Thunderstruck Motors
EV Charger Controller
EVCC v2.0

Contents

Overview

The Electric Vehicle Charger Controller (EVCC) integrates charger CANBUS control and J1772 functionality in a simple to use, cost effective, and environmentally robust enclosure. Charge parameters such as maximum voltage, maximum current, and total charge time are configured, saved in nonvolatile memory, and used when charging to control a CAN enabled charger. The EVCC connects directly to analog "cell loop" Battery Management Systems (BMSs) and replaces the head end board, acting as a BMS master.

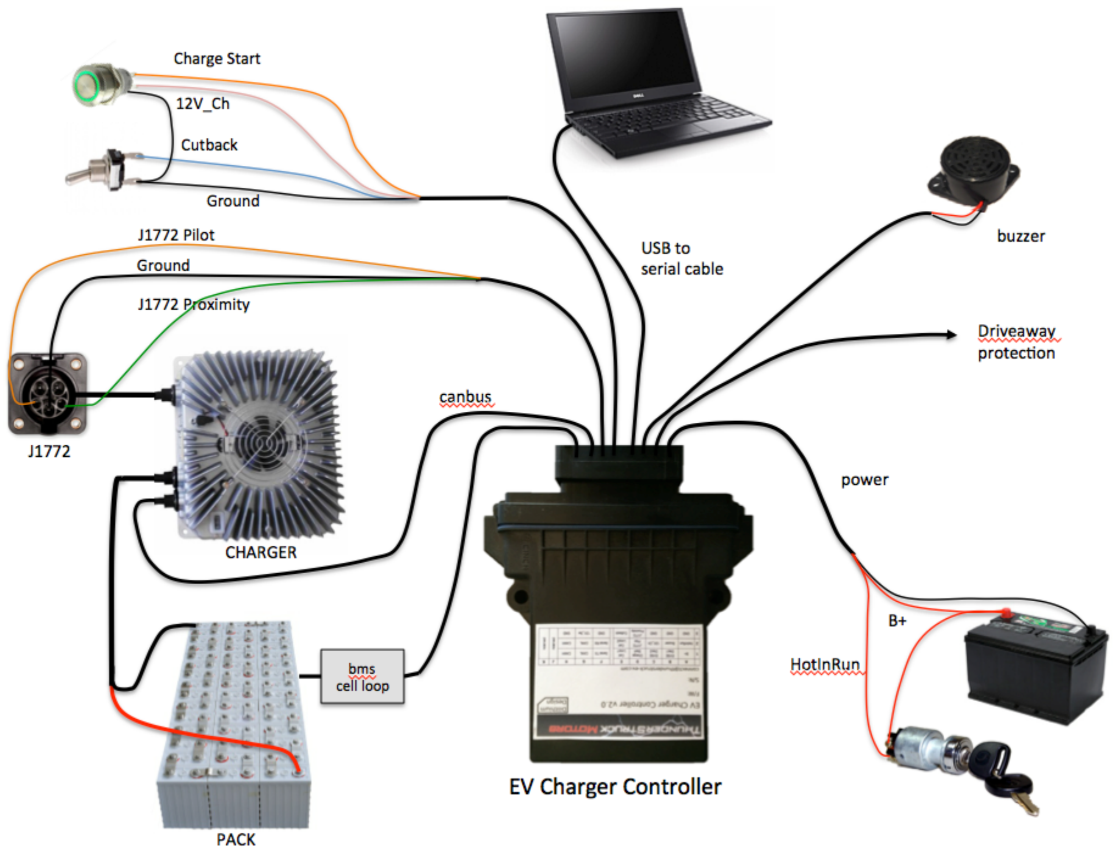


Figure 1 – EVCC System Diagram

The EVCC draws negligible current (less than 0.1 mA) when off. When charging, the EVCC is started by a momentary pushbutton and turns itself off when the charge cycle is completed. When charging, a 12V output is provided which can light an indicator light or drive a relay.

The EVCC is configured using a simple serial interface. The serial interface is used for configuration and debugging, but is not required for normal operation. Diagnostic commands are supported to verify proper wiring, to trace CANBUS messages, and to retrieve charging history.

The EVCC supports the SAE J1772 standard. J1772 defines the physical connector and protocols used between the charging station (known as the "Electric Vehicle Service Equipment"), and the Electric Vehicle. The J1772 Proximity signal is used to determine if the charger plug is present. "Driveaway protection" is supported so that the EV cannot be driven if the charge cable is still plugged in. The J1772 Pilot signal is used to start and stop charging. (The EVCC uses this signal to enable and disable the contactor in the EVSE).

The EVCC supports CH4100 and CAN-enabled ELCON chargers. Charging will stop if: the J1772 plug becomes unlocked, a cell loop error occurs, there is loss of communication between the EVCC and Charger, or the maximum configured charge time is reached. Charging also stops at the end of a normal charge cycle, which is achieved when the charging current drops below the minimum configured charge current.

Charging history is provided for the last sixteen charge cycles and includes: the reason that charging stopped, total charge time, maximum voltage, maximum current, final current, and watt hours.

When driving, the EVCC is started by the keyswitch. The EVCC can be used as a simple “BMS Master”. An output is provided that can be used to sound a buzzer if the cell loop is interrupted.

EVCC features work largely independently and it is not necessary to wire up or use all features. Installation may be customized per customer requirements.

The EVCC is housed in a 4.55” x 5.13” x 1.67” automotive grade water-resistant enclosure. All connections are made with a single 30pin connector. The EVCC is shipped with a pre-wired harness and with a USB to serial port cable.

Installation

Mechanical

The enclosure outline is shown below. It can be mounted in any convenient location, however it would ideally be located physically close to both the charger and the J1772 charge port.

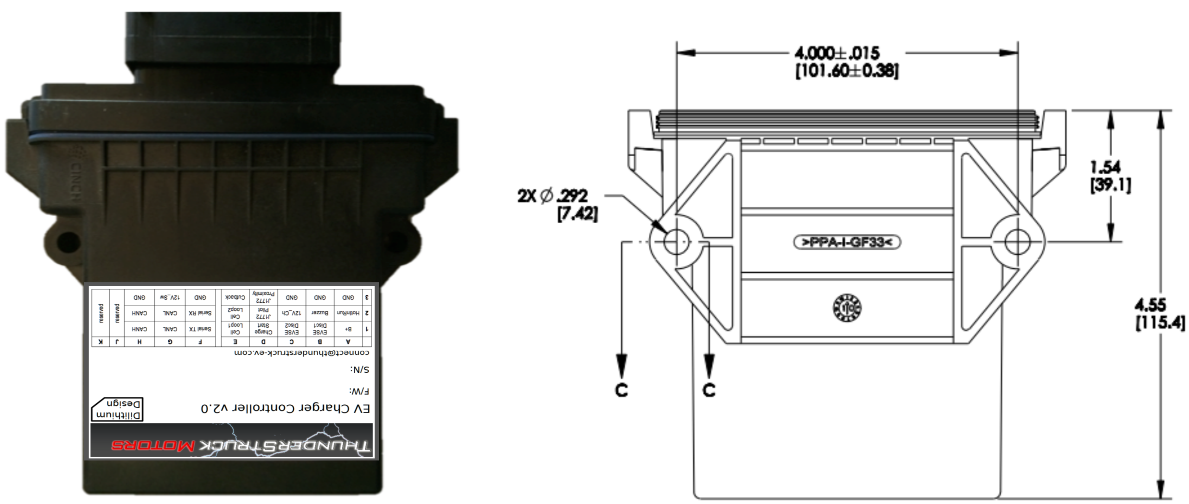


Figure 2 – EVCC Enclosure

The figure below shows the 30 pin connector and wiring harness. Note the LED to the right of the connector.

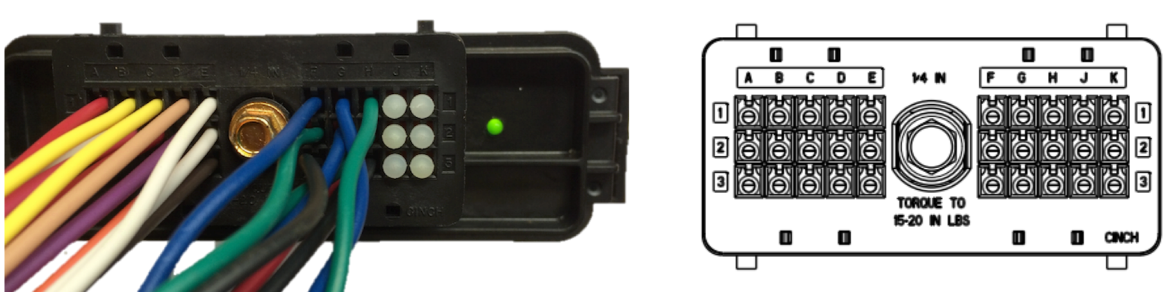


Figure 3 – EVCC Connector

The figure below shows the EVCC pinout.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 4 – EVCC Pinout

Power

B+ and **GND** (A3) are Power Inputs and should be connected to the EV 12V accessory battery.

