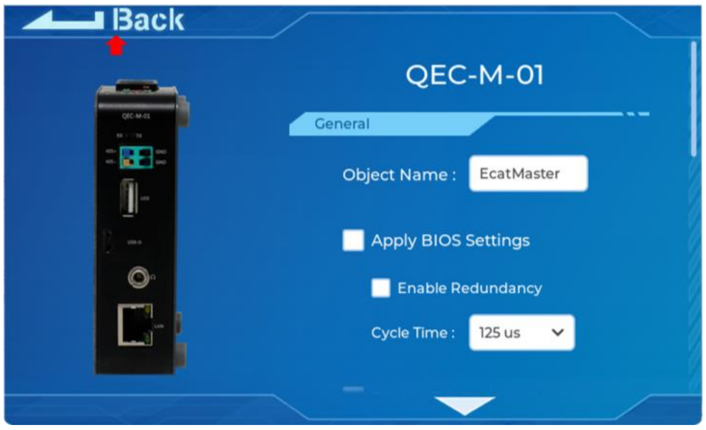
Press twice on the scanned device image to enter the corresponding parameter setting screen.



QEC-M-01

Press twice on the image of the QEC-M-01 to see the parameter settings.

This example will use the default settings and not change any settings; please click "Back" in the upper left corner to return.



Object Name : EcatMaster

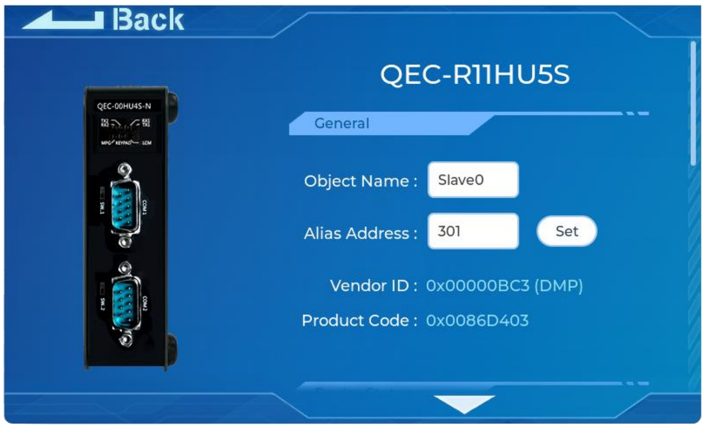
Apply BIOS Settings

Enable Redundancy

Cycle Time : 125 us

QEC-R11HU5S-N

Double-click on the image of the QEC-R11HU5S to view the parameter settings.



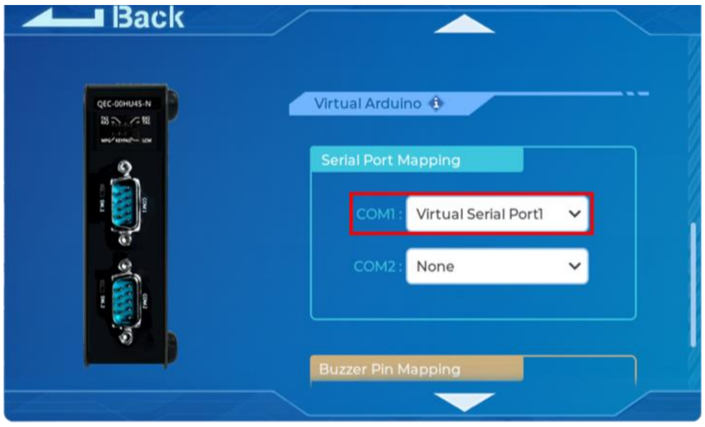
Object Name : Slave0

Alias Address : 301

Vendor ID : 0x00000BC3 (DMP)

