Start Guide

NPM EC-AD1441A4 EtherCAT Driver CiA402 PP Mode (4-axis)

86Duino Coding IDE 501

EtherCAT Library

(Version 1.0)

Revision

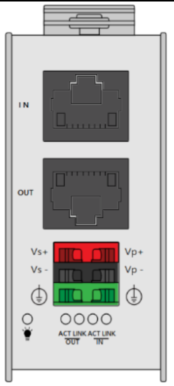
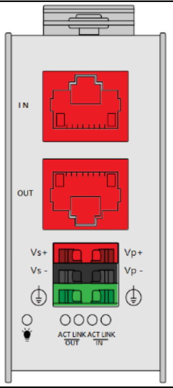
Date	Version	Description
2025/1/23	Version1.0	New Release.

Preface

In this guide, we will show you how to use the EtherCAT MDevice QEC-M-01 and the NPM EC-AD1441A4 (2-phase Bipolar 4-axis stepping motor driver).

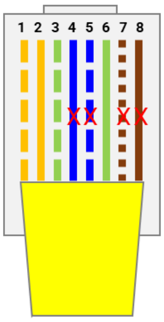
Notes QEC’s PoE (Power over Ethernet)

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.

 IN OUT Vs+ Vs - Vp+ Vp - ACT LINK ACT LINK OUT IN	 IN OUT Vs+ Vs - Vp+ Vp - ACT LINK ACT LINK OUT IN
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-

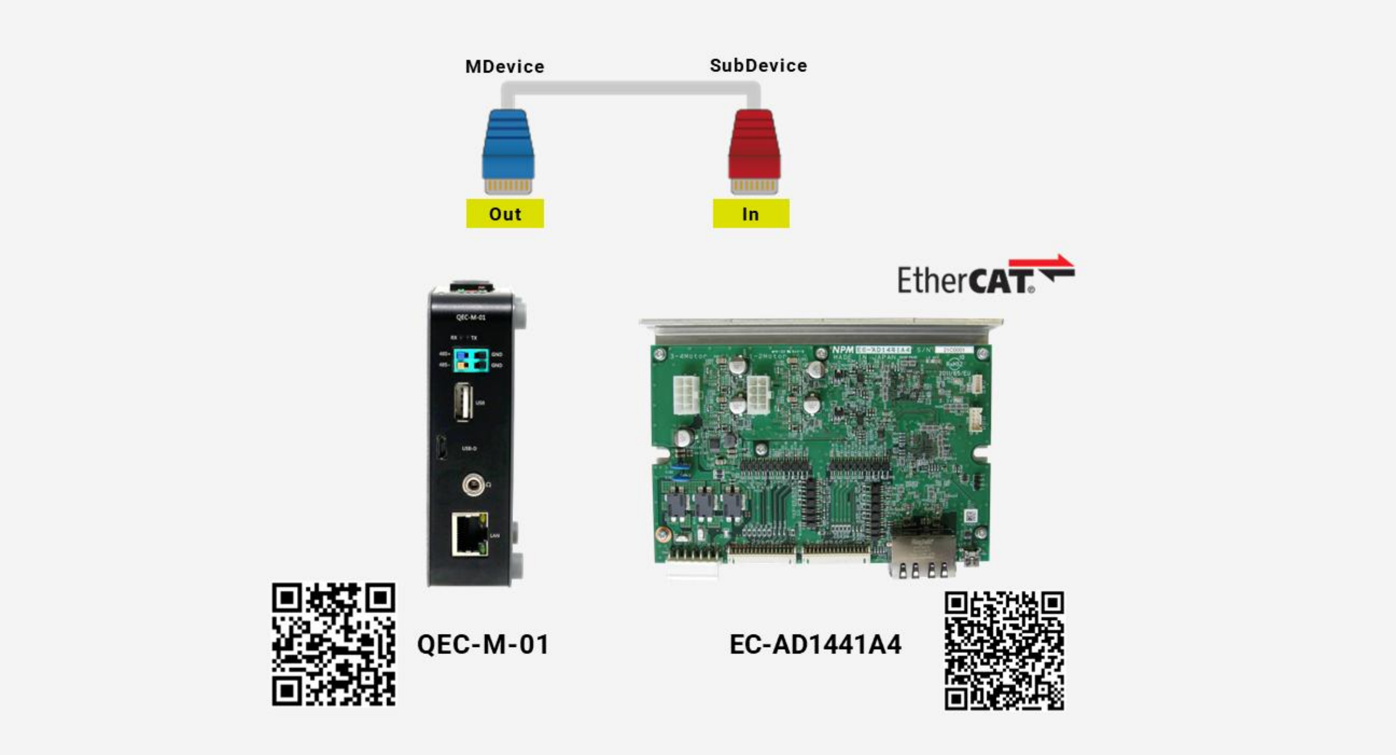
2. When connecting PoE and non-PoE devices, make sure to disconnect Ethernet cables at pins 4, 5, 7, and 8 (e.g., when a PoE-supported QEC EtherCAT MDevice connects with a third-party EtherCAT SubDevice).