HotInRun is connected to the Ignition swich. Supplying +12V to **HotIn** will turn the EVCC on.

Charge Start is used to start charging. By grounding this input (e.g., by a momentary pushbutton switch), the EVCC will power up and latch the power on. The EVCC automatically turns itself off when charging is complete.

12V_Ch and **12V_Sw** are outputs that can be used to drive 12V indicators, relays or instrumentation. **12V_Sw** is switched to B+ when the EVCC is powered up. **12V_Ch** is switched to B+ when the EVCC is Charging. These outputs are protected by 350ma resettable fuses.

> Note: The design intent of **Charge Start** and **12V_Ch** is to mount a momentary pushbutton and a 12V indicator near the J1772 charge port. Charging is begun by plugging in the charger plug, pushing the button, and observing the light come on. See EVCC System Diagram.

The figure below shows the Power connections.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 5 - Power Connections

J1772

The figure below shows the J1772 EV side connector and locations of the J1772 Proximity and J1772 Pilot signals. These are connected directly to corresponding signals at the EVCC.

> Note: It is important to insure that there be a good ground connection between the J1772 Ground and both the EV chassis / EVCC GND. This is not just good practice, but is required in order that the J1772 Pilot and J1772 Proximity signals work correctly. One way to insure that is to make sure that the charger enclosure itself has a good connection to EV chassis ground.

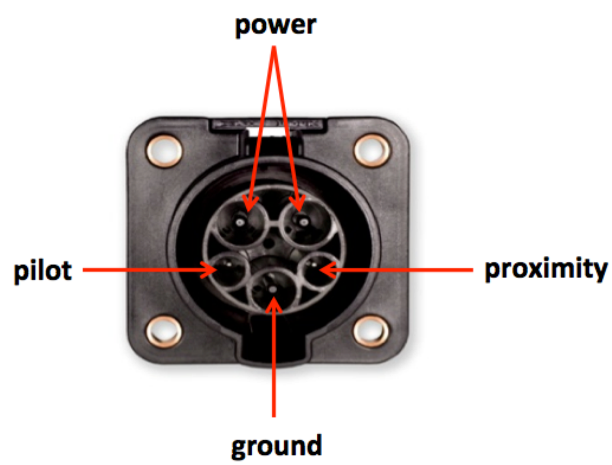


Figure 6 – Face of J1772 Socket

The **J1772 Proximity** signal allows the EV and the EVSE to determine whether the J1772 charge plug is "disconnected", "connected" or "locked". When the J1772 charge plug is fully inserted, it is "locked". When the charger release button is pressed (by thumb on the charger plug), the charge plug becomes "unlocked", or simply "connected". Should the plug become "unlocked" while charging, charging will immediately stop.

The **J1772 Pilot** signal is used by the EV to indicate to the EVSE that it is ready for charging. Using this signal, the the EVCC can enable and disable the relay in the EVSE that supplies line power to the charger.

The figure below shows the J1772 connections.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 7 – J1772 Connections

For more information on J1772 see http://en.wikipedia.org/wiki/SAE_J1772 and https://code.google.com/p/open-evse/wiki/J1772Basics).

Wiring Without J1772

Although J1772 is recommended, its use is optional. When using J1772, the EVCC **J1772 Proximity** signal is connected to ground through a 150 ohm resistor built into the J1772 charge plug to indicate that the plug is "locked".

When J1772 is not being used, the EVCC J1772 Proximity may be connected to GND through an external 150 ohm resistor directly. However, the EVCC is also tolerant of a direct (e.g., 0 ohm) connection to ground as well, and so the 150 ohm resistor is optional.

Here are two wiring options that do not use J1772:

Option 1 retains most EVCC functionality.

- Wire **J1772 Proximity** to **GND** through a switch (the "charger present" switch). To charge, plug in the charger, turn the "charger present" switch ON, and press **ChargeStart**. Charging operates as designed and the EVCC turns itself off when complete. The EVCC Drive mode operates as designed (**HotInRun** enables the EVCC, the cell loop operates the buzzer). If driveaway protection is implemented, the "charger present" switch must be turned OFF in order to operate the EV.

Option 2 is used when the EVCC is only used for charging.

- Wire **J1772 Proximity** directly to **GND**. Do not wire **Charge Start**. To charge, plug in the charger, and apply 12V to **HotInRun**. The EVCC will power up and begin charging. When the EVCC completes charging, it will stop sending CAN messages to the charger and turn off 12V_Ch, but will remain powered ON until power is removed from **HotInRun**. To start charging again, it is necessary to cycle power to the EVCC.

Cell Loop and Buzzer

The EVCC is intended to be installed with a Battery Management System that monitors per-cell over voltage conditions when charging and per-cell undervoltage when driving.

The EVCC Cell Loop surveillance circuit measures the resistance of the circuit between Cell Loop 1 and Cell Loop 2, if the circuit is open, then the cell loop is considered failed. The circuit applies +5v to Cell Loop1 and limits the current to about 2ma. It is expected that the Cell Loop be provided by a solid state relay or optoisolator. (Connecting the cell loop to the contacts of a mechanical relay is not recommeneded, as the cell loop current may not be enough "wetting current" for the relay contacts).

> WARNING: It is strongly recommended that per-cell monitoring be performed on the pack so that charging can be stopped if any cell exceeds a high voltage or low voltage cutoff. Lithium batteries can be dangerous if overcharged or undercharged.

The EVCC sounds the buzzer if the cell loop is open. The Buzzer output is connected to B+, fused to 350ma.

The figure below shows the Cell Loop and Buzzer connections.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 8 – Cell Loop and Buzzer Connections

Driveaway Protection

Driveaway Protection is a failsafe mechanism that prevents the EV being driven if the charger plug is connected. This feature is implemented by the relay contacts **EVSE Disc1** and **EVSE Disc2**. These contacts are fused to 350ma and are open if the J1772 cable is plugged in (or if the EVCC is not powered). Conversely, the contacts are only closed, and it is safe to drive, if the EVCC is powered up and the cable is not plugged in.

How to actually disable the EV from driving is not specified, however, the contacts could be wired into the control logic of the primary contactor.

> Note: The EVSE Disc1/2 contacts may not be suitable for directly control of a primary contactor. A typical primary contactor requires 1A or more of holding current which is well above the 350ma fused limit.

The figure below shows the connections used for Driveaway Protection.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 9 – Driveaway Protection Connections

Charge Cutback

Usually charging will be performed with the maximum current that the EVSE and Charger can support. In some cases (such as opportunity charging with a 110v outlet), it may be necessary to limit the maximum charge current to avoid tripping a circuit breaker. The Charge Cutback feature is designed for this case. To use this feature, it is first necessary to configure a maximum cutback current in the Command Line Interface. (Use the command "**set maxc_cb**").

Once configured, the Cutback signal is used to determine the charging current. If the Cutback signal is not grounded, then the maximum current specified ("**set maxc**"), is used; if the Cutback signal is grounded, then the maximum cutback current ("**set maxc_cb**") is used.

The diagram below shows the charge cutback connections.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 10 – Charge Cutback Connections

CANBUS

CAN is a robust communications protocol designed for automotive applications. CAN uses a two wire interface; the signals are designated CANH ("CAN high") and CANL ("CAN low"). A CAN network is a daisy-chain, multistation network that should be terminated on both ends of the string by 120ohm termination resistors. See below for a simple network diagram.

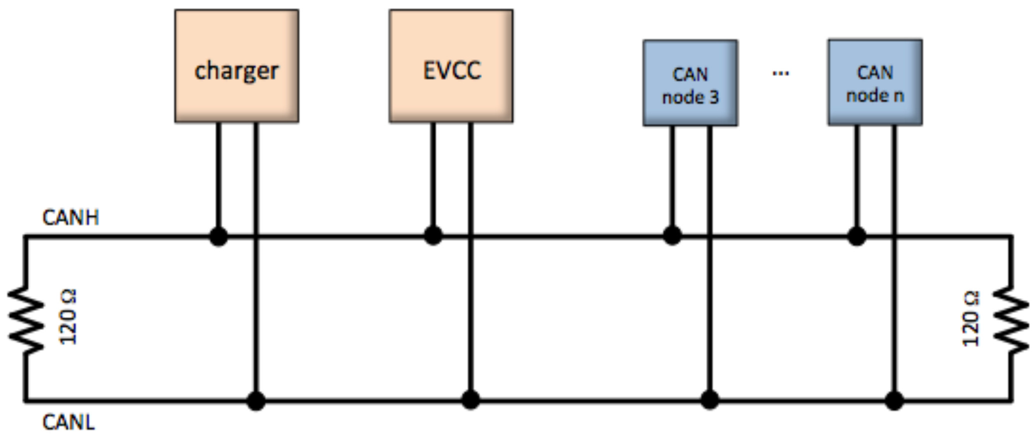


Figure 11 – CAN Network Diagram

CAN wiring should be kept short and the conductors should be twisted. Wiring should be placed away from EMI (ElectroMagnetic Interference) such as the motor and controller, and parallel runs next to the traction cabling should be avoided.

