Start Guide

QEC-M-01 Modbus TCP Control Digital I/O

86Duino Coding IDE 501

Ethernet Library & Modbus Library

(Version 1.0)

Revision

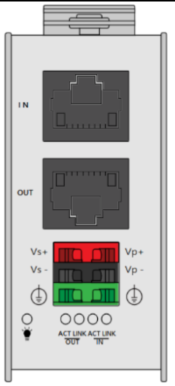
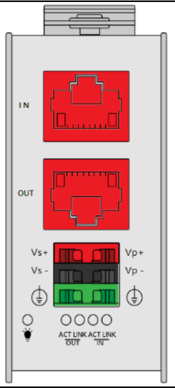
Date	Version	Description
2025/06/12	Version1.0	New Release.

Preface

In this guide, we will show you how to use the EtherCAT MDevice QEC-M-01 control Modbus TCP Digital I/O Modbus via Modbus Library.

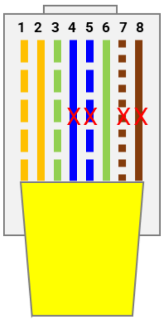
Notes QEC’s PoE (Power over Ethernet)

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:



T568B Color

PINS	10/100 DC on Spares (Mode B)
1	Rx+
2	Rx-
3	Tx+
4	DC+
5	DC+
6	Tx-
7	DC-
8	DC-

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 MDevice connects with a third-party EtherCAT SubDevice).

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

1. Connection and wiring hardware

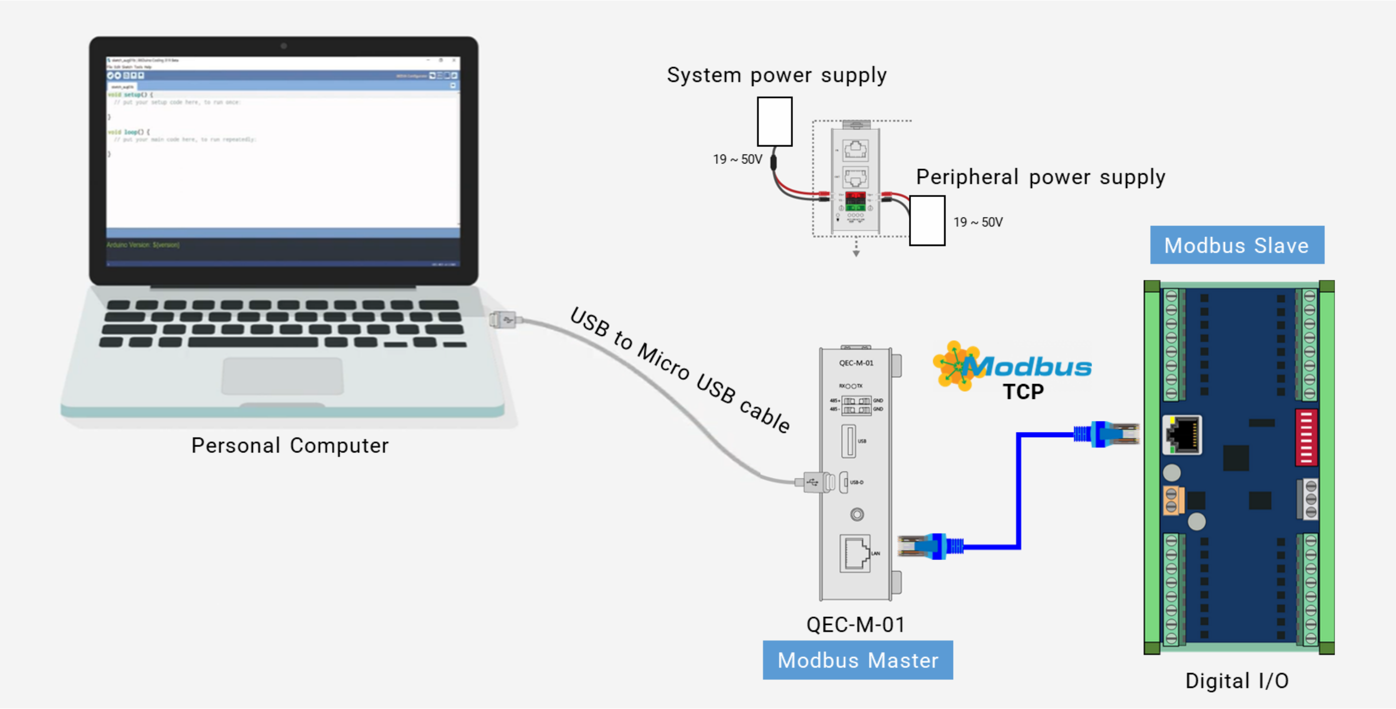
The following devices are used here:

1. QEC-M-01 (EtherCAT MDevice)
2. Modbus TCP Digital I/O Module (8-in & 8-out)
3. 24V power supply

1.1 QEC-M-01

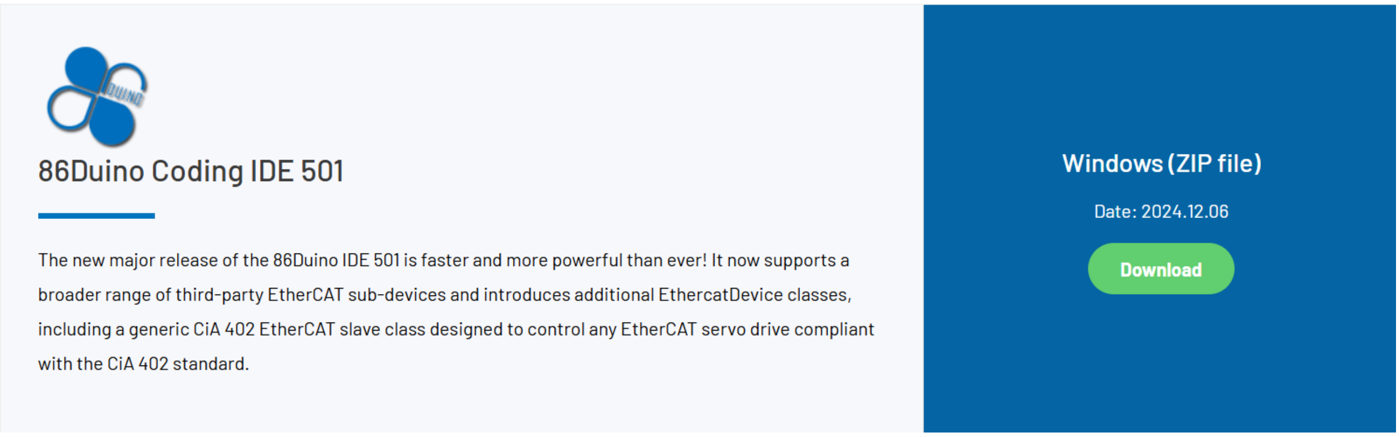
QEC EtherCAT MDevice.

1. Power Supply:
   Connect to Vs+/Vs- and Vp+/Vp- power supplies via EU terminals for 24V power.
2. Ethernet Connection:
   Using the Ethernet port (On the front side) connected to the LAN port on the Modbus IO Module via an RJ45 cable.



2. Software/Development Environment

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



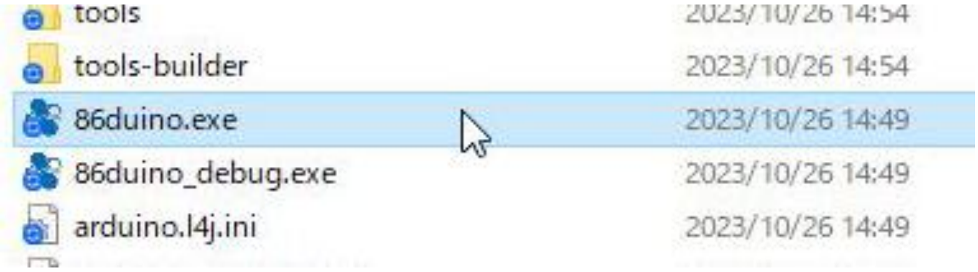
86Duino Coding IDE 501

The new major release of the 86Duino IDE 501 is faster and more powerful than ever! It now supports a broader range of third-party EtherCAT sub-devices and introduces additional EthercatDevice classes, including a generic CiA 402 EtherCAT slave class designed to control any EtherCAT servo drive compliant with the CiA 402 standard.