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

1. Connection and wiring hardware

The following devices are used here:

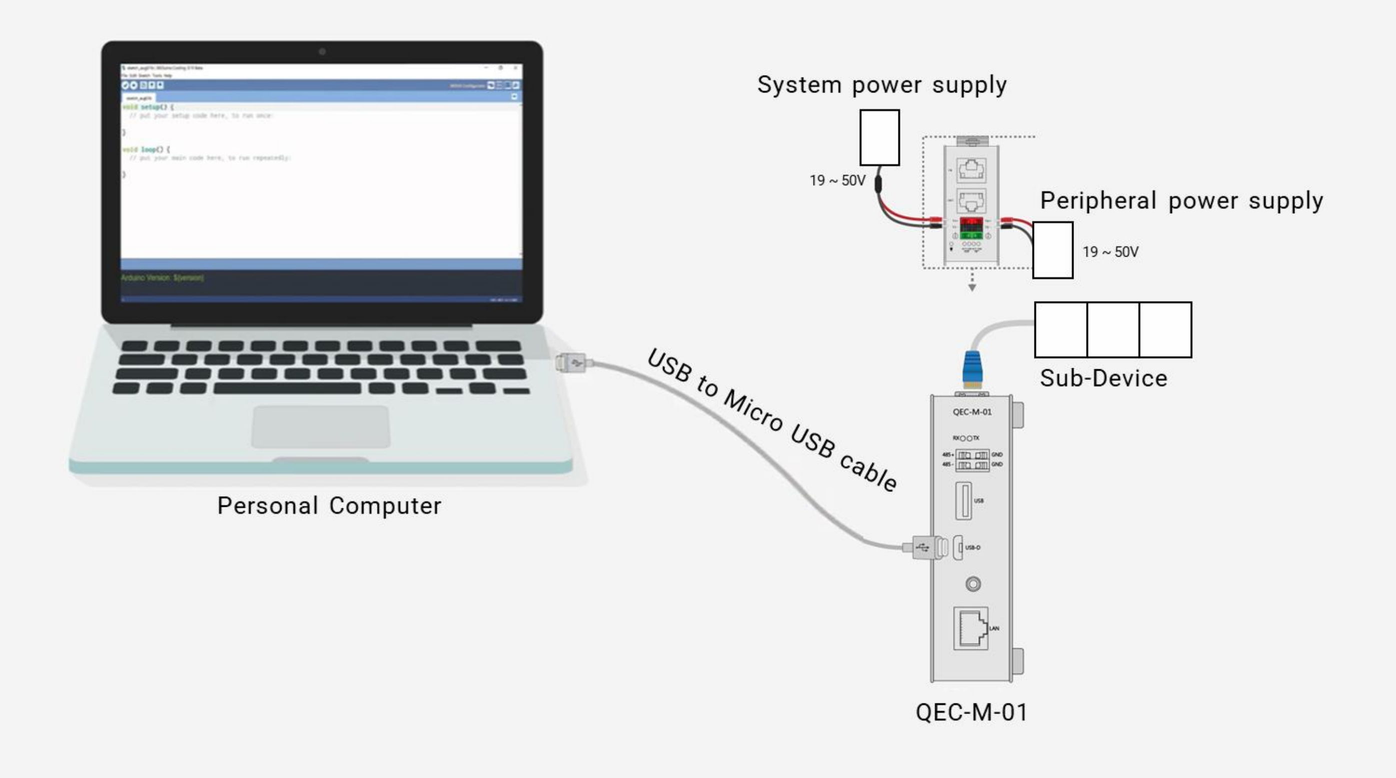
1. QEC-M-01 (EtherCAT MDevice)
2. NPM EC-AD1441A4 (2-phase Bipolar 4-axis stepping motor driver)
3. 24V power supply & EU-type terminal cable & LAN cable
4. 86STEP-42*4 (Encoder in Stepper Motor, size 42 mm square)



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. EtherCAT Connection: Using the EtherCAT Out port (On the top side) connected to the EtherCAT In port of EtherCAT SubDevice via RJ45 cable.

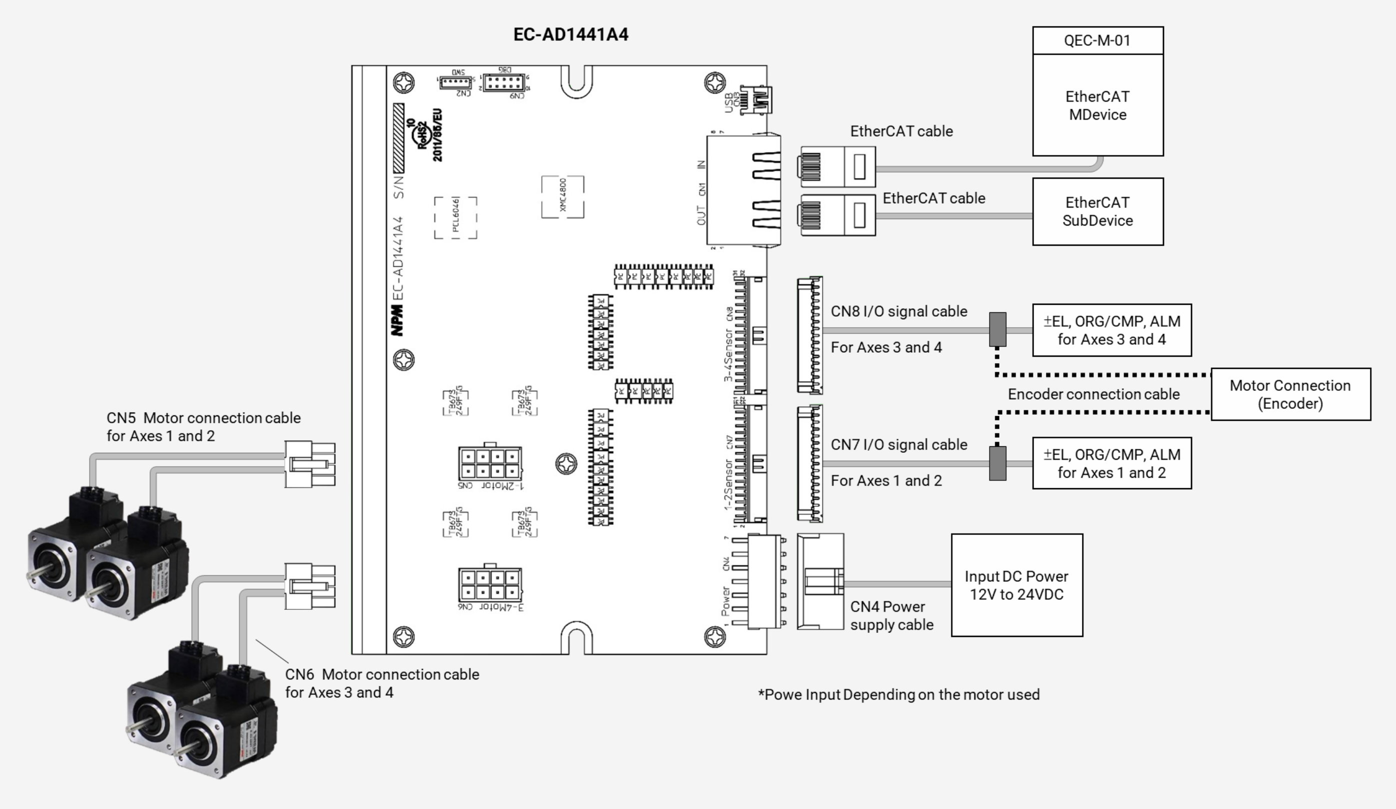


```
void setup() {
  // put your setup code here, to run once:

}

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

}
```