In a simple installation, there will be only two nodes on the CAN network: the charger and the EVCC, with a short and direct connection between the two. In this case, hand-twisted wiring should be fine.

For longer runs, more nodes, or cases where EMI may be an issue, shielded cable is desirable. If a shielded cable is used, the shield should be connected to chassis ground at a single place.

The figure below shows the connections used for CAN.

	A	B	C	D	E	F	G	H	J	K
1	B+	EVSE Disc1	EVSE Disc2	Charge Start	Cell Loop1	Serial Port	CANL	CANH	reserved	reserved
2	HotInRun	Buzzer	12V_Ch	J1772 Pilot	Cell Loop2		CANL	CANH		
3	GND	GND	GND	J1772 Proximity	Cutback		12V_Sw	GND		

Figure 12 – CAN Connections

Note that the EVCC supports a single CAN interface but brings out two sets of CANH/CANL pins on its connector. One pair (G1, H1) is wired to a CAN termination resistor in the harness. If it is necessary to extend the CAN network to add additional nodes, the resistor can be removed and the CAN string may be extended.

The EVCC supports a CAN data rate of 250Kbs and 11-bit CAN addressing. These parameters are not software configurable, however, both the CH4100 and ELCON chargers require this rate.

The EVCC uses two types of messages to control a CAN enabled charger. The first, from EVCC to Charger, provides the Charger with the allowable maximum values of charge voltage and charge current, and the second message, from Charger to EVCC that reports the actual Charging Voltage and Current (as well as additional charger status).

EVCC/Charger CAN messages are sent approximately twice a second, both from EVCC to Charger and from Charger to EVCC. If either the EVCC or the Charger does not receive these messages within a short time (on the order of a few seconds), the charging will terminate.

Charger progress messages can be logged to the serial port (using the command "**trace charger**"). There is also a low level "raw" trace ("**trace can**") that gives a hexadecimal dump of the raw message contents.

Configuration

Serial Port

This section describes how to install the serial port drivers and establish serial communications from a host computer and the EVCC. To use the serial cable, a Virtual Comm Port driver (VCP driver) and a terminal application (or "telnet client") is required.

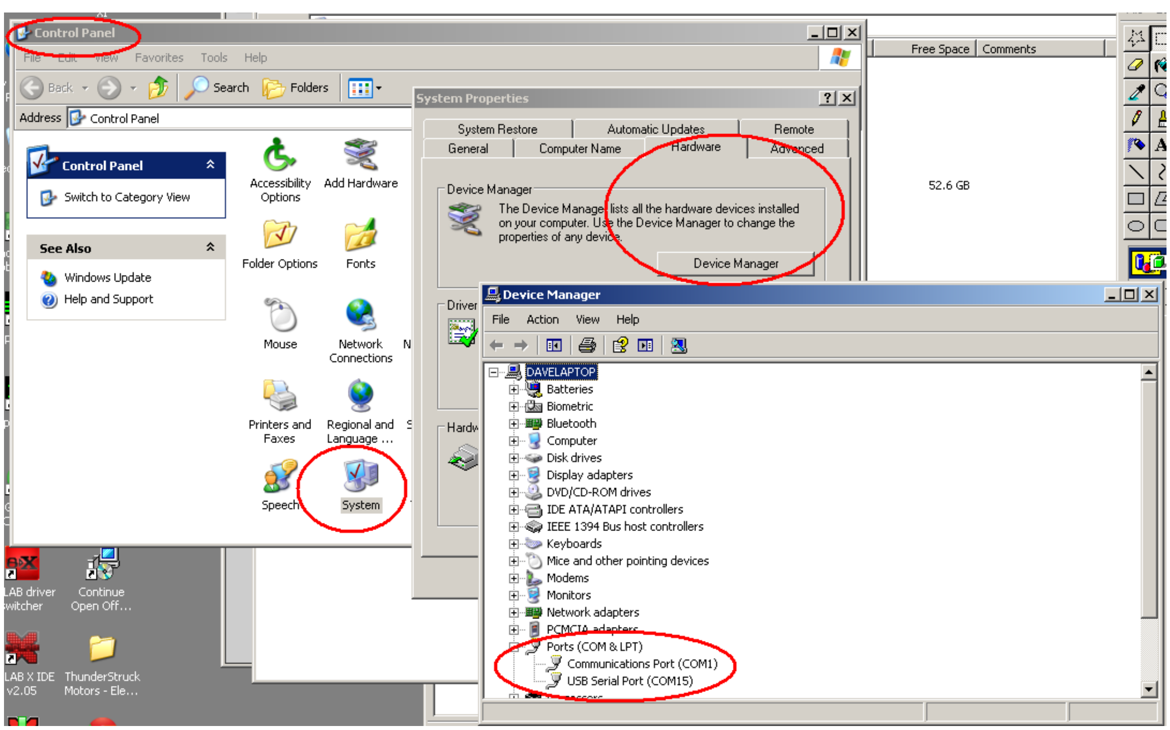
Using a USB to serial bridge is a generic and popular way to connect a host computer to a microcontroller, and the steps are basically the same regardless of the host computer and operating system. Detailed installation instructions are given below for Windows XP. See Mac OSX Support, below, for recommendations on how to enable the serial port on a MAC OSX machine. Note that there are good tutorials on how to install the necessary drivers and application software available on the Internet (for other versions of Windows, MAC, Linux, etc). (Search for "ftdi installation", "putty installation", etc).

Step 1: Install the Virtual Comm Port (VCP) driver on the host computer. The VCP driver is software on the host computer that emulates a serial port "on top of" a USB connection.

- VCP drivers are available at: www.ftdichip.com/Drivers/VCP.htm.
- Installation documentation is available at www.ftdichip.com/Support/Documents/InstallGuides.htm.

Step 2: Plug in the USB to serial port cable. If the drivers are correctly installed, the host computer will recognize the new virtual serial port device.; to use this device, is necessary to determine the virtual serial port device name.

- The virtual serial port device name is of the form "COM<n>", where n is a small number. This number can be determined by looking at "Control Panel -> System -> Device Manager -> Ports". In the example below, it is "COM15".

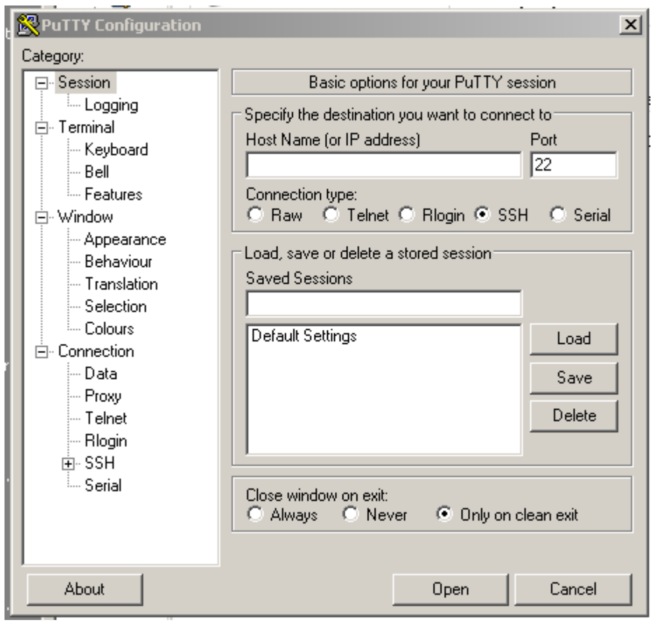


Step 3: Install a terminal console program (e.g., a "telnet client") on the host computer.

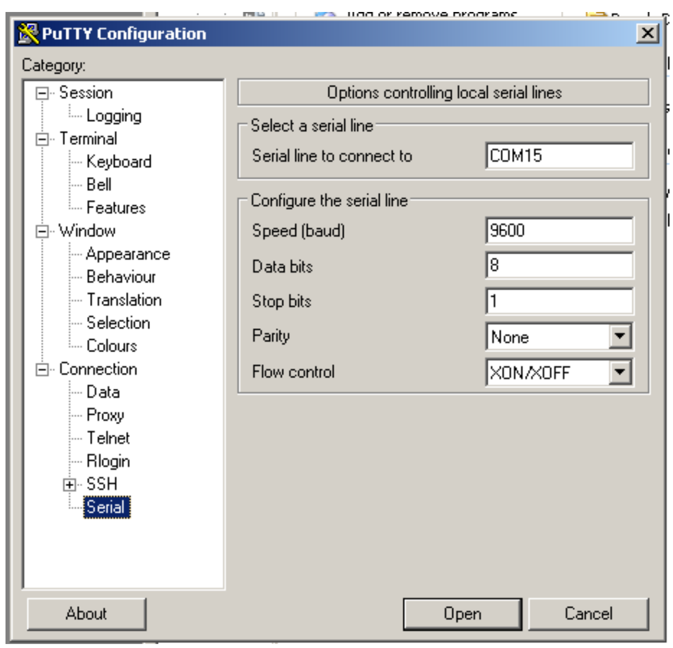
There are many suitable telnet clients that may be used. For Windows (and linux), one popular choice is PuTTY, available for download at http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html.