Windows (ZIP file)

Date: 2024.12.06

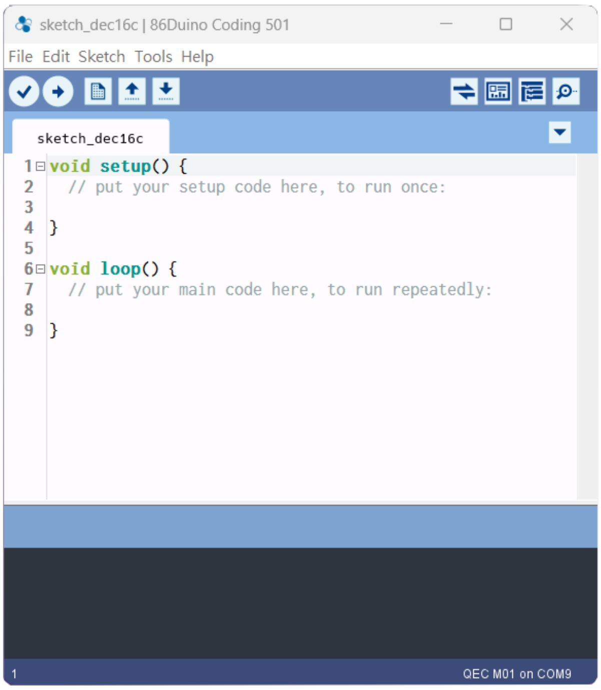
After downloading, please unzip the downloaded zip file, no additional software installation is required, just double-click 86duino.exe to start the IDE.



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 501+ looks like below.



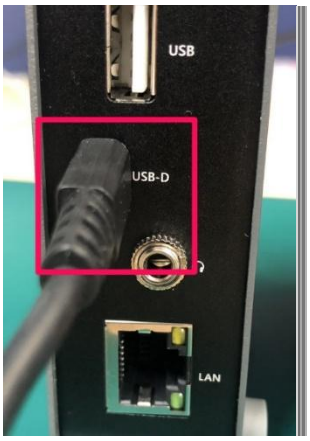
sketch_dec16c | 86Duino Coding 501

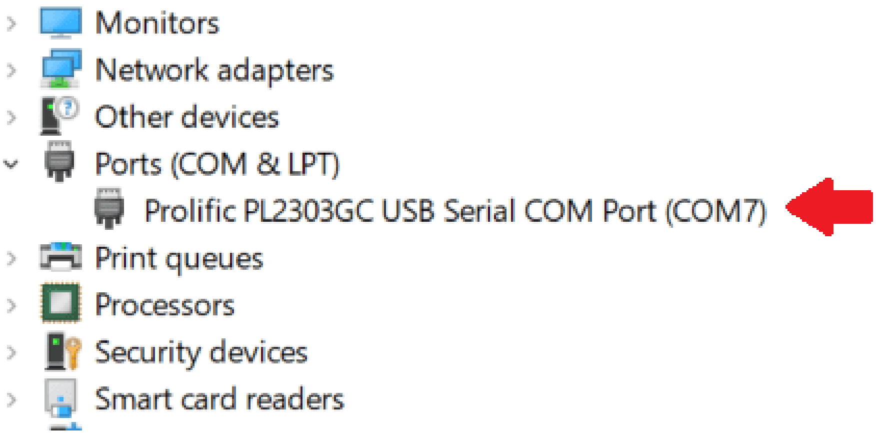
```
void setup() {
  // put your setup code here, to run once:

}

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

}
```

3. Connect to PC and set up the environment

Follow the steps below to set up the environment:

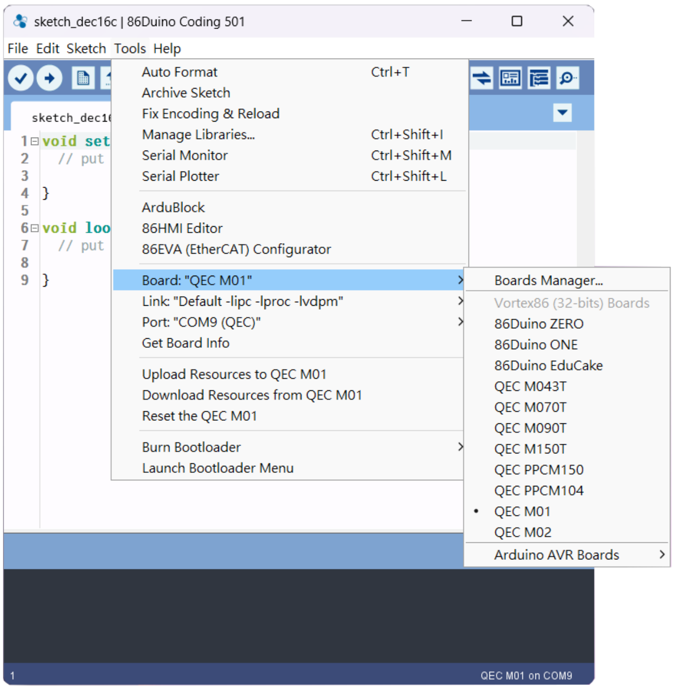
1. Connect the QEC-M-01 to your PC via a Micro USB to USB cable (86Duino IDE installed).
2. Turn on the QEC power.
3. Open "Device Manager" (select in the menu after pressing Win+X) ->" Ports (COM & LPT)" in your PC and expand the ports; you should see that the "Prolific PL2303GC USB Serial COM Port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

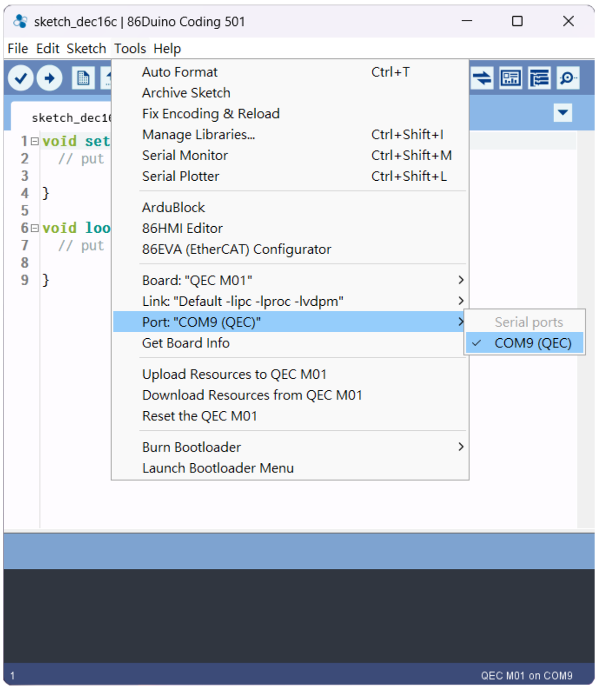




Monitors
Network adapters
Other devices
Ports (COM & LPT)
    Prolific PL2303GC USB Serial COM Port (COM7)
Print queues
Processors
Security devices
Smart card readers

4. Open the 86Duino IDE.
5. Select the correct board: In the IDE's menu, select Tools> Board > QEC-M-01 (or the QEC MDevice model you use).
6. Select Port: In the IDE's menu, select Tools > Port and select the USB port to connect to the QEC MDevice (in this case, COM9 (QEC)).




sketch_dec16c | 86Duino Coding 501
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: "COM9 (QEC)"
Get Board Info
Upload Resources to QEC M01
Download Resources from QEC M01
Reset the QEC M01
Burn Bootloader
Launch Bootloader Menu
Serial ports
✓ COM9 (QEC)
sketch_dec16
void set
// put
}
void loo
// put
}
QEC M01 on COM9

4. Write code

The QEC-M-01 controller can communicate with a remote DIO (Digital Input/Output) module via the Modbus TCP protocol. The development is carried out using the 86Duino IDE and the **Ethernet** and **Modbus** libraries.

