

# OpenEVSE

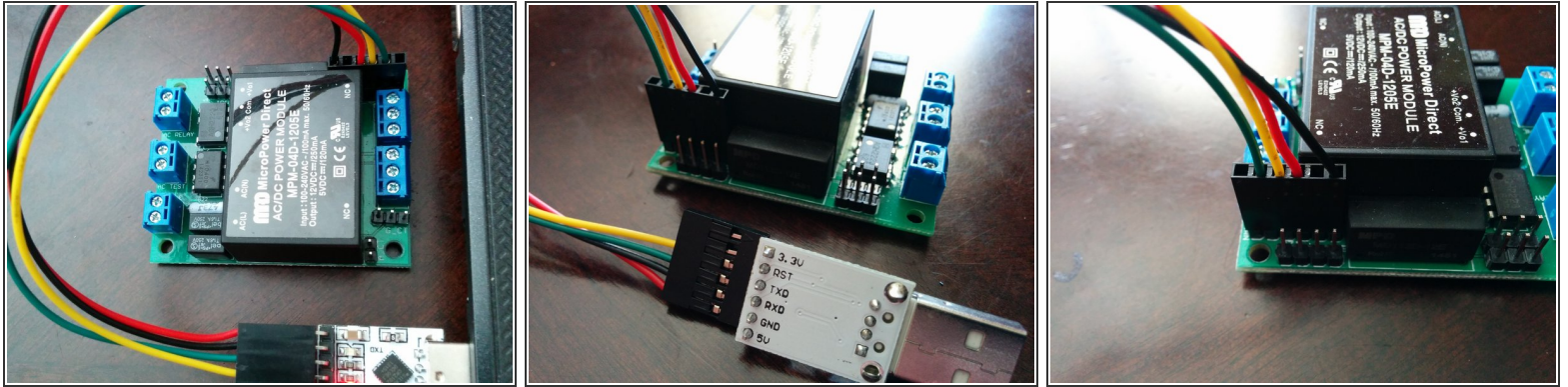
## Serial Communications with OpenEVSE

OpenEVSE has included serial communications for the life of the project. Early versions of firmware supported a Command Line Interface (CLI) newer versions include a Remote API.