Step 4: Configure the telnet client for use.

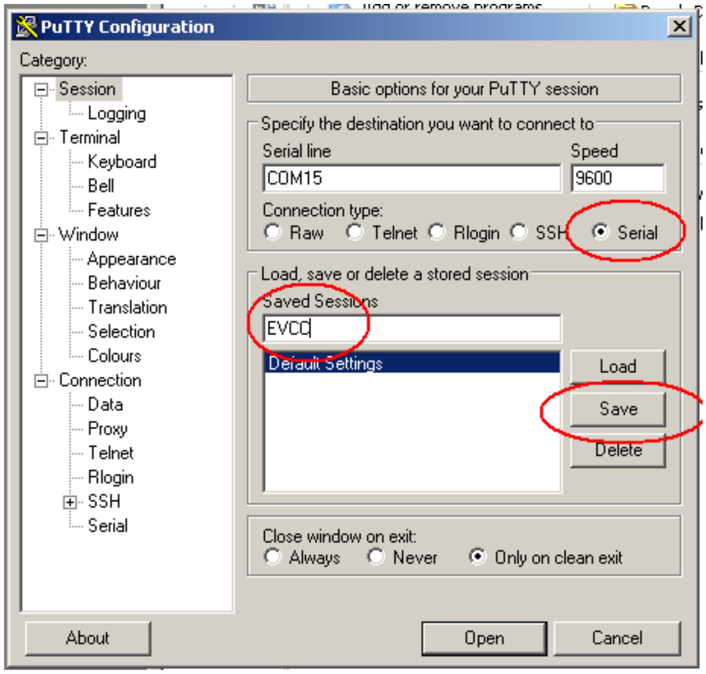
The first time PuTTY is opened, it will present the following:



Click on "Serial" in the Category column. Verify that the Speed is 9600, 8 data bits, 1 stop bit. Enter the Serial Line to connect to (in this case, "COM15").

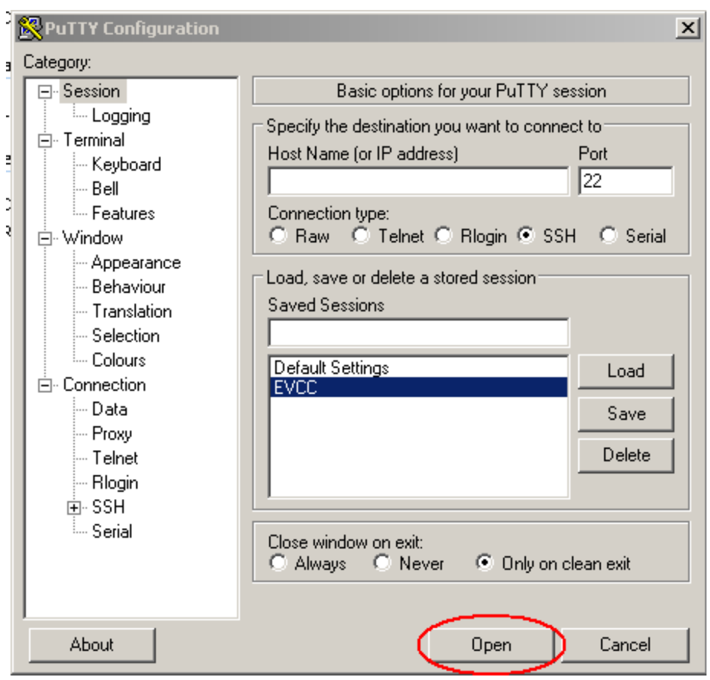


Do not hit "Open" just yet. Go back to "Session" by clicking the word "Session" in the Category window.

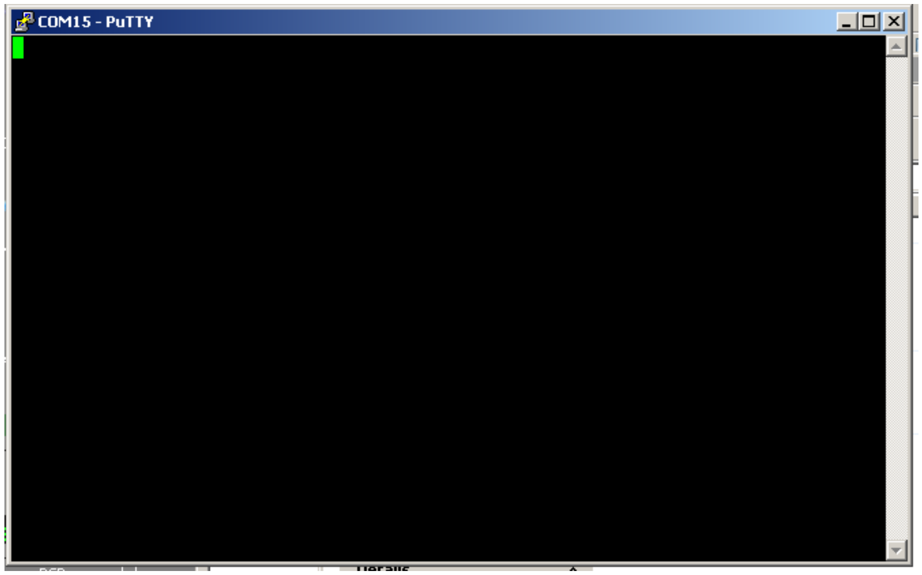


Set the Connection type to Serial. Give the new session a name (in this case "EVCC" in the Saved Sessions window) and press "Save" to save the session. PuTTY is now configured.

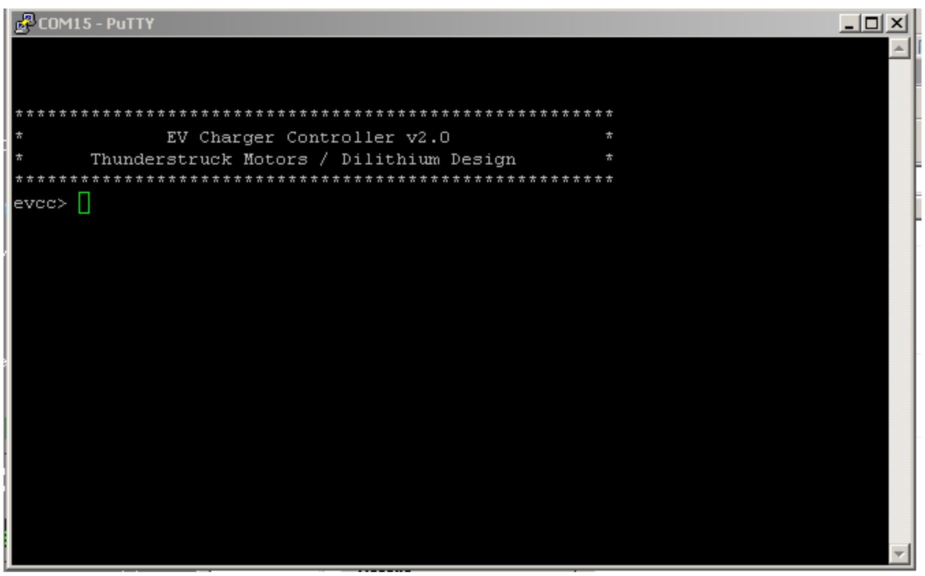
Step 4: Open the comm port. Select the saved session "EVCC" and click Open.



A screen like the following should appear:



Step 5: Connect the serial cable to the EVCC. Apply power to the EVCC by providing a 12V supply to **B+** and **GND**. Connect +12V to **HotInRun**. The EVCC **LED** should start blinking (assuming the cell loop has not been hooked up yet), and the following banner should be displayed:



Step 6: At this point, the EVCC may be configured. Configuration is stored in non-volatile memory and retained across a power cycle. See below, Command Line Interface, for details on what commands are supported and their syntax.

The EVCC is supplied with defaults, but at the very minimum, it will be necessary to set the Maximum Charging Voltage (using the command "**set maxv**") and Maximum Charging Current (using the command "**set maxc**").

> WARNING: Lithium batteries can be dangerous if overcharged and it is strongly recommended that the user check with their battery supplier to determine appropriate charging parameters.

A bringup checklist is provided below. The EVCC also has several diagnostic commands that can be used to verify proper wiring ("**measure**"), to trace can messages ("**trace can**"), to trace EVCC internal state changes ("**trace state**") and to trace charger operation ("**trace charger**").

LED Operation

The LED has the following operating states:

- Solid ON – Drive Mode
- Blink (once per second) – Charging
- Fast Blink (eight times a second) – Cell Loop Error

Charger Support

This section gives details on which chargers are supported by the EVCC.

CH4100

The CAN connections are found on the four pin connector J3. CANL is pin #8 (wired with a blue wire) and CANH is pin #9 (wired with a green wire). No other connections are required on J3.

> ➢ **Note**: The CH4100 charger contains an integrated termination resistor and supports point to point wiring directly from EVCC to charger.