Arduino Version: $[version]

1.2 EC-AD1441A4

EC-AD1441A4, an EtherCAT Bipolar constant current, Four axes 2-phase stepping motor driver. This figure shows an example of when the **86STEP-42** motor is connected.



1. EtherCAT Connectivity:
   - Two EtherCAT communication ports (IN and OUT) connect the MDevice (QEC-M01) and the other SubDevices.
2. Signal Connections:
   - CN7: I/O signal interface for Axes 1 and 2 (EL, ORG, CMP, ALM signals).
   - CN8: I/O signal interface for Axes 3 and 4 (EL, ORG, CMP, ALM signals).
   - Encoder Connection: Tracks motor positions for each axes.
3. Motor Connections:
   - CN5: Motor connection cable for Axes 1 and 2.
   - CN6: Motor connection cable for Axes 3 and 4.
4. Power Supply:
   - CN4: Input DC power (12V~24V), powering the device and motors.
5. Safety and Status:
   - Emergency signals (EMG, ALM) are integrated for safety.
   - LED indicators display device and EtherCAT status.

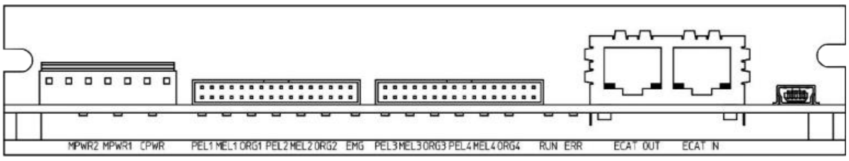
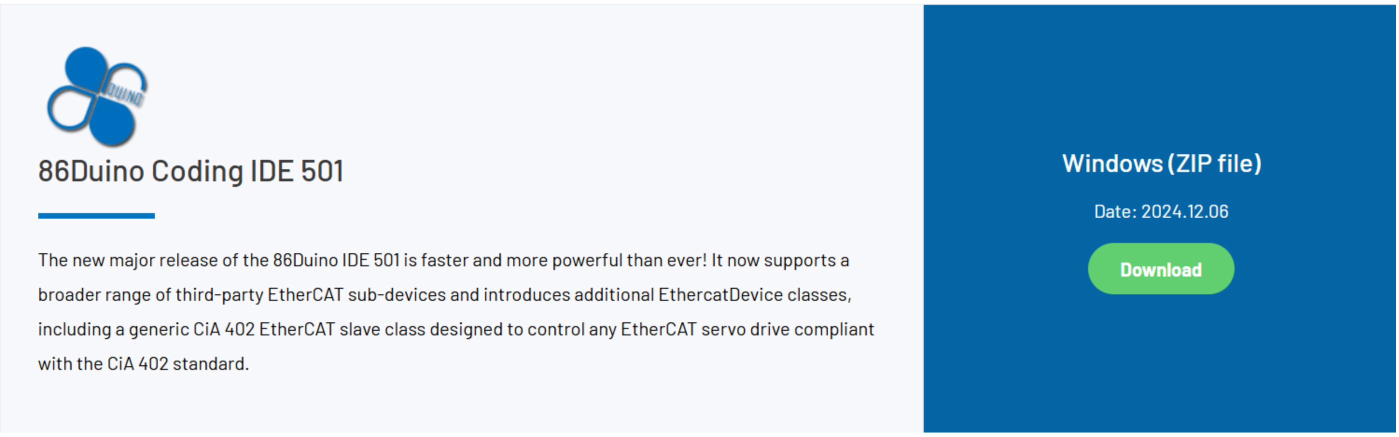