Product Code : 0x0086D403

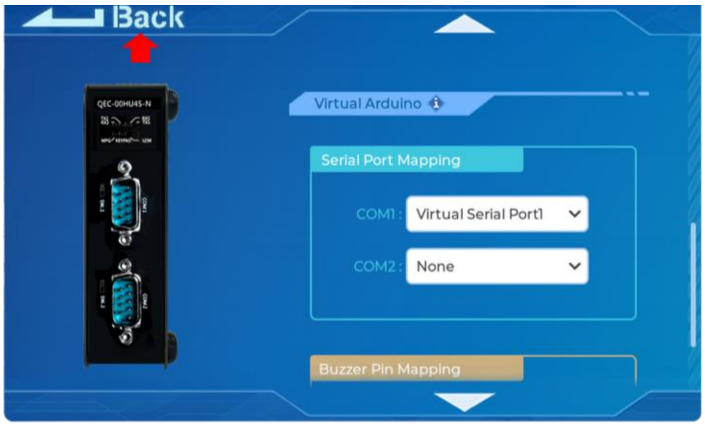
Continue scrolling down to the "Serial Port Mapping" section. Here, select "Virtual Serial Port1" from the dropdown menu for COM1 in the "Serial Port Mapping" area.



COM1 : Virtual Serial Port1

COM2 : None

Once completed, click 'Back' in the upper left corner to return.



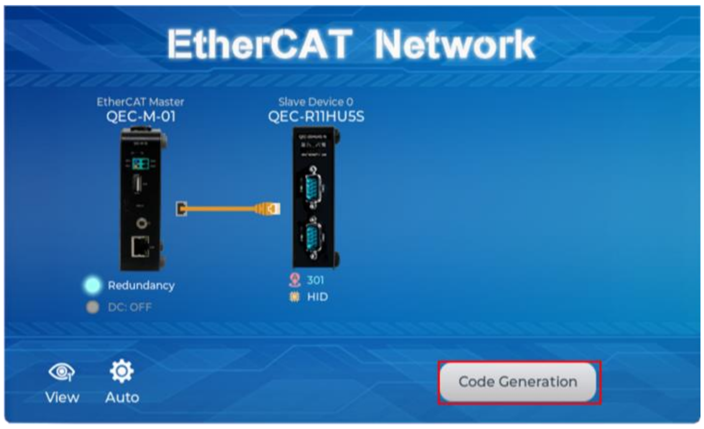
COM1 : Virtual Serial Port1

COM2 : None

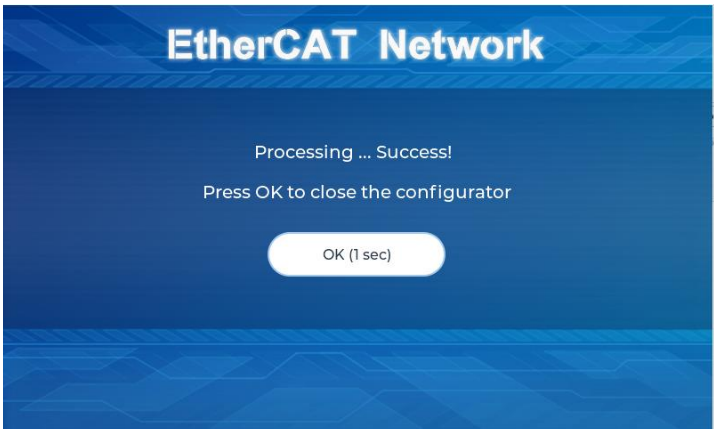
This action sets COM1 on the QEC-R11HU5S as the Virtual Serial Port1 in EVA.

Step 3: Generate the code

Once you've set your device's parameters, go back to the home screen and press the "Code Generation" button in the bottom right corner.

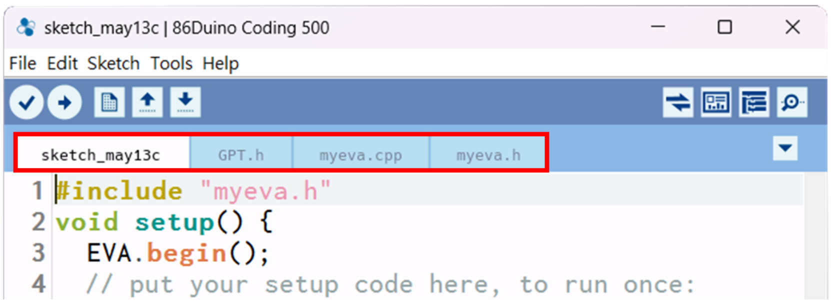


When you're done, double-click the OK button to turn off 86EVA, or it will close in 10 seconds.