See *CH4100 Series High Efficiency Intelligent Charger, User Manual Ver 1.5.3*.

ELCON

ELCON chargers must be ordered with or reprogrammed for the CAN option. An external ELCON provided CAN module is needed that terminates the CAN and then provides the serial interface for the charger. Only two pins are provided for the CAN connection: CANH and CANL.

> ➢ **Note**: The ELCON CAN module does NOT contain an integrated termination resistor. If the ELCON and the EVCC are connected by point to point wiring, it is recommended that a 120ohm termination resistor be placed across CANH and CANL as close as practical to the ELCON CAN module.

Bringup Checklist and Troubleshooting Hints

EV Installation

1) Connect B+, GND, HotInRun
2) Connect J1772 Proximity, J1772 Pilot, J1772 GND
3) Connect Cutback, if used

Verify Analog Inputs

1) Type “**measure”** with no parameters to get the expected readings for each analog input. Note that if there is not a good ground connection between J1772 ground and EV chassis ground that the J1772 readings will be erratic.
2) Verify Cell Loop, using “**measure loop**”
    a. Disconnect J1772 plug if connected
    b. Verify readings with cell loop open and closed.
3) Verify J1772 Proximity, using “**measure proximity**”
    a. Disconnect cell loop, if connected
    b. Verify readings with charger plug disconnected, connected, and unlocked.
4) Verify Cutback, if used, using “**measure cutback**”.
    a. Verify readings with cutback enabled and disabled

Verify Charge Start and J1772

1) Connect Cell Loop
2) Plug in J1772 Plug
3) Apply 12V to HotInRun. The EVCC should start charging (LED blinks once per second), 12V_Ch should be enabled, and the relay in the EVSE should operate after a short delay.
4) Assuming the CAN bus is not connected to the charger yet, the charge cycle should stop after 10-15 seconds.
5) Remove 12V from HotInRun, the EVCC should lose power (LED goes off).
6) Ground Charge start. The EVCC should power up and go into Charge state.
7) For debugging, use “**trace state**” to verify that the EVCC attempts to start charging if the J1772 plug is in and the user powers up the EVCC.

Verify Charger and CAN

1) Connect Charger to J1772, connect CAN between Charger and EVCC.
2) Now verify that when a charge cycle is started, that messages are exchanged between EVCC and Charger. (Use “**trace charger**” or “**trace can**” to log the messages).
3) If the pack is not yet connected to the Charger, the charge cycle will stop after a minute.

Systems Test

1) Verify all systems functions.

Command Line Interface

Startup Banner Message

When the EVCC is powered up, it will print the following:

```
*******************************************************
*              EV Charger Controller v2.0             *
*       Thunderstruck Motors / Dilithium Design       *
*******************************************************
evcc>
```

help

The **help** command prints out command help.

```
evcc> help
SHow [<>|Version|Config|History]
        <>         - status
        version    - firmware version
        config     - configuration
        history    - charge history
  SEt  [<>|CHARGER|MAXV|MAXC|MAXC_CB|TERMC|TERMT]
        <>         - show config
        charger    - charger type (one of ELCON, CH4100)
        maxv       - maximum charging voltage
        maxc       - maximum charging current
        maxc_cb    - maximum charging current (if cutback is enabled)
        termc      - termination charging current
        termt      - termination timeout
  REset [History]
        History    - reset charge history
  TRace [CHarger|CANbus|STate|OFF]
        <>         - trace toggle ON/OFF
        charger    - trace charger messages
        canbus     - trace canbus messages
        state      - trace EVCC state changes
        off        - disable all tracing
  MEasure [<>|LOOP|PROXimity|CUTback]
        <>         - help
        loop       - measure Cell Loop A/D
        proximity  - measure J1772 Proximity A/D
        cutback    - measure Cutback A/D
evcc>
```

In most cases, either a full version or an abbreviated version of a command (or command parameter) can be used. This is shown in the "help" with the use of uppercase and lowercase letters. For example, the abbreviation for **show** is **sh**, and the abbreviation for **show config** is **sh c**.

show

The **show** command displays configured parameters or status. If "show" is entered without parameters, current status will be displayed.

In the Drive mode, the EVCC monitors the cell loop and operates the buzzer when the cell loop indicates a pack fault.

```
evcc> show
  state    : DRIVE
```

```
  cell loop: OK
  proximity: EVSE not connected
  buzzer   : OFF
  charger  : not communicating
  uptime   : 0 hour(s), 0 minute(s), 33 second(s)
```

In the CHARGE mode, the EV is charging.

```
evcc> show
  state    : CHARGE
  cell loop: OK
  proximity: EVSE Connected and locked
  buzzer   : OFF
  voltage  :  147.7V
  current  :    5.9A
  charger  : 306 msgs sent; 320 msgs received
  uptime   : 0 hour(s), 3 minute(s), 30 second(s)
evcc>
```

Here is an example of CHARGE mode with Cutback is enabled:

```
evcc> show
  state    : CHARGE
  cell loop: OK
  proximity: EVSE Connected and locked
  cutback  : enabled
  buzzer   : OFF
  voltage  :  146.5V
  current  :    1.9A
  charger  : 349 msgs sent; 364 msgs received
  uptime   : 0 hour(s), 4 minute(s), 51 second(s)
evcc> show
```

show version

The **version** command displays firmware version number and build date.

```
evcc> show version
  version  :    v2.0; Sep 23 2014 12:04:16
evcc>
```

show config

The **show config** command displays configuration parameters.

```
evcc> show config
  charger  :  CH4100
  maxv     :   40.0V
  maxc     :    2.0A
  maxc_cb  :     n/a
  termc    :    0.2A
  termt    :4320 min
evcc>
```

These are

- `charger` - with v2.0, the ELCON and CH4100 chargers are supported
- `maxv` - maximum charging voltage (in Volts). This is provided to the charger.
- `maxc` - maximum charging current (in Amps). This is provided to the charger.

- `max_cb` - maximum charging current if cutback is enabled (in Amps). See text.
- `termc` - terminating charging current (in Amps). See text.
- `termt` - maximum charging time (in minutes). See text.

show history

The **show history** command displays data about the last sixteen charge cycles. See also **reset history**, below.

In the first example, the system has no charge history yet.

```
evcc> show history
no charge history
evcc>
```

The next example shows some charge history, with different “termination reasons”. The termination reason contains the reason that the charge cycle stopped. In the most recent charge attempt, the user disconnected the J1772 plug one minute after charging started. (EVSE disc, 1 mins). The previous attempt (“-1”) shows a normal charge completion with a charge time of 214 minutes and includes the number of watt hours delivered.

Note that the voltage and current measurements are provided by the charger in the CAN message to the EVCC. The EVCC does not measure pack voltage or current.

```
evcc> show history
      |    term    |   charge |    watt | maximum| maximum|  ending|
  num |   reason   |    time  |   hours | voltage| current| current|
----------------------------------------------------------------
 last | EVSE disc|    1 mins|      7Wh| 148.9V |   7.9A |   7.9A |
  - 1 |    normal | 214 mins|   3249Wh| 152.9V |   7.9A |   0.5A |
  - 2 | EVSE disc|    1 mins|      0Wh| 144.8V |   0.0A |   0.0A |
  - 3 | comm err |    0 mins|      0Wh|   0.0V |   0.0A |   0.0A |
evcc>
```

The full set of “term reason” codes is:

- `normal` - normal completion (charge current is less than terminating charging current)
- `cell loop` - a cell loop fault was detected
- `EVSE disc` - J1772 charge plug became unlocked while charging
- `comm err` - communications error with the charger
- `charger err` - charger has indicated an error
  One of: hardware, overtemp, pack voltage, input voltage, comm err
- `timeout` - the maximum charge time was reached
- `pack disc` - no pack was detected

set

This command sets the configurable parameters. For voltage and current, whole numbers (145) or decimal numbers (145.2) can be entered. The EVCC supports one decimal digit of precision.

The syntax of these commands is straightfoward, examples follow:

set charger

This sets the charger type. Either “ELCON” or “CH4100” can be entered.

```
evcc> set charger CH4100
```

set maxv

This sets the maximum charging voltage, in Volts.

```
evcc> set maxv 155.0
```

set maxc

This sets the maximum charging current, in Amps.

```
evcc> set maxc 8
```

set termc

This sets the termination charging current, in Amps. If the current drops below this setpoint then the charging stops.

```
evcc> set termc .5
```

set termt

This sets the maximum charging time, in minutes.

```
evcc> set termt 480
```

set maxc_cb

This sets the maximum cutback current, in Amps.

If the **Cutback** input is not enabled, maxc is used as the charging current. If the **Cutback** input is enabled, and if maxc_cb is nonzero, then the maxc_cb is used as the charging current.

```
ts> set maxc_cb 4
```

set

The **set** command with no parameters will show all the configured data. Note that this is the same as the command **show config**.

```
evcc> set
  charger  :  CH4100
  maxv     :  145.2V
  maxc     :    8.0A
  maxc_cb  :     n/a
  termc    :    0.5A
  termt    : 480 min
evcc>
```

reset history

The **reset history** command resets the charge history.

```
evcc> reset history
charge history has been reset
evcc>
```

trace

The **trace** command enables various forms of message or state tracing. These commands show a timestamp (uptime) and can be useful for logging or debugging. CHARGER, STATE, and CANBUS tracing may be independently enabled.