Figure: EC-AD1441A4Side view

2. Software/Development Environment

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



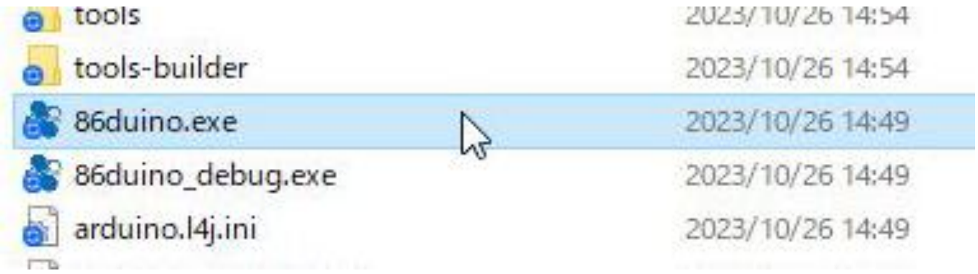
86Duino Coding IDE 501

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

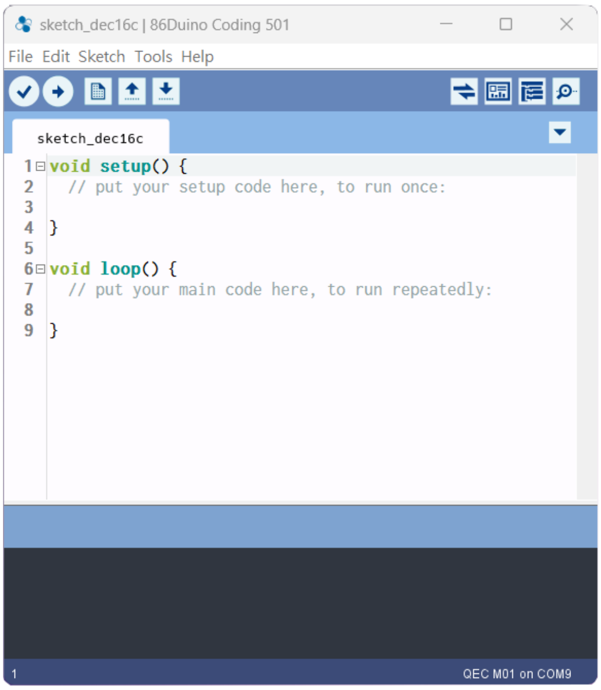
After downloading, please unzip the downloaded zip file, no additional software installation is required, just double-click 86duino.exe to start the IDE.



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.



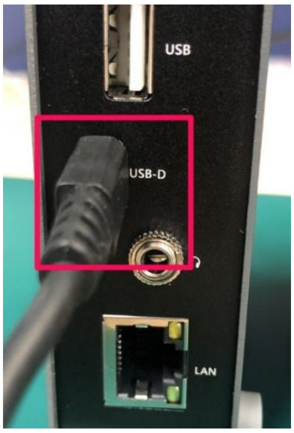
sketch_dec16c | 86Duino Coding 501
File Edit Sketch Tools Help
sketch_dec16c