In the Arduino programming model, the `setup()` function is used for initial configuration, and the `loop()` function handles the main logic. This example demonstrates reading a holding register and writing to digital outputs (Coils) every second.

Hardware Configuration

- Modbus Master Device: QEC-M-01
- Modbus Slave Device: DIO module at IP address 192.168.1.80, and ID is 28.
- Communication Protocol: Modbus TCP
- Ethernet MAC/IP: Assigned to QEC-M-01 for network access

A. In Setup Function

The `setup()` function initializes the Modbus TCP connection and prepares the system for communication.

1. Begin serial communication at 115200 baud for debugging purposes.
2. Set the MAC address and local IP address of the QEC-M-01.
3. Begin Modbus TCP communication with the slave device.

B. In Loop Function

The `loop()` function executes once per second and performs the following:

1. Read 1 word (16 bits) from holding register address 4 of slave device 28.
2. Use the read value to set 8 digital outputs (coils) starting from address 0.
3. Print the binary value of the read data to the Serial Monitor.
4. If the Modbus read fails, an error message is printed.

Key API Functions:

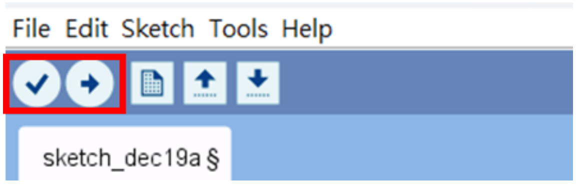
- `readHoldingRegisters(uint8_t slave_id, uint16_t address, uint16_t size, uint16_t *data);`
- `writeMultipleCoils(uint8_t slave_id, uint16_t address, uint16_t size, uint16_t *data);`

The example code is as follows:

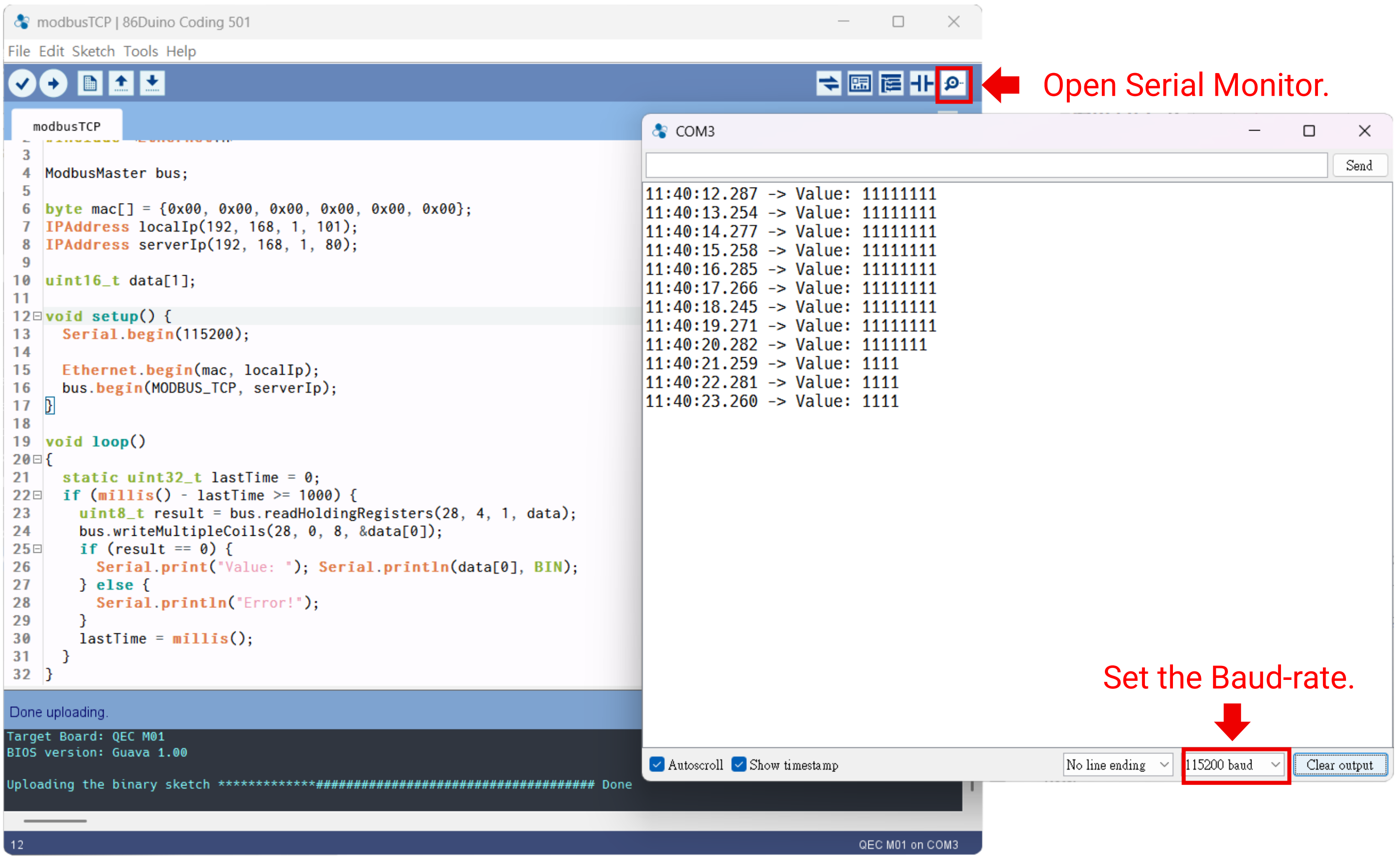
```
#include "Modbus.h"
#include <Ethernet.h>

ModbusMaster bus;

byte mac[] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
IPAddress localIp(192, 168, 1, 101);
IPAddress serverIp(192, 168, 1, 80);

uint16_t data[1];

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

  Ethernet.begin(mac, localIp);
  bus.begin(MODBUS_TCP, serverIp);
}

void loop()
{
  static uint32_t lastTime = 0;
  if (millis() - lastTime >= 1000) {
    uint8_t result = bus.readHoldingRegisters(28, 4, 1, data);
    bus.writeMultipleCoils(28, 0, 8, &data[0]);
    if (result == 0) {
      Serial.print("Value: "); Serial.println(data[0], BIN);
    } else {
      Serial.println("Error!");
    }
    lastTime = millis();
  }
}
```

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.



After you successfully upload the program to the QEC-M-01, you can open the Serial Monitor on 86Duino IDE. Please check that the Serial baud rate is the same as your setting.



Open Serial Monitor.

Set the Baud-rate.