Trace configuration is stored in EEPROM and is present after reboot.

trace <>

Trace with no parameters toggles state trace on and off.

trace charger

The **trace charger** command displays messages from the charger. This trace also shows the current number of charging watts and the accumulated WattHours of charge.

```
evcc> trace charger
charger tracing is now ON
evcc> 00:08:22.7  V=148.6, A= 7.9, W=1173, Wh= 0.96
00:08:23.1  V=148.6, A= 7.9, W=1173, Wh= 1.12
00:08:23.6  V=148.6, A= 7.9, W=1173, Wh= 1.28
00:08:24.1  V=148.6, A= 7.9, W=1173, Wh= 1.45
00:08:24.6  V=148.6, A= 7.9, W=1173, Wh= 1.61
00:08:25.1  V=148.6, A= 7.9, W=1173, Wh= 1.77
00:08:25.6  V=148.6, A= 7.9, W=1173, Wh= 1.93
00:08:26.1  V=148.6, A= 7.9, W=1173, Wh= 2.08
00:08:26.6  V=148.6, A= 7.9, W=1173, Wh= 2.25
00:08:27.1  V=148.6, A= 7.9, W=1173, Wh= 2.41
00:08:27.6  V=148.6, A= 7.9, W=1173, Wh= 2.57
00:08:28.0  V=148.6, A= 7.9, W=1173, Wh= 2.73
00:08:28.6  V=148.6, A= 7.9, W=1173, Wh= 2.89
00:08:29.0  V=148.6, A= 7.9, W=1173, Wh= 3.05
00:08:29.6  V=148.9, A= 7.9, W=1176, Wh= 3.22
```

trace canbus

The **trace canbus** command displays canbus messages to and from the charger. Each line gives a timestamp, the originator of the message (if known), the CAN ID and CAN message contents, in hexadecimal.

```
evcc> trace can
canbus tracing is now ON
evcc> 00:05:47.6      evcc: 18e54024 fc dc 05 6c 0c ff ff ff
00:05:56.8    ch4100: 18eb2440 9c bf 2f fe 00 d1 0f d8
00:05:57.0      evcc: 18e54024 fc dc 05 6c 0c ff ff ff
00:05:57.2    ch4100: 18eb2440 04 fd b1 05 80 0c 56 ff
00:05:57.5      evcc: 18e54024 fc dc 05 6c 0c ff ff ff
00:05:57.7    ch4100: 18eb2440 00 fc b9 05 6d 0c 56 ff
00:05:58.1      evcc: 18e54024 fc dc 05 6c 0c ff ff ff
00:05:58.2    ch4100: 18eb2440 00 fc b9 05 6d 0c 56 ff
00:05:58.6      evcc: 18e54024 fc dc 05 6c 0c ff ff ff
```

trace state

The **trace state** command displays internal EVCC state transitions. It shows whether the EVCC is in DRIVE, CHARGE, or CHARGE/WARMDOWN, as well as the state of the J1772 charge plug.

Here is an example of state trace output that shows the charger plug being plugged in and unplugged.

```
evcc> trace state
state tracing is now ON
evcc> 00:06:53.4  old state=DRIVE, new state=CHARGE, j1772=LOCKED, term rsn=0
00:07:16.9  old state=CHARGE, new state=CHARGE/WARMDOWN, j1772=WAITING FOR DISC, term
rsn=EVSE UNLOCKED
00:07:17.2  old state=CHARGE/WARMDOWN, new state=CHARGE/WARMDOWN, j1772=DISCONNECTED,
term rsn=0
00:07:28.9  old state=CHARGE/WARMDOWN, new state=DRIVE, j1772=DISCONNECTED, term rsn=0
```

trace off

The **trace off** command turns off all tracing.

```
evcc> tr off
all tracing is now OFF
```

measure

The **measure** command is used to verify the A/D inputs. When this command is issued, the EVCC will repeatedly measure and print the value of an analog input. The command will run for 30 seconds and then automatically turn itself off. Alternately, the user can stop the command by typing any character.

The **measure** command with no parameters will display the expected values of the A/D inputs.

```
evcc> measure
This command repeatedly shows an analog input for 30 seconds.
Press any key to stop display

  The following values are expected
    loop      - Cell Loop A/D
                > 2.5V - OK
    proximity - J1772 Proximity A/D
                > 4.0V - disconnected
                > 2.5V - connected
                else   - locked
    cutback   - Cutback A/D
                < 4.0V - enabled

evcc>
```

measure loop

The **measure loop** command gives a real time measurement of the **cell loop**.

```
evcc> measure loop
evcc> Loop A/D= 4.97V
Loop A/D= 4.97V
Loop A/D= 4.97V
Loop A/D= 4.97V
Loop A/D= 4.97V
```

measure cutback

The **measure cutback** command gives a real time measurement of the **cutback** input.

```
evcc> me cutback
evcc> Cutback A/D= 4.99V
Cutback A/D= 4.99V
Cutback A/D= 4.99V
Cutback A/D= 4.99V
Cutback A/D= 4.99V
```

measure proximity

The **measure proximity** command gives a real time measurement of the **J1772 proximity** input.

In the example given below, both the **measure proximity** and **trace state** commands are enabled. Initially the J1772 charge plug is connected, then it becomes unlocked, and then finally, removed.

```
evcc> me prox
```

```
evcc> Proximity A/D= 1.50V
Proximity A/D= 1.50V
Proximity A/D= 1.50V
00:06:07.5  old state=CHARGING, new state=WARMDOWN, j1772=WAITING FOR DISC, term
rsn=EVSE UNLOCKED
Proximity A/D= 2.76V
Proximity A/D= 2.76V
Proximity A/D= 4.45V
00:06:12.0  old state=WARMDOWN, new state=WARMDOWN, j1772=DISCONNECTED, term rsn=0
Proximity A/D= 4.45V
Proximity A/D= 4.45V
```

Mac OSX Support

Before starting the procedure below, ensure the 12V power is hooked up to EVCC B+ and GND, and that 12V is connected to HotInRun. Finally, insure that the USB to serial cable is plugged into the computer.

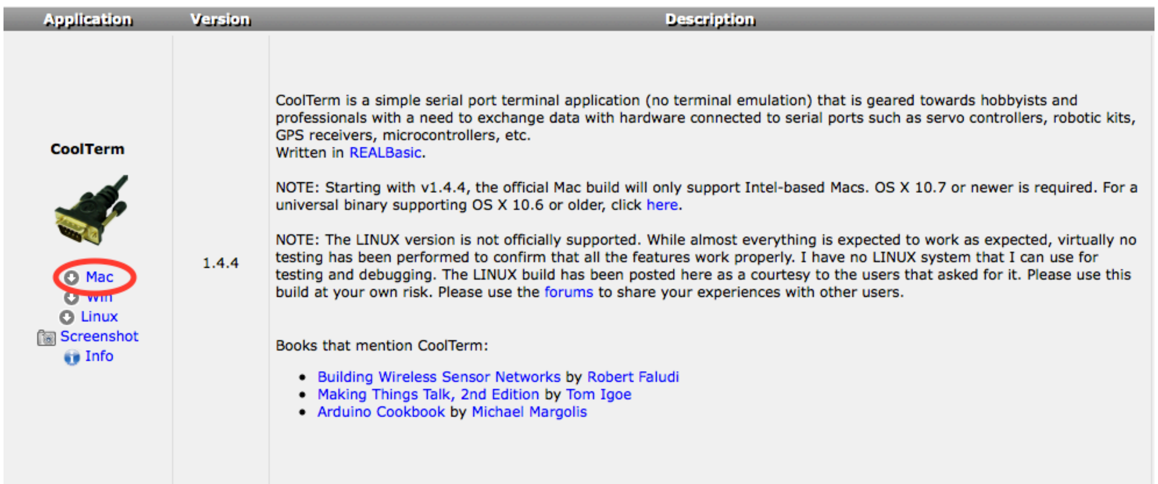
For MAC OS X, the virtual serial port device name is of the form "usbserial-<sn> where <sn> is the serial number of the USB to serial device. An example of what the name of the EVCC would look like is the following: usbserial-FTGDTR8M.

The MAC OSX distribution includes the applications "terminal" and "screen", which may be used. However, we have found that CoolTerm is simpler to install and use.