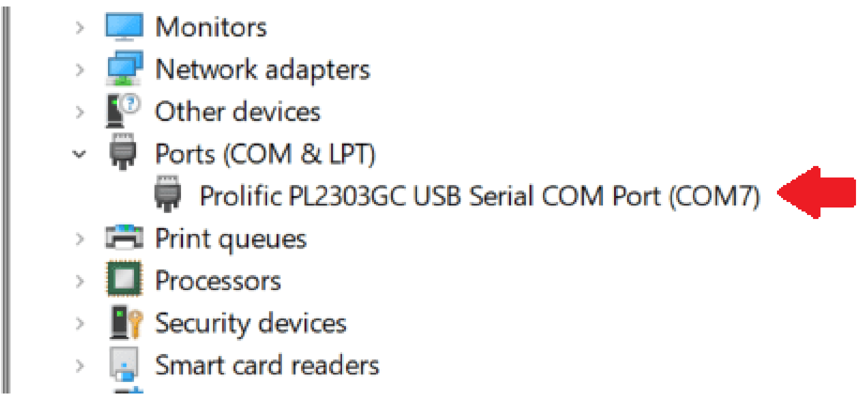
```
void setup() {
  // put your setup code here, to run once:

}

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:

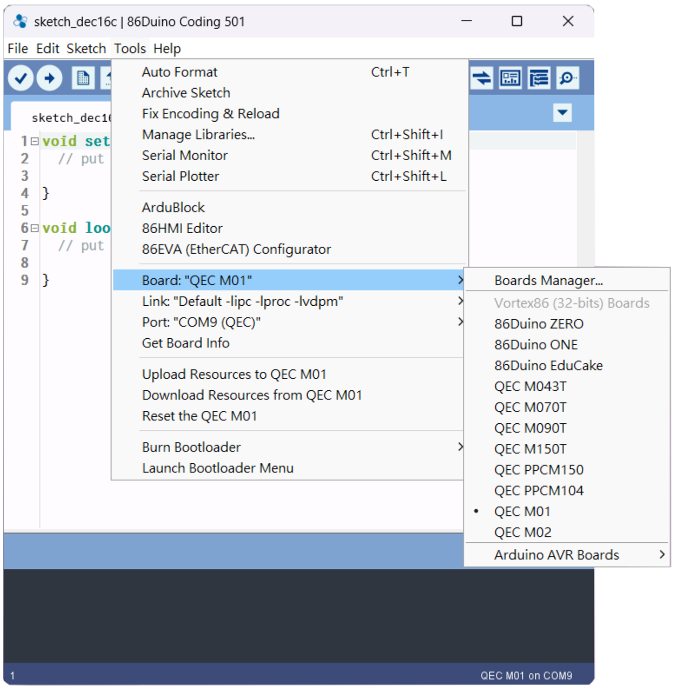
1. Connect the QEC-M-01 to your PC via a Micro USB to USB cable (86Duino IDE installed).
2. Turn on the QEC power.
3. Open “Device Manager” (select in the menu after pressing Win+X) ->” Ports (COM & LPT)” in your PC and expand the ports; you should see that the “Prolific PL2303GC USB Serial COM Port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

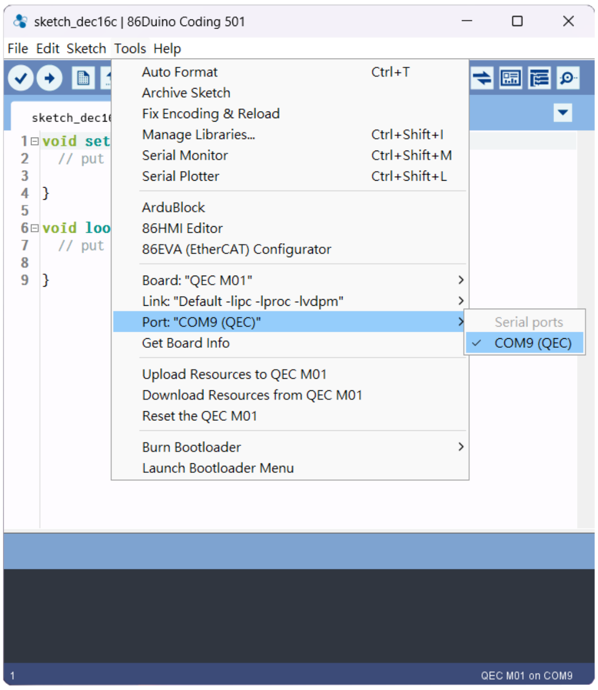




- Monitors
- Network adapters
- Other devices
- Ports (COM & LPT)
  - Prolific PL2303GC USB Serial COM Port (COM7)
- Print queues
- Processors
- Security devices
- Smart card readers

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)
QEC M01 on COM9

4. Write code

The EtherCAT MDevice (QEC-M-01) and the NPM EC-AD1441A4 (2-phase Bipolar 4-axis stepping motor driver) 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.

The following program sets the EC-AD1441A4 into CiA402 Profile Position (PP) mode:

- EtherCAT Cycle Time: 1 millisecond.
- EtherCAT Mode: ECAT_SYNC.

The EthercatMaster object ("master") represents the QEC-M-01, while there are 4 EthercatDevice_CiA402 objects ("motor[i]") represent 4 axes on the NPM EC-AD1441A4 stepper motor driver.

A. In Setup Function:

In the setup() function initializes communication and configures the motor for CiA402 Profile Position (PP) mode. Follow the steps below:

1. Initialize Serial Communication
   - Start serial communication at a baud rate of 115200.
2. Start the EtherCAT MDevice
   - Use the begin() function to initial the EtherCAT MDevice and set the EtherCAT state machine to the PRE-OPERATIONAL state.
3. Attach EtherCAT CiA402 Motors to the EtherCAT MDevice:
   - Use the attach(0, i, master) function to attach the stepper motor axis on the EtherCAT SubDevice to the EtherCAT Network.
4. Config Distributed Clock (DC) Mode
   - Use the setDc() function to set the EC-AD1441A4 Driver to DC mode and synchronize the cycle time with EtherCAT communication.
5. Start the EtherCAT MDevice
   - Use the start() function to switch the EtherCAT state machine to the OPERATIONAL state. Set the cycle time to 1ms and ECAT_SYNC mode.
6. Set Profile Position (PP) Mode
   - Configure the motor to PP mode using setCiA402Mode(CIA402_PP_MODE).
7. Enable the Motor
   - Use the enable() function to enable the motor and transition it to CIA402_OPERATION_ENABLED.
8. Configure Profile Parameters
   - Motion Profile Type: Linear Ramp, Profile Velocity: 50,000, Acceleration: 3,000, Deceleration: 3,000.

B. In Loop Function:

In the loop() function, the motor's current position is displayed on the Serial Monitor, and the motor alternates its movement back and forth.

1. State Machine Logic
    - case 0: Start the motor and move to the target position (100,000 units). Once the command is successfully executed, transition to the next state.
    - case 1: Wait for the motor to reach the target position. Once the target is reached, proceed to the next state.
    - case 2: Start the motor and move back to the original position (-100,000 units). Once the command is successfully executed, transition to the next state.
    - case 3: Wait for the motor to return to the original position. Once the target is reached, reset the state machine back to case 0 to repeat the movement cycle.
2. Tracks the number of completed cycles using pp_done. When all motors complete their cycles, resets all state machines (pp_state[]) and the pp_done counter.

Code Logic Summary

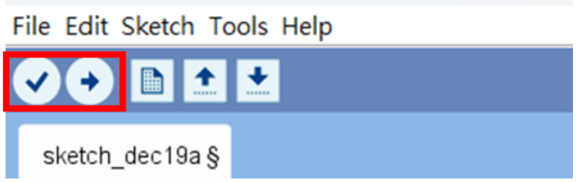
- Use the pp_Run() function to initiate position movement.
- Use the pp_IsTargetReached() function to confirm whether the target position has been reached.
- The state machine starts at case 0 and resets after completing case 3.

The example code is as follows:

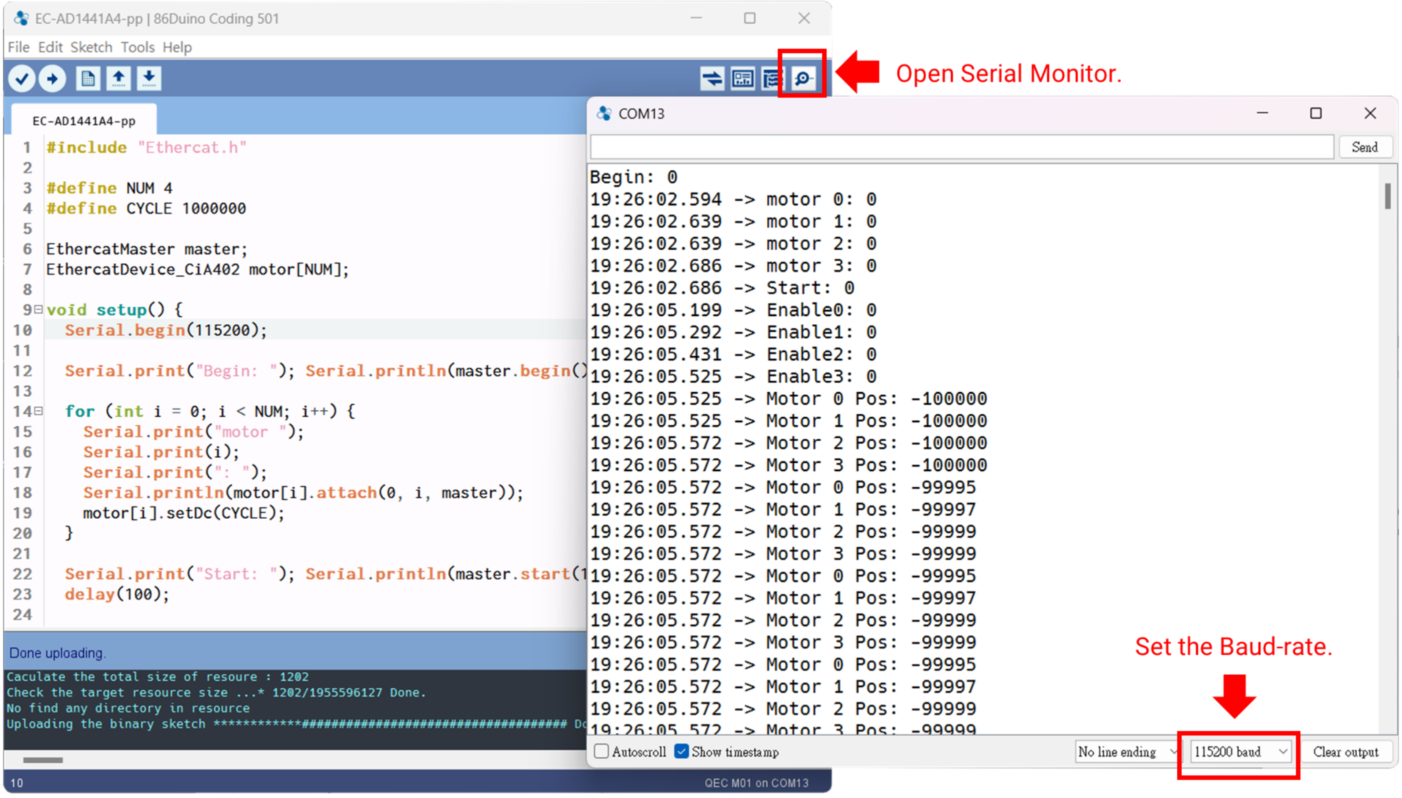
```
#include "Ethercat.h"