The generated code and files are as follows:

- sketch_may13c: Main Project (.ino file, depending on your project name)
- GPT.h: Parameters for ChatGPT reference
- myeva.cpp: C++ code for 86EVA
- myeva.h: Header file for 86EVA


sketch_may13c | 86Duino Coding 500
File Edit Sketch Tools Help
sketch_may13c
GPT.h
myeva.cpp
myeva.h

```
#include "myeva.h"
void setup() {
  EVA.begin();
  // put your setup code here, to run once:
```

Additional note: After 86EVA generates code, the following code will be automatically generated in the main program (.ino), and any of them missing will cause 86EVA not to work.

1. `#include “myeva.h”` : Include EVA Header file
2. `EVA.begin()` in `setup();` : Initialize the EVA function

Step 4: Write the code

In the example below, we integrate EtherCAT and Modbus communications using the EtherCAT and Modbus libraries. First, we initialize an EtherCAT master station and an EtherCAT device (QEC-R11HU5S), and set up the EtherCAT network. Then, we configure a virtual serial port (COM1) and set its baud rate to 115200. In the main loop (`loop()`), we read the value of holding register address 4 from a specific slave device with ID 30 every second using Modbus commands, and display the value in binary format on the serial monitor. If the read operation fails, we can add error handling logic.

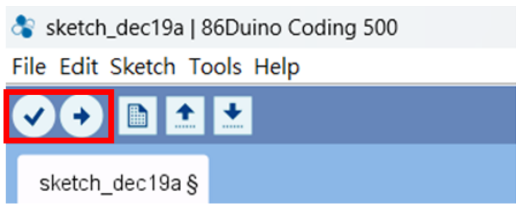
For detailed Modbus library description, please refer to Modbus Library.

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

ModbusMaster bus;

void setup() {
  Serial.begin(115200);
      EVA.begin();
  // put your setup code here, to run once:
  bus.begin(MODBUS_RTU, VirtualSerial1);
}