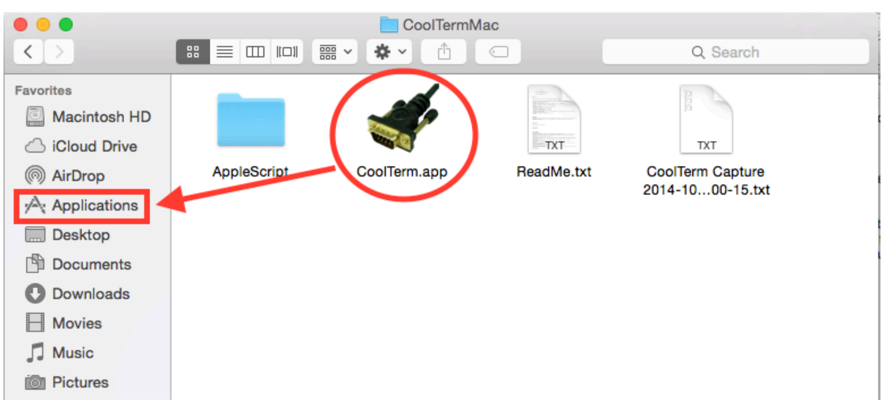
CoolTerm is a program that allows the user to easily access and program the EVCC via OS X.

1. Go to http://freeware.the-meiers.org

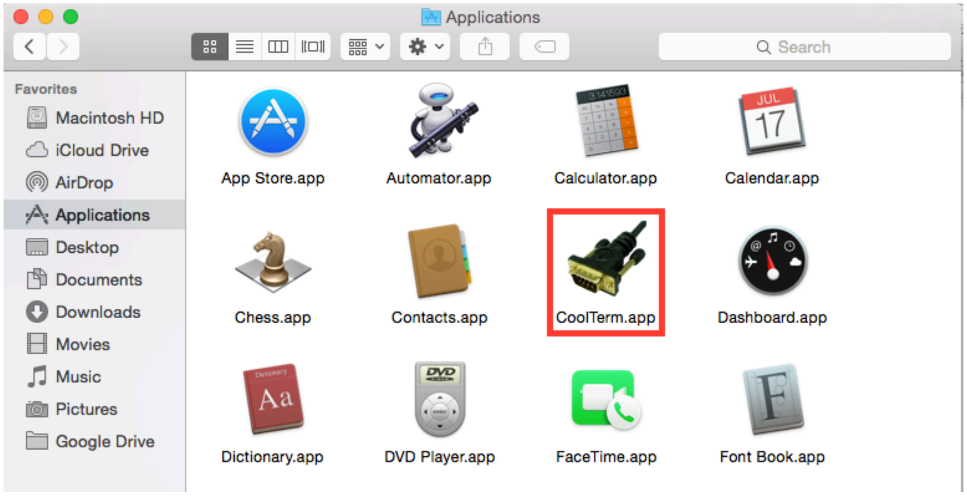
2. Click download for mac



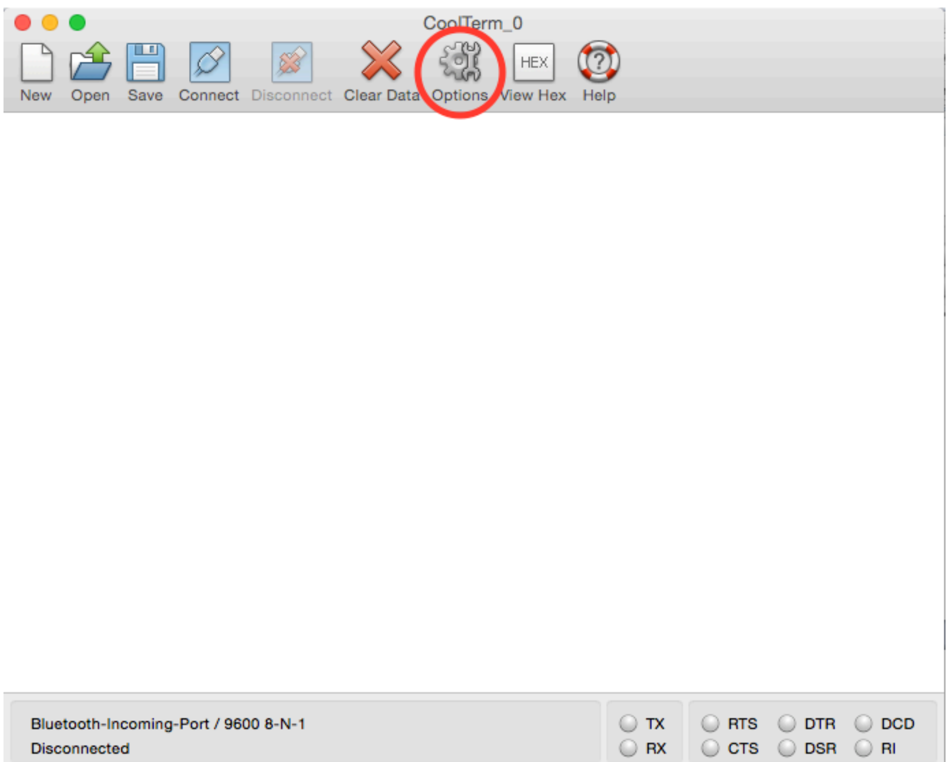
3. Extract the .zip file, open the CoolTermMac folder and drag the CoolTerm app into the applications folder.



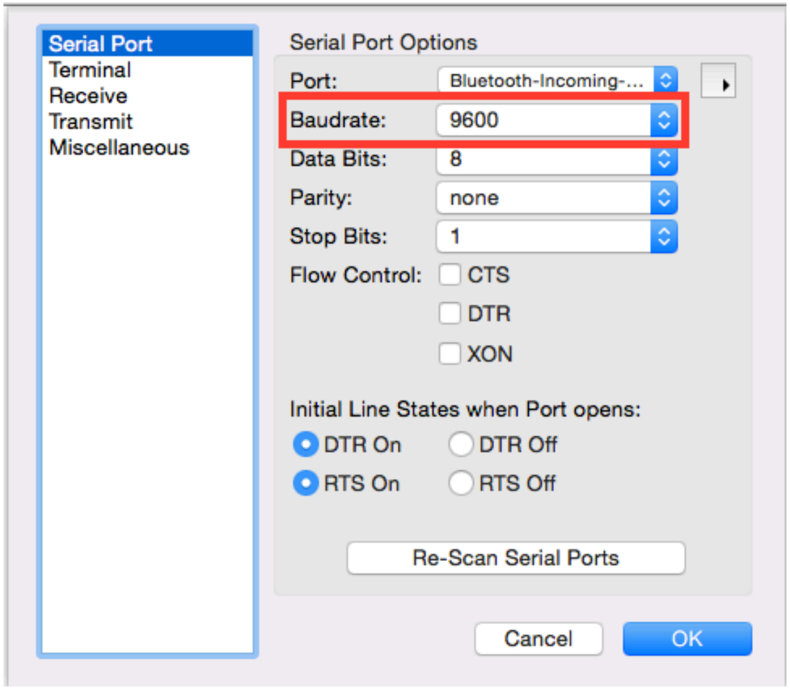
4. Open the applications folder and double click CoolTerm.app