#define NUM 4
#define CYCLE 1000000

EthercatMaster master;
EthercatDevice_CiA402 motor[NUM];

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

  Serial.println(master.begin());

  for (int i = 0; i < NUM; i++) {
    Serial.println(motor[i].attach(0, i, master));
    Serial.println(motor[i].setDc(CYCLE));
  }

  Serial.println(master.start(CYCLE, ECAT_SYNC));
  delay(100);
  for (int i = 0; i < NUM; i++)
    Serial.println(motor[i].setCiA402Mode(CIA402_PP_MODE));


  for (int i = 0; i < NUM; i++) {
    delay(100);
    Serial.print("Enable");
    Serial.print(i);
    Serial.print(": ");
    Serial.println(motor[i].enable());
    motor[i].pp_SetMotionProfileType(0);
    motor[i].pp_SetVelocity(50000);
    motor[i].pp_SetAcceleration(3000);
    motor[i].pp_SetDeceleration(3000);
  }
}

int pp_state[NUM]; // State machine for PP
int pp_done = 0;
void loop() {

  for (int i = 0; i < NUM; i++) {
    Serial.print("Motor ");
    Serial.print(i);
    Serial.print(" Pos: ");
    Serial.println(motor[i].getPositionActualValue());

    switch (pp_state[i]) {
      case 0:
        if (motor[i].pp_Run(100000) == 0)
          pp_state[i]++;
        break;
      case 1:
        if (motor[i].pp_IsTargetReached())
          pp_state[i]++;
        break;
      case 2:
        if (motor[i].pp_Run(-100000) == 0)
          pp_state[i]++;
        break;
      case 3:
        if (motor[i].pp_IsTargetReached())
          pp_state[i] = 0;
        pp_done++;
        break;
    }
  }