void loop() {
  static uint32_t lastTime = 0;
  uint32_t currentTime = millis();
  // Check if one second has passed
  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 {
      // Handle error
    }
    lastTime = millis();   // Update the last time a write was attempted
  }
}
```

When the code is successfully uploaded, you can read the value of holding register address 4 from a specific slave device with ID 30 every second using Modbus commands, and display the value in binary format on the serial monitor. If the read operation fails, you can add error handling logic.

This setup can be used for automation control applications that require periodic data updates, such as remote monitoring systems. By reading the holding register value every second, we can continuously monitor the device's status, enabling real-time data updates and fault detection.

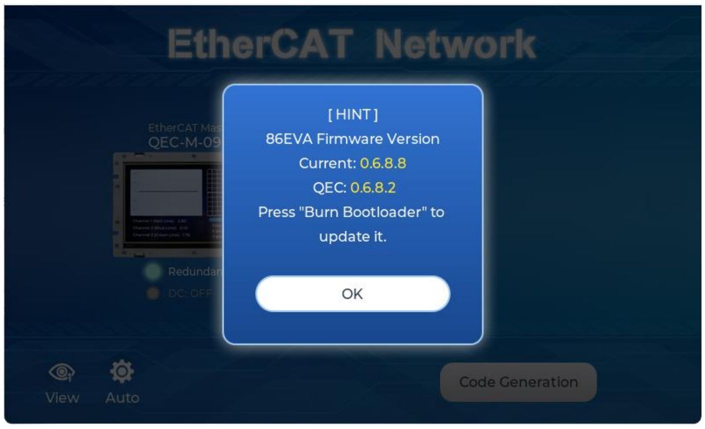
Note: Once the code is written, click on the toolbar to ✓ compile, and to confirm that the compilation is complete and error-free, you can click ➔ to upload. The program will run when the upload is complete.


sketch_dec19a | 86Duino Coding 500
File Edit Sketch Tools Help
sketch_dec19a §

Troubleshooting

QEC-M-01 cannot successfully upload code

When you are unable to successfully upload code, please open 86EVA to check if your QEC EtherCAT Master's environment is abnormal. As shown in the figure below, please try updating your QEC EtherCAT Master's environment, which will include the following three items: Bootloader, EtherCAT firmware, and EtherCAT tool.


[HINT]
86EVA Firmware Version
Current: 0.6.8.8
QEC: 0.6.8.2
Press "Burn Bootloader" to update it.

Now, we will further explain how to proceed with the update:

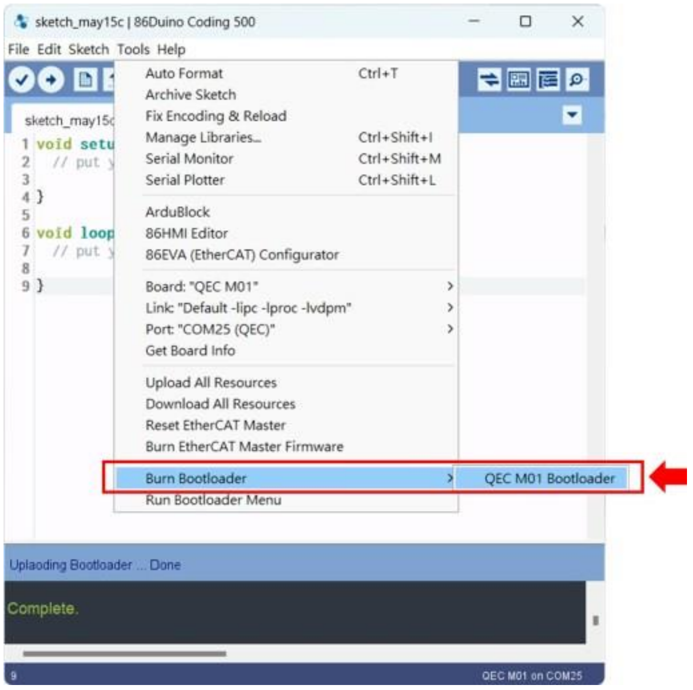
Step 1: Setting up QEC-M

1. Download and install 86Duino IDE 500 (or a newer version): You can download it from Software.
2. Connect the QEC-M: Use a USB cable to connect the QEC-M to your computer.
3. Open 86Duino IDE: After the installation is complete, open the 86Duino IDE software.
4. Select Board: From the IDE menu, choose “Tools” > “Board” > “QEC-M-01” (or the specific model of QEC-M you are using).
5. Select Port: From the IDE menu, choose “Tools” > “Port” and select the USB port to which the QEC-M is connected.

Step 2: Click "Burn Bootloader" button

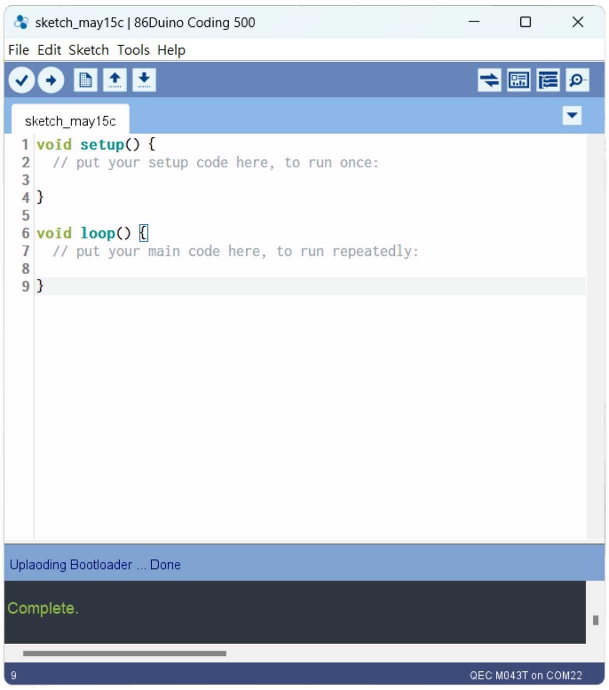
After connecting to your QEC-M product, go to "Tools"> "Burn Bootloader". The currently selected QEC-M name will appear. Clicking on it will start the update process, which will take approximately 5-20 minutes.

QEC-M-01:



```
Uplaoding Bootloader ... Done
Complete.
```

Step 3: Complete the Update


sketch_may15c | 86Duino Coding 500
File Edit Sketch Tools Help
sketch_may15c
QEC M043T on COM22

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
Uplaoding Bootloader ... Done
Complete.
```

After completing the above steps, your QEC-M has been successfully updated to the latest version of the development environment.

For more information and sample requests, please write to info@icop.com.tw, call your nearest ICOP branch, or contact our official global distributor.