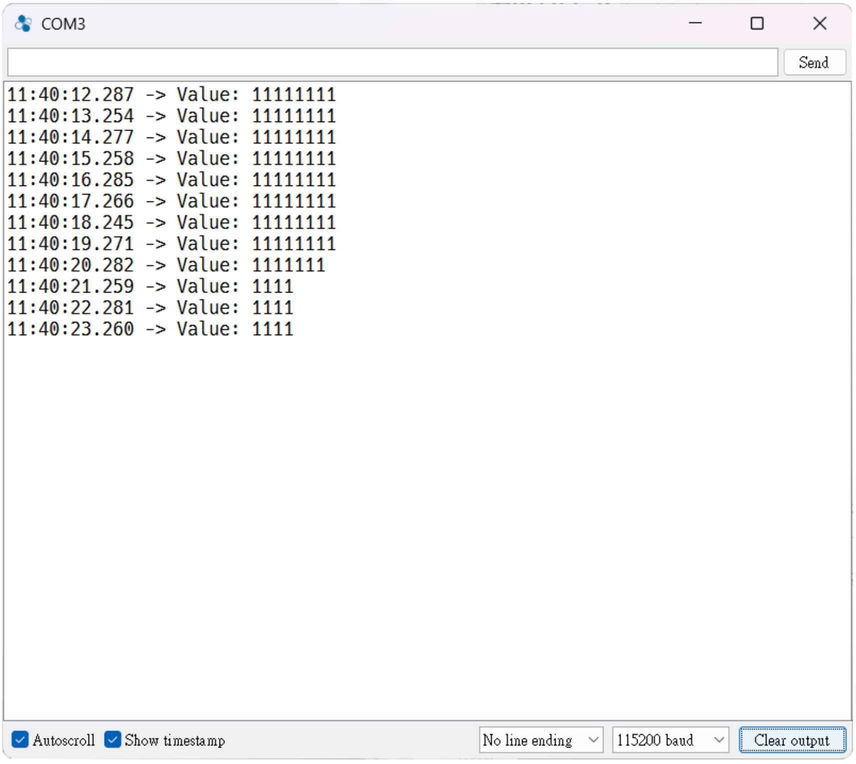
```
ModbusMaster bus;

byte mac[] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
IPAddress localIp(192, 168, 1, 101);
IPAddress serverIp(192, 168, 1, 80);

uint16_t data[1];

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

  Ethernet.begin(mac, localIp);
  bus.begin(MODBUS_TCP, serverIp);
}

void loop()
{
  static uint32_t lastTime = 0;
  if (millis() - lastTime >= 1000) {
    uint8_t result = bus.readHoldingRegisters(28, 4, 1, data);
    bus.writeMultipleCoils(28, 0, 8, &data[0]);
    if (result == 0) {
      Serial.print("Value: "); Serial.println(data[0], BIN);
    } else {
      Serial.println("Error!");
    }
    lastTime = millis();
  }
}
```

```
Done uploading.
Target Board: QEC M01
BIOS version: Guava 1.00

Uploading the binary sketch *************######################################## Done
```

```
11:40:12.287 -> Value: 11111111
11:40:13.254 -> Value: 11111111
11:40:14.277 -> Value: 11111111
11:40:15.258 -> Value: 11111111
11:40:16.285 -> Value: 11111111
11:40:17.266 -> Value: 11111111
11:40:18.245 -> Value: 11111111
11:40:19.271 -> Value: 11111111
11:40:20.282 -> Value: 1111111
11:40:21.259 -> Value: 1111
11:40:22.281 -> Value: 1111
11:40:23.260 -> Value: 1111
```

If the Modbus communication configuration is successful and starts to read/write, the Serial Monitor will print “Value: ...”, which means all configurations are successful.

It will print the Modbus IO Module's current status of Input and control Output to the serial monitor.



```
11:40:12.287 -> Value: 11111111
11:40:13.254 -> Value: 11111111
11:40:14.277 -> Value: 11111111
11:40:15.258 -> Value: 11111111
11:40:16.285 -> Value: 11111111
11:40:17.266 -> Value: 11111111
11:40:18.245 -> Value: 11111111
11:40:19.271 -> Value: 11111111
11:40:20.282 -> Value: 1111111
11:40:21.259 -> Value: 1111
11:40:22.281 -> Value: 1111
11:40:23.260 -> Value: 1111
```

Warranty

This product is warranted to be in good working order for a period of one year from the date of purchase. Should this product fail to be in good working order at any time during this period, we will, at our option, replace or repair it at no additional charge except as set forth in the following terms. This warranty does not apply to products damaged by misuse, modifications, accident or disaster. Vendor assumes no liability for any damages, lost profits, lost savings or any other incidental or consequential damage resulting from the use, misuse of, originality to use this product. Vendor will not be liable for any claim made by any other related party. Return authorization must be obtained from the vendor before returned merchandise will be accepted. Authorization can be obtained by calling or faxing the vendor and requesting a Return Merchandise Authorization (RMA) number. Returned goods should always be accompanied by a clear problem description.

All Trademarks appearing in this manuscript are registered trademark of their respective owners. All Specifications are subject to change without notice.

©ICOP Technology Inc. 2025