  // Once all motors are done, reset states
  if (pp_done == NUM) {
    pp_done = 0;
    for (int i = 0; i < NUM; i++)
      pp_state[i] = 0;
  }
}
```

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 the Serial baud rate is same as your setting.



Open Serial Monitor.

Set the Baud-rate.

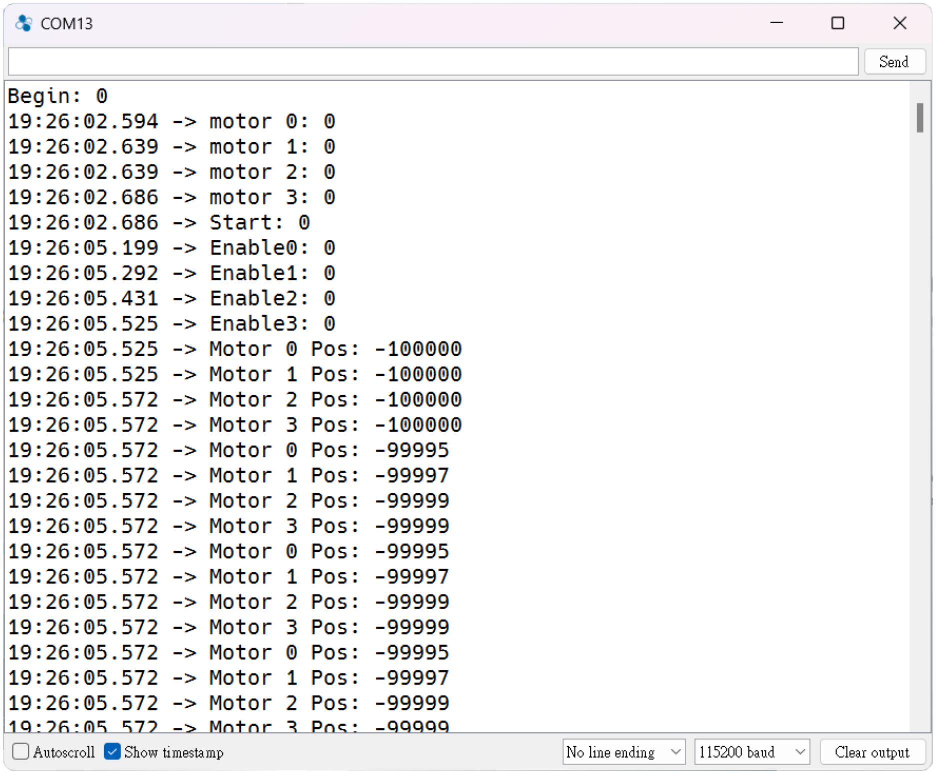
```
#include "Ethercat.h"

#define NUM 4
#define CYCLE 1000000

EthercatMaster master;
EthercatDevice_CiA402 motor[NUM];

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

  Serial.print("Begin: "); Serial.println(master.begin(

  for (int i = 0; i < NUM; i++) {
    Serial.print("motor ");
    Serial.print(i);
    Serial.print(": ");
    Serial.println(motor[i].attach(0, i, master));
    motor[i].setDc(CYCLE);
  }

  Serial.print("Start: "); Serial.println(master.start(1
  delay(100);

```

Done uploading.

```
Caculate the total size of resoure : 1202
Check the target resource size ...* 1202/1955596127 Done.
No find any directory in resource
Uploading the binary sketch ************#################################### Do
```

```
Begin: 0
19:26:02.594 -> motor 0: 0
19:26:02.639 -> motor 1: 0
19:26:02.639 -> motor 2: 0
19:26:02.686 -> motor 3: 0
19:26:02.686 -> Start: 0
19:26:05.199 -> Enable0: 0
19:26:05.292 -> Enable1: 0
19:26:05.431 -> Enable2: 0
19:26:05.525 -> Enable3: 0
19:26:05.525 -> Motor 0 Pos: -100000
19:26:05.525 -> Motor 1 Pos: -100000
19:26:05.572 -> Motor 2 Pos: -100000
19:26:05.572 -> Motor 3 Pos: -100000
19:26:05.572 -> Motor 0 Pos: -99995
19:26:05.572 -> Motor 1 Pos: -99997
19:26:05.572 -> Motor 2 Pos: -99999
19:26:05.572 -> Motor 3 Pos: -99999
19:26:05.572 -> Motor 0 Pos: -99995
19:26:05.572 -> Motor 1 Pos: -99997
19:26:05.572 -> Motor 2 Pos: -99999
19:26:05.572 -> Motor 3 Pos: -99999
19:26:05.572 -> Motor 0 Pos: -99995
19:26:05.572 -> Motor 1 Pos: -99997
19:26:05.572 -> Motor 2 Pos: -99999
19:26:05.572 -> Motor 3 Pos: -99999
```

If the EtherCAT communication config successful, Serial Monitor will print “motor i: 0”, “Start: 0”, and “Enable i: 0”. Then, it will print the motor’s current position to the serial monitor.

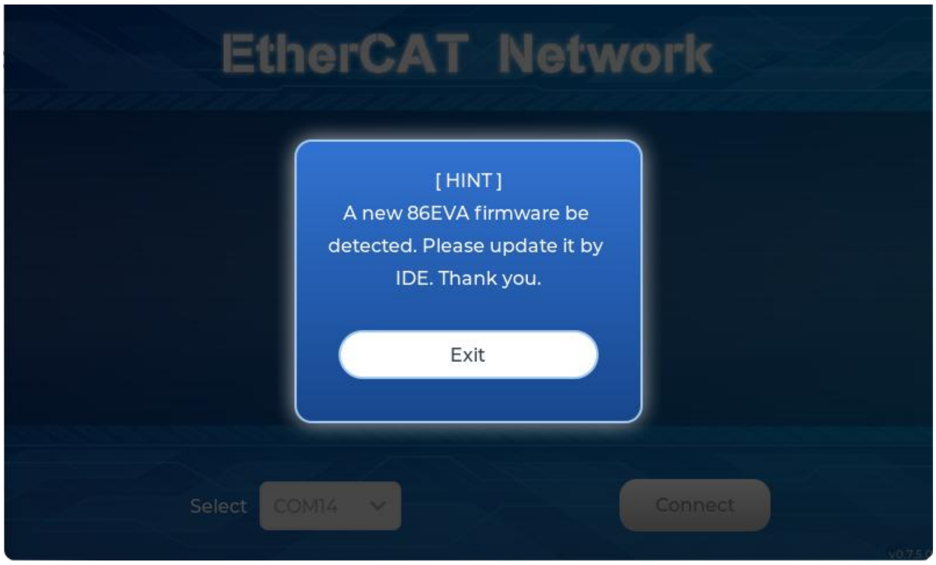