Written By: Christopher Howell

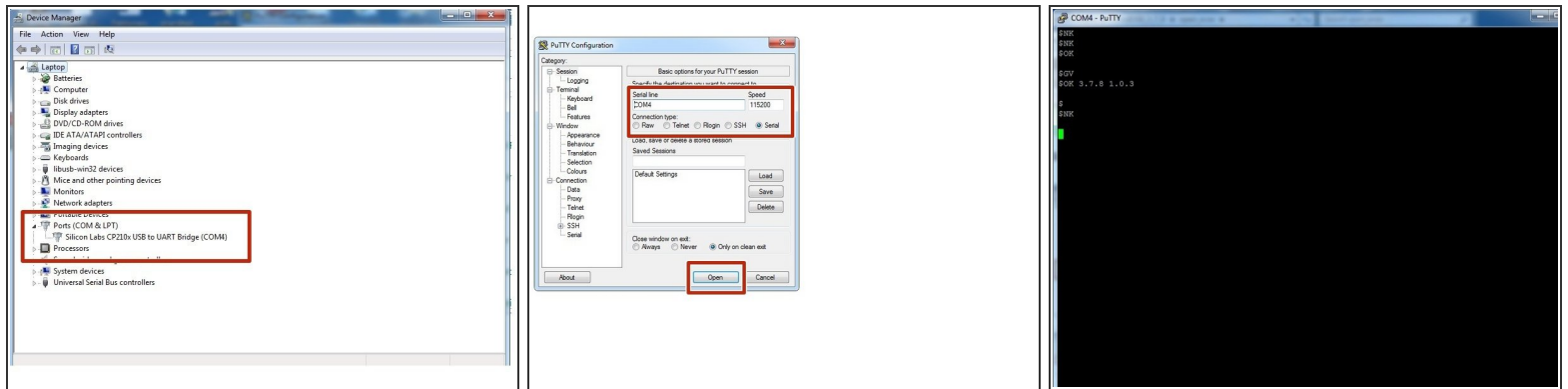


## Step 1 — Hardware



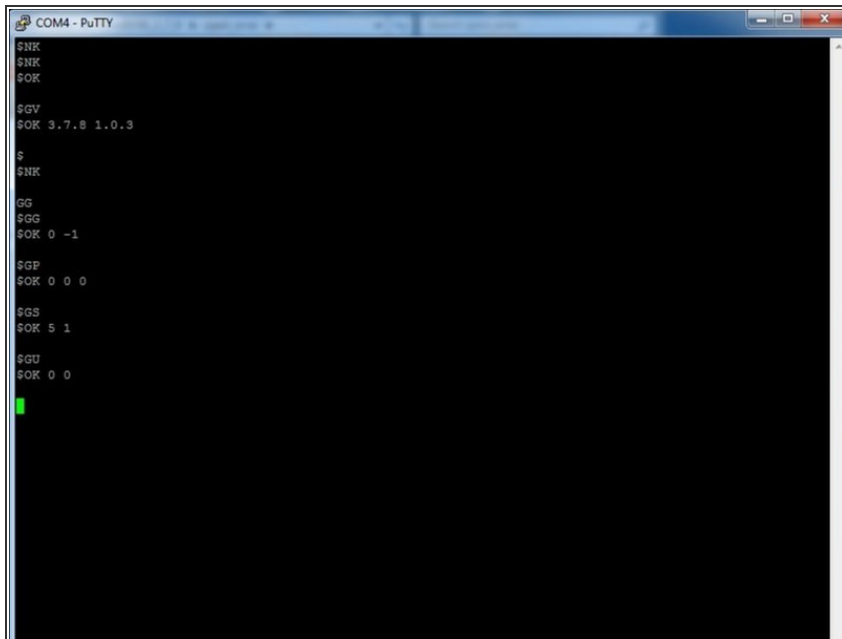
- OpenEVSE supports many Serial TTL Devices at 5v on the 6 pin header.
  - The header is the common FTDI format.
    - Pin 1 - Ground (Black)
    - Pin 2 - No Connection
    - Pin 3 - 5v (Red) - **75ma max** Higher power devices must use external power.
    - Pin 4 - Recieve (Yellow) connects to transmit on the remote device.
    - Pin 5 - Transmit (Green) connects to receive on the remote device.
- ⚠ It is recommended to update the latest version of firmware. Many changes have been made over time. This guide was written with Firmware version 3.7.8.

## Step 2 — USB-TTL - Serial Console



- Insert the serial to USB adapt or and install the appropriate driver
- ① The windows driver for the adaptor sold in the OpenEV store is here: [Windows Driver](#)
- Using a terminal console (PuTTY recommended) set the COMM port to the USB-TTL adapter and the baud rate to 115200. Then click Open.
- ① If you do not know the comm port you can go into the Device manager and look under Ports.
- In the console window enter \$ and press enter. You should receive a \$NK response.
- Next enter \$SE 1. This will turn on local echo so you can see what you send. The response should be \$OK.
- Now enter the command \$GV. The response should be \$OK followed by the Firmware version and the Remote API version.

## Step 3 — Remote API Get commands



```
COM4 - PuTTY
$NRK
$NRK
$OK
$GV
$OK 3.7.8 1.0.3
$
$NK
$GG
$GG
$OK 0 -1
$GP
$OK 0 0 0
$GS
$OK 5 1
$GU
$OK 0 0
█
```

⚠ Full documentation of RAPI is located in the `rapi_proc.h` file in the OpenEVSE Source code. The file is located here: [rapi\\_proc.h](#)

- The Remote API is a very powerful tool for extending OpenEVSE, useful for information, configuration and external applications.
- Here are some interesting Get commands:
  - \$GG - Get real time charging current. Returns \$OK current voltage(future hardware)
  - \$GP - Get real time Temperature values from RTC chip, MCP9808 and TMP007 IR sensors. Returns \$OK RTC, MCP9808, TMP007 - 0 is returned if sensor is not found.
  - \$GS - Set EVSE State. Returns the current state \$OK State - 1 Not Connected - 2 Connected - 3 Charging - 4 Error - 5 Error.
  - \$GU - Get usage statistics. Returns \$OK Energy used last session and lifetime

## Step 4 — Remote API Set Commands

```
COM4 - PuTTY
$SO 1
$NK
$SO 1
$NK
$SO 0
$NK
$OK
$SC 24
$NK
$SS 2
$NK
$OK
$SC 24
$NK
$OK
$SL 1
$NK
$OK
$SS 12
$NK
$OK
$SL A
$NK
$OK
```

⚠ Full documentation of RAPI is located in the `rapi_proc.h` file in the OpenEVSE Source code. The file is located here: [rapi\\_proc.h](#)

- The Remote API is a very powerful tool for extending OpenEVSE, useful for information, configuration and external applications.
  - Here are some interesting Set commands:
    - \$SC amp - Set current value in amps. Subject to Mix and Max for each setting min 6A Max 80A.
    - \$SL 1 or 2 or A - set service level L1 /L2 / Auto
- 📌 Tip you can set Service level 1 set current for L1 then set Service Level 2 and set current. Each has its own setting.
- S1 yr mo day hr min sec - Set current time 2 digit value for each hour 24 hour value.
  - \$SO 0 or 1 - Set LCD Type 0 = Monochrome 1 = RGB.

## Step 5 — External Communications



- OpenEVSE can be extended with many devices using 5v TTL (or 3.3v with level shifting).
- ⚠ External power is required if device draws more than 75ma at 5v.