5. Click “Options”

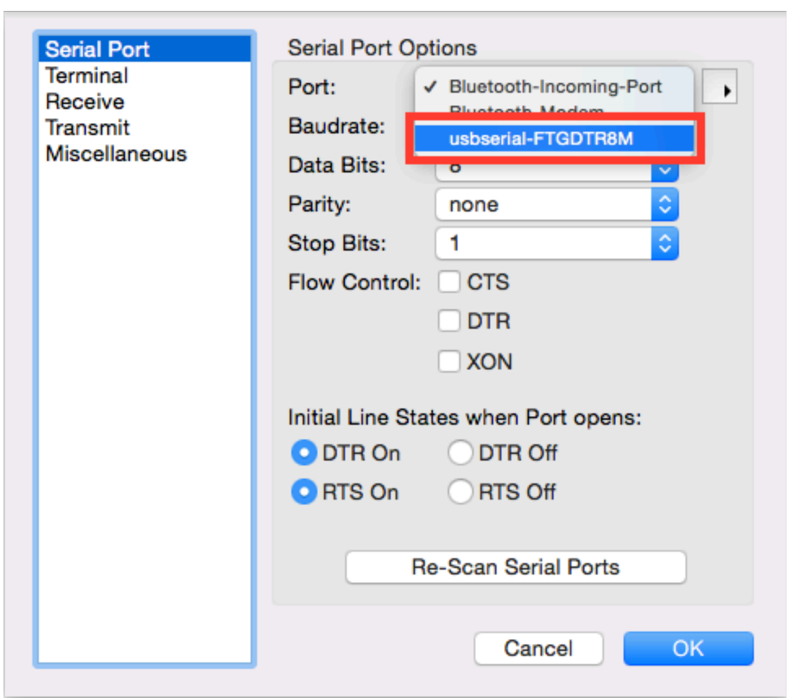


6. Ensure the “baudrate” is set to 9600 (which should already be set by default).

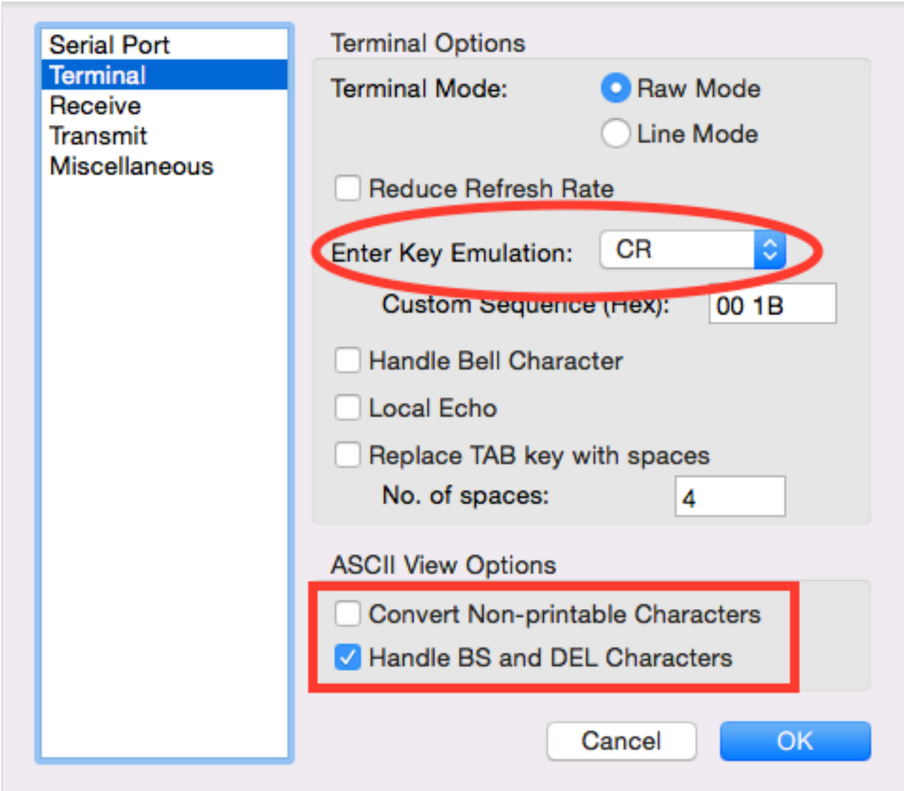


7. Click the drop down menu and select “usbserial-<sn>” where <sn> is the specific serial number of the EVCC as discussed earlier.

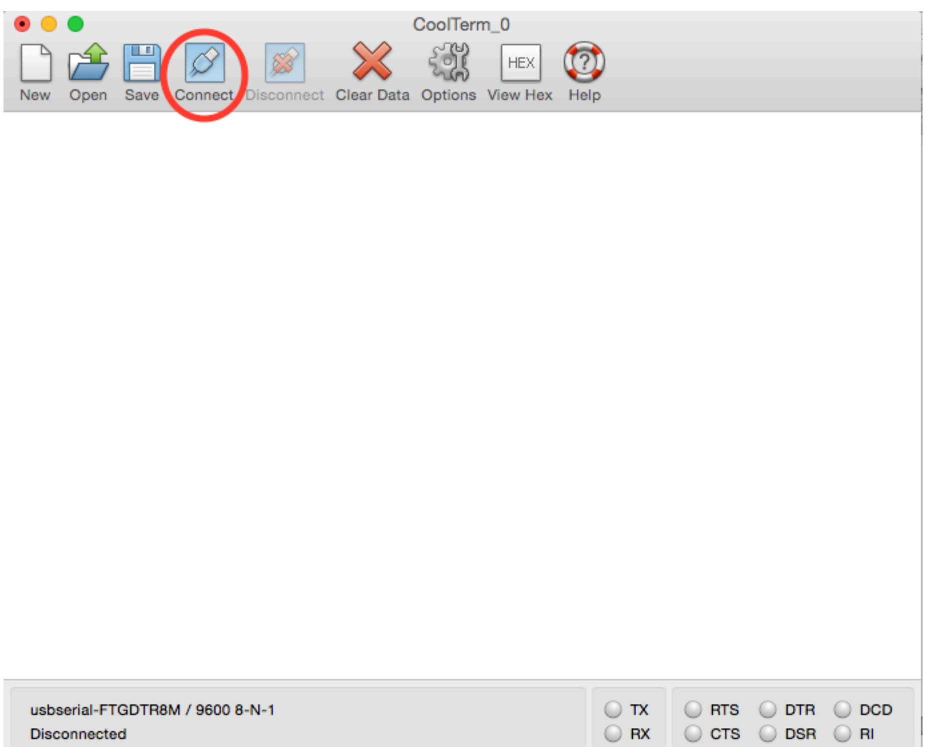
- **Note**: The usbserial-<sn> will not show up in the drop down menu if the USB is not plugged in prior to starting the program. If this occurs, exit CoolTerm, plug in the USB cable and restart CoolTerm.



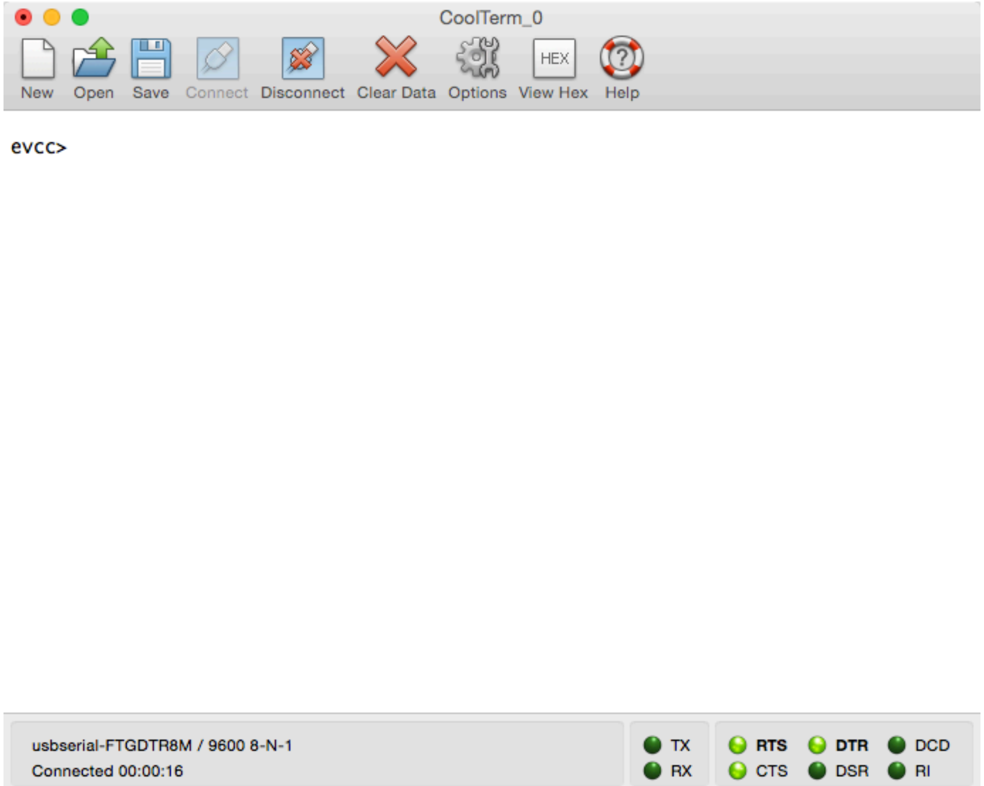
8. Still in “Options” go to the left hand column and click “terminal.” Then change the window to match the settings below.



9. Click “Connect”



10. Press the “return” key, the EVCC command prompt should come up.



- **Note**: Although the operation of the serial port is very similar to the Windows examples, above, there is one important difference. Windows keyboards generate an ASCII "DEL" character when a "delete" is pressed. MAC keyboards generate an ASCII "BS" character. Current EVCC firmware only interprets the DEL key and the MAC "delete" key may not work as expected. However, the ASCII "DEL" character can usually be generated by MAC keyboards (look for another "delete" key with an "x" or try pressing FN-DEL).

Warrantee and Support

The Thunderstruck return policy is available at http://www.thunderstruck-ev.com/return-policy.html.

The EV Charger Controller is warranted to be free from defects in components and workmanship under normal use and service for a period of 1 year.

When failing to perform as specified during the warranty period we will undertake to repair, or at our option, replace this product at no charge to its owner, provided the unit is returned undamaged and shipping prepaid, to Thunderstruck motors.

The product is intended for non-commercial use by hobbyists. The warranty does not apply to defects arising from miswiring, abuse or negligence, accidents, opening the enclosure, or reverse engineering. Thunderstruck Motors and Dilithium Design shall not be responsible for any incidental or consequential damages.

Thunderstruck Motors and Dilithium Design reserve the right to make changes or improvements in design or manufacturing without assuming any obligation to change or improve products previously manufactured and / or sold.

For general support and warrantee issues, contact
connect@thunderstruck-ev.com

For errors in this document, or comments about the product, contact
djmdilithium@gmail.com

Document History

Rev 2.0.0	Sept 22, 2014	In review
Rev 2.0.1	Sept 30, 2014	Production Version
Rev 2.0.2	Nov 10, 2014	Added Mac OSX serial support