```
Begin: 0
19:26:02.594 -> motor 0: 0
19:26:02.639 -> motor 1: 0
19:26:02.639 -> motor 2: 0
19:26:02.686 -> motor 3: 0
19:26:02.686 -> Start: 0
19:26:05.199 -> Enable0: 0
19:26:05.292 -> Enable1: 0
19:26:05.431 -> Enable2: 0
19:26:05.525 -> Enable3: 0
19:26:05.525 -> Motor 0 Pos: -100000
19:26:05.525 -> Motor 1 Pos: -100000
19:26:05.572 -> Motor 2 Pos: -100000
19:26:05.572 -> Motor 3 Pos: -100000
19:26:05.572 -> Motor 0 Pos: -99995
19:26:05.572 -> Motor 1 Pos: -99997
19:26:05.572 -> Motor 2 Pos: -99999
19:26:05.572 -> Motor 3 Pos: -99999
19:26:05.572 -> Motor 0 Pos: -99995
19:26:05.572 -> Motor 1 Pos: -99997
19:26:05.572 -> Motor 2 Pos: -99999
19:26:05.572 -> Motor 3 Pos: -99999
19:26:05.572 -> Motor 0 Pos: -99995
19:26:05.572 -> Motor 1 Pos: -99997
19:26:05.572 -> Motor 2 Pos: -99999
19:26:05.572 -> Motor 3 Pos: -99999
```

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 MDevice's environment is abnormal. As shown in the figure below, please try updating your QEC EtherCAT MDevice's environment, which will include the following three items: Bootloader, EtherCAT firmware, and EtherCAT tool.



[HINT]
A new 86EVA firmware be detected. Please update it by IDE. Thank you.

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

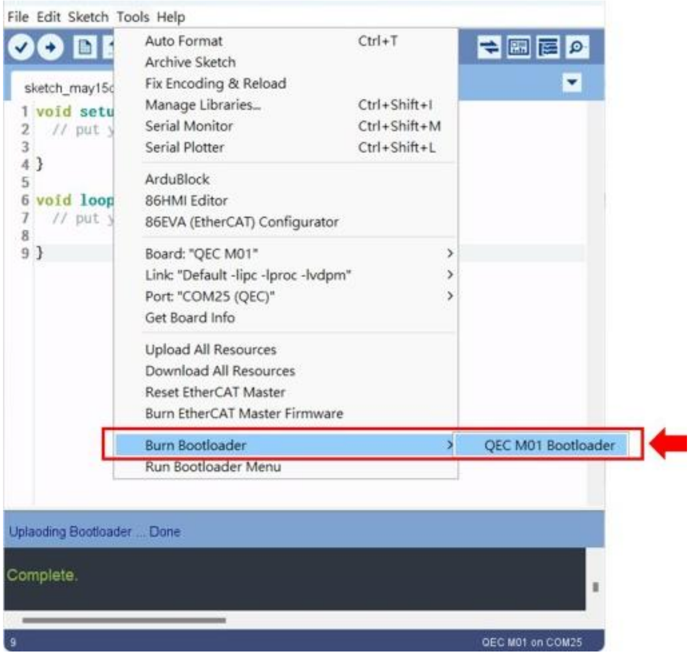
Step 1: Setting up QEC-M

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:

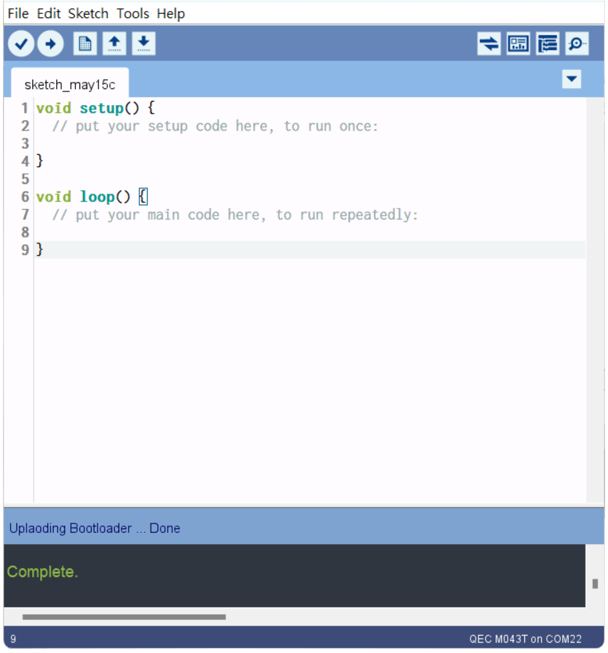


```
void setu
  // put y

}

void loop
  // put y

}
```

Uplaoding Bootloader ... Done

Complete.

Step 3: Complete the Update



```
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